Sweep Coverage with Mobile Sensors

Mo Li, *Member*, *IEEE*, Weifang Cheng, Kebin Liu, *Member*, *IEEE*,
Yuan He, *Member*, *IEEE*, Xiang-Yang Li, *Senior Member*, *IEEE*, and Xiangke Liao

Abstract—Many efforts have been made for addressing coverage problems in sensor networks. They fall into two categories, full coverage and barrier coverage, featured as static coverage. In this work, we study a new coverage scenario, sweep coverage, which differs with the previous static coverage. In sweep coverage, we only need to monitor certain points of interest (POIs) periodically so the coverage at each POI is time-variant, and thus we are able to utilize a small number of mobile sensors to achieve sweep coverage among a much larger number of POIs. We investigate the definitions and model for sweep coverage. Given a set of POIs and their sweep period requirements, we prove that determining the minimum number of required sensors (min-sensor sweep-coverage problem) is NP-hard, and it cannot be approximated within a factor of 2. We propose a centralized algorithm with constant approximation ratio 3 for the min-sensor sweep-coverage problem. We further characterize the nonlocality of the problem and design a distributed sweep algorithm, DSWEEP, cooperating sensors to provide efficiency with the best effort. We conduct extensive simulations to study the performance of the proposed algorithms. Our simulations show that DSWEEP outperforms the randomized scheme in both effectiveness and efficiency.

Index Terms—Sweep coverage, mobile sensors, dynamic coverage, DSWEEP.

1 Introduction

Wireless sensor networks have been widely studied for environment surveillance applications. In such applications, achieving specific coverage requirements is essential. There has been tremendous work done for different coverage problems in sensor networks under two main existing coverage scenarios, full coverage and barrier coverage. In full coverage [17], [21], [23], [25], sensors deployed over the field continuously monitor the entire area. Any point within the area is ensured to be covered by at least one or k sensors. A full coverage is required usually when users need to fully monitor the entire environment. In barrier coverage [5], [11], [20], sensors are deployed to form a barrier for detecting any intruders crossing the given strip area. Sensors cooperate to guard barrier coverage by covering the crossing paths. Barrier coverage is usually required for guarding safeties from intruders.

In either of the above two coverage scenarios, the monitored area requires being covered all the time, featured as static coverage. On the opposite, some applications set requirements with more dynamics along the time dimension. In a typical application of patrol inspection, we only need provide monitoring on certain Points Of Interest (POI) periodically instead of all along, which is featured as a *sweep coverage*. Sweep coverage differs with the static coverage, in the sense that in sweep coverage the coverage at each POI is time-variant as long as a coverage period is guaranteed. Therefore, directly applying traditional work under static coverage to the sweep coverage scenario is not feasible, suffering from poor efficiency and unnecessary extra overhead.

The concept of sweep coverage initially comes from the context of robotics. In this work, we investigate the sweep coverage problem in sensor networks. While the different objectives and different restrictions make the techniques proposed in the domain of robotics not applicable to the problem studied in this work, we propose lightweight and fully distributed solutions for sensor networks.

We propose a model for sweep coverage, in which each POI is covered by a sensor at a specific time instance *iff* the sensor is located at the position of the POI. A POI is t-sweep covered if it is covered at least once every t time units, and t is the sweep period of this POI. Different POIs could have different sweep periods. For periodical monitoring, we can utilize a small number of mobile sensor nodes [2] to achieve sweep coverage among a much larger number of POIs. If stationary sensors are deployed, much more sensors are required and they need not work most of the time, leading to significant waste of sensor nodes. In this scenario, we assume that all sensors are mobile, since the situation consisting of both stationary and mobile sensors can easily be reduced to scheduling mobile nodes for sweep coverage among those POIs not covered by stationary sensors.

Given the sweep coverage model with a set of POIs and the requirement of their sweep periods, a natural problem is to determine the minimum number of mobile sensors for required sweep coverage, which we define as min-sensor sweep-coverage. Unfortunately, we prove that this min-sensor sweep-coverage problem is NP-hard and it cannot be approximated within a factor of 2 unless $\mathrm{P} = \mathrm{NP}$. It is even

- *M. Li is with the School of Computer Engineering, Nanyang Technological University, N4-02c-108, 50 Nanyang Avenue, Singapore 639798, Singapore. E-mail: limo@ntu.edu.sg.*
- *W. Cheng and X. Liao are with the School of Computer, National University of Defense Technology, Changsha 410073, China. E-mail: {wfangch, xkliao}@nudt.edu.cn.*
- *K. Liu and Y. He are with TNLIST, School of Software, Tsinghua University, and also with the Computer Science Engineering Department, Hong Kong University of Science and Technology. E-mail: {kebin, heyuan}@cse.ust.hk.*
- *X.-Y. Li is with the Department of Computer Science, Illinois Institute of Technology, 10 West 31st Street, Chicago, IL 60616. E-mail: xli@cs.iit.edu.*

Manuscript received 5 Nov. 2009; revised 12 June 2010; accepted 1 Oct. 2010; published online 16 Dec. 2010.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2009-11-0483. Digital Object Identifier no. 10.1109/TMC.2010.237.

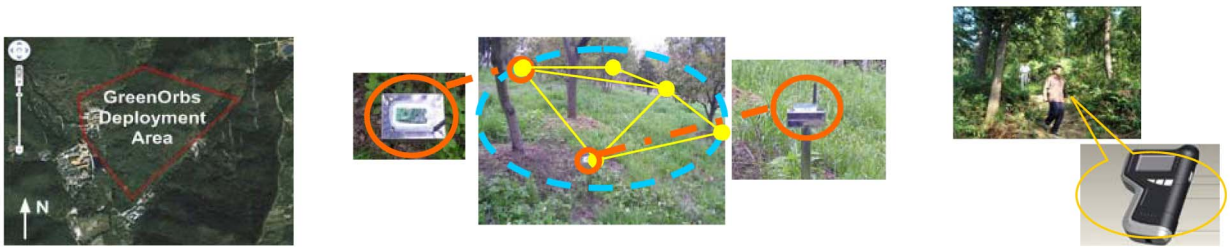


Fig. 1. GreenOrbs system: (a) the deployment area, (b) the system deployed in the field, and (c) the rangers with mobile devices.

challenging whether we can design a polynomial algorithm achieving constant approximation ratio. We further characterize the nonlocality of the sweep coverage problem, i.e., an individual mobile sensor cannot locally say "yes" or "no" to the question of whether a given set of POIs are globally t-sweep covered. As a result, how to design a sound distributed algorithm to cooperate the sensors achieving the sweep coverage efficiently is nontrivial.

We first target at a simplified min-sensor sweep-coverage problem where the sweep periods of all POIs are assumed to be identical. We propose a centralized sweep algorithm, CSWEEP, to schedule the sensors, which has an approximation ratio $2+\varepsilon$ for any $\varepsilon > 0$ on the minimum number of required sensors. Then we extend to general min-sensor sweep-coverage problem, and propose the GSWEEP algorithm, with an approximation ratio 3. In either CSWEEP or GSWEEP, the moving route of each mobile sensor is predetermined to guarantee the coverage. For practicability and scalability, we propose a distributed sweep algorithm, DSWEEP, which cooperates sensors efficiently to provide required coverage with the best effort. In DSWEEP, each sensor decides its moving path individually in runtime with the knowledge of the traces of others. Therefore, each sensor maintains a sweep table to save the swept POI ID and swept time. Sensors propagate their sweep tables to the network through the epidemic exchange. A filtered table exchange mechanism is utilized to omit transmitting most redundant table entries. Our simulations show that DSWEEP outperforms the randomized scheme in both effectiveness and efficiency.

The rest of this paper is organized as follows: Section 2 presents the background and motivation application for this study. Section 3 discusses related work. Section 4 describes the preliminaries on sweep coverage. We also prove the NP hardness of the min-sensor sweep-coverage problem and present the centralized algorithms. In Section 5, we present the design of DSWEEP, including the information exchange and local decision processes. We conduct the performance evaluation of DSWEEP in Section 6 and finally, we conclude this work in Section 7.

2 Background and Application

There has been increasing attention in the past decade from human beings on the earth's environment and climate change. In particular, forest, which is regarded as the earth's lung, is a major force in the battle against global warming. Regarding that, forest management and surveillance become important missions nowadays. A current project we have launched is GreenOrbs, a sustainable and large-scale wireless sensor network system in the forest [1], [26]. GreenOrbs aims at all-year-round ecological surveillance in the forest in Tianmu Mountain, collecting various sensory data, such as temperature, humidity, illumination, and content of carbon dioxide. The collected information can be utilized to support various forestry applications as follows:

2.1 Canopy Closure Estimates

Canopy closure is defined as the percentage of ground area vertically shaded by overhead foliage. It is a widely used significant forestry indicator but the traditional measurement techniques have either poor accuracy or prohibitive cost. Based on the readings of illumination sensors and Monte Carlo Theory, GreenOrbs realizes accurate and efficient canopy closure estimates of vast forest.

2.2 Research on Biodiversity

The sensor readings of temperature, humidity, illumination, and carbon dioxide, precisely characterize the forest microclimate. Those data, which quantify the biological activity and multispecies competition, can be utilized to support research on biodiversity.

2.3 Carbon Sequestration

To maximize the utility of forest carbon sequestration, the capacity of carbon sequestration of different tree species need to be accurately measured. This can be realized with carbon dioxide sensors in the 3D forest space. By comparing the sensor readings at different heights, the amount of carbon dioxide a tree canopy absorbs can be continuously monitored.

2.4 Fire Risk Prediction

Using the sensor data in the forest, namely temperature and humidity, GreenOrbs continuously monitors the environmental, supporting fine-grained real-time fire risk prediction.

The first GreenOrbs deployment was carried out in July 2008. Ever since then, GreenOrbs has experienced a number of deployments at different places, with different scales, and for different durations. The most recent deployment includes up to 330 nodes and till now has consistently worked for nearly one year. Fig. 1a depicts the deployment area of this system in Tianmu Mountain. Fig. 1b depicts the stationary sensor nodes we deployed within the timberland. The sensor nodes cooperatively work together to consistently monitor the critical parameters in the forest. Beside the basic surveillance requirement, we are planning a framework that integrates the stationary sensor network

with other ubiquitous technologies, making GreenOrbs system more pervasive.

In particular, for the application of fire risk prediction, we are not only using stationary sensor nodes. We also plan to equip currently employed hundreds of rangers with mobile handheld devices and let them patrol around in the forest. Fig. 1c depicts the mobile device they are currently using. The mobile device can provide the rangers location references from GPS as well as long distance communication between each other. The rangers are highly professional in detecting the early indications in fire risk prediction. With the equipped devices, they can actually be treated as mobile sensors. There are some important places in the forest that are critical or of high potential in taking fire. Letting rangers periodically visit those places is significant in early detecting any possible trouble and securing the forest from fire. It is exactly an instance of the sweep coverage problem. Such an application motivates our study.

3 Related Work

The coverage problem has been a hot issue in wireless sensor networks. Many efforts have been made on the full coverage problem, such as area coverage [29] and point coverage [9]. There has been some work using mobile sensors to assist static coverage under a hybrid network architecture [33], [36]. Wang et al. investigate the optimized movement of mobile sensors to provide k-coverage in both mobile sensor networks and hybrid sensor networks [36]. The authors in literature [33] propose a distributed relocation algorithm, where each mobile sensor only requires local information to achieve optimal relocation. They explore the potentials of mobile sensors to extend the network lifetime. Also many researchers study the coverage of mobile sensor networks. Howard et al. [16] propose a potential-field-based algorithm and ensure that the initial configuration of nodes quickly spreads out to maximize coverage area. Wang et al. [34] present another virtual-force-based sensor movement strategy to enhance network coverage after an initial random placement of sensors. Sensor nodes are redeployed according to the virtual force calculation. They also consider the coverage holes in the network and move sensors to the desired target positions in order to improve the coverage [35]. Above algorithms aim to spread sensors over the field in a stationary configuration to maximize the coverage area. A complete survey of the full coverage problem is provided by Cardei and Wu [10].

Kumar et al. extensively study the barrier coverage problem [5], [11], [20], where the sensors form a barrier to prevent intruders from crossing a thin strip. The work in literature [20] is the first one to study the theoretical foundations of barrier coverage. A localized algorithm providing local barrier coverage is proposed in literature [11]. Balister et al. [5] further derive reliable density estimates for achieving barrier coverage and connectivity in thin strips.

Most of existing work focuses on static coverage with stationary configurations of the sensors. Even with mobile sensors, they mostly focus on achieving an optimized deployment through their mobility without exploring the dynamic coverage. Obviously, the results and approaches of the work do not directly apply to the sweep coverage scenario. One previous work [24] studies the dynamic aspects of the coverage in a mobile sensor network. It shows that while the area coverage at any given time instance remains unchanged, a larger area will be covered during a time interval. The targets that not detected in a stationary sensor network can now be detected by moving the sensors. However, it focuses on providing coverage for the full area and does not consider the sweep coverage scenario.

The concept of sweep coverage initially comes from the context of robotics [6], [15] which mainly concerns the metric of coverage frequency, i.e., the frequency of the coverage of each point. Robots coordinate or randomly move on the field and deploy communication beacons in the environment to mark previously visited areas. Robots then make local decisions on their motion strategy through communications with those beacons. The techniques proposed in the domain of robotics cannot be directly applied to sensor networks due to the highly integrated intelligence and costly hardware requirements of robots. Some works focus on multirobot-based coverage [7], [12], [14]. Though similarly targeting at the coverage issues, those works mainly consider using robots to achieve the area coverage in the field. The aim of multirobot-based coverage is usually exhaustively exploring the target area for one time with multiple robots. The proposed approaches are not fully distributed. Some of them assume that communication between all robots is always available without any physical restrictions. Some of them divide robots into independent teams for different areas, and robots in the same team share all coverage information [19], [22]. The different objectives and different restrictions make the solutions of those works not applicable to the problem studied in this paper. To the best of our knowledge, this work is the first to introduce the sweep coverage in sensor networks which builds the theoretical foundation and proposes practical protocols.

There are some research efforts in robotics community that are aimed to address robot path planning or motion planning problems that shares some similarity with this work [8], [18], [28], [31]. The proposed solutions try to minimize the path length for the robot to travel from origin to destination while avoiding any obstacles within the field. Some works consider multirobot systems where multiple robots exist in the same field and the ordering of the robots' movements is carefully considered to prevent them from colliding with one another [27]. The focuses of those works, however, are essentially different from what we are going to address in this paper. The goal of robot path planning is to navigate robots from start points to destination points with a field of obstacles, while the goal of this work is to circulate mobile sensors among a set of POIs such that the coverage requirement for those POIs is fulfilled. The major research challenge for robot path planning resides on how the robots find optimal paths that bypass all the obstacles with the shortest distances, while the major research problem in this paper is to find an optimal schedule for mobile sensors such that a small number of sensor nodes suffice to provide sweep coverage over all the POIs. For robot path planning, full knowledge of the field is usually assumed instantly available to the decision maker, while in

this paper the mobile sensors are operated in a distributed manner. Mobile sensor nodes have to acquire instant sweep coverage statuses of POIs through information exchange and such knowledge is usually incomplete. All such differences make existing solutions for robot path planning hardly usable to our problem.

4 The Sweep Coverage Problem

In this section, we first give some definitions of sweep coverage problem. We prove the NP hardness of determining the minimum number of sensors to provide required sweep coverage (min-sensor sweep-coverage problem). We find that this problem cannot be approximated within a factor of 2 unless $\mathrm{P} = \mathrm{NP}$. We then propose centralized approximation algorithms against the min-sensor sweep-coverage problem with constant approximation ratio. At the end of this section, we characterize the nonlocality property of sweep coverage problem.

4.1 Sweep Coverage

Assume that n mobile sensors $S = \{s_1, s_2, \ldots, s_n\}$ are (randomly or strategically) utilized to monitor m points of interest $H = \{h_1, h_2, \ldots, h_m\}$ in a region.

Let $d_{i,j}$ be the euclidean distance between POI h_i and h_j. We assume that all mobile sensors will move at the same speed v. At a specific time instance, a POI is covered by a sensor *iff* the sensor is located at the position of that POI. We assume that all sensors are mobile, since the situation consisting of both stationary and mobile sensors can easily be reduced to scheduling mobile nodes for sweep coverage among those POIs not covered by stationary sensors.

Sweep coverage is different with traditional full coverage or barrier coverage in which users need provide static and continuous coverage all the time. In sweep coverage we only require that the POIs are covered at least once every certain time interval, so that we can guarantee event detection within a certain delay bound. Based on this, we define t-sweep coverage as follows:

Definition 1 (t-sweep coverage). *A POI is said to be t-sweep covered by a coverage scheme F iff it is covered at least once every t time units by the mobile sensors scheduled by F.*

Coverage scheme F is a schedule of mobile sensor movement under the constraint of their moving speeds. Since we assume that all the mobile sensors move at a constant speed, a coverage scheme F can be simply represented by the moving trajectories of all sensors, one trajectory for one sensor. If a POI is t-sweep covered, time interval t is called the *sweep period* of the POI. In practice, different POIs may have different sweep period requirements. We assume that the POI h_i needs to be covered once every t_i time units. In such a case, the POI h_i is said to be t_i-sweep covered.

Definition 2 (Global sweep coverage). *A set of POIs are said to be globally sweep covered by a coverage scheme F iff every POI hi is ti-sweep covered under F.*

When $t_i = t$ for all POIs, it becomes a simplified problem we call *global t-sweep coverage*.

4.2 Problem Hardness

The most fundamental problem we concern is, given a set U of POIs, the minimum number of mobile sensors with a moving speed v such that there is a coverage movement schedule F that can satisfy the required global sweep coverage under the t_i-sweep coverage constraints for each POI. We denote this problem as min-sensor sweep-coverage problem (U, t, v), where t is a vector denoting the coverage requirement of POIs. We show by Theorem 1 that the min-sensor sweep-coverage problem is NP-hard by a reduction from the Traveling Salesman Problem (TSP). Let $OPT(U, t, v)$ be the minimum number of mobile sensors with moving speed v needed to cover the set U of POIs for t-sweep coverage.

Theorem 1. *Given a set U of POIs and their sweep coverage time-period requirement t, and the moving speed v of sensors, determining the minimum number of required mobile sensors such that the problem (U, t, v) has a valid coverage scheme is NP-hard. Unless $\mathrm{P} = \mathrm{NP}$, there is no polynomial time algorithm that can always find a coverage scheme using only $(2-\varepsilon) \cdot OPT(U, t, v)$ sensors for the min-sensor sweep-coverage problem (U, t, v).*

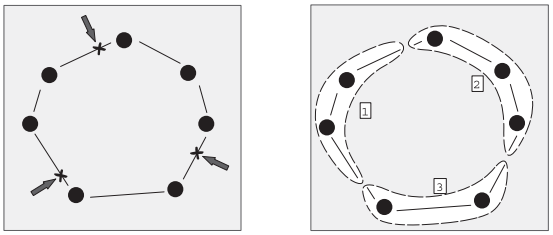
Proof. To prove the NP-hardness of the min-sensor sweep-coverage problem, we reduce the TSP problem to the min-sensor sweep-coverage problem as follows:

For a TSP problem, we are given a set of m sites $U = \{u_1, u_2, \ldots, u_m\}$ in a 2D domain, TSP seeks the shortest route to visit all sites once and return to the starting point. The corresponding decision problem TSP (U, L) asks whether there is a cycle connecting all m sites with length not exceeding a given value L. Given a decision problem of TSP (U, L), we define a min-sensor sweep-coverage problem accordingly: the POIs are the m sites $U = \{u_1, u_2, \ldots, u_m\}$, and the sweep period t_i of each POI is L/v , where v is the moving speed of mobile sensors.

Apparently, if the given TSP problem (U, L) has a solution, then one sensor is enough to provide L/v-sweep coverage for the corresponding min-sensor sweep-coverage problem[1]: the cycle that visits all sites defines a moving scheme F such that all sites will be visited by this sensor at least once every L/v time units. On the other hand, if the min-sensor sweep-coverage problem has a solution of using only one sensor, the decision problem of TSP has a yes solution. Notice that for any interval of $t = L/v$ time units, each site must be visited at least once by this sensor during this time interval by the coverage scheme F. This implies that the scheme F provides a route such that all sites are visited at least once. Obviously, the total length of this route is at most $L/v \cdot v = L$.

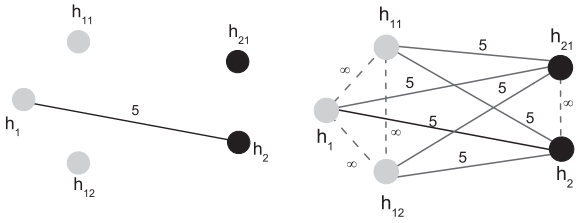
The above reduction proves that the min-sensor sweep-coverage problem is NP-hard. We then show that this problem does not have any polynomial time algorithm with approximation ratio $\leq 2 - \varepsilon$ for an arbitrary constant $\varepsilon > 0$, unless $\mathrm{P} = \mathrm{NP}$. For the sake of contradiction, assume that such a polynomial time approximation algorithm exists, denoted by APPR. Consider the decision TSP (U, L) with L as the length

1. In other words, the solution to this min-sensor sweep-coverage problem is 1.



(a) (b)

Fig. 2. The illustration of CSWEEP algorithm. (a) All the POIs are connected by the route computed by approximation algorithm PTAS of TSP. Then, this route is divided into three equal pieces. (b) Each mobile sensor is assigned to move continuously on one individual piece of route back and forth and monitors the POIs on its route.



(a) (b)

Fig. 3. The illustration of duplicating POIs in GSWEEP. (a) The link weight between h_1 and h_2 is weighted at their distance 5. Two virtual POIs are derived for h_1, denoted by h_{11} and h_{12}. One virtual POI is derived from h_2, denoted by h_{21}. (b) A complete graph among all POIs and virtual POIs is created. The link weights of the clique among $\{h_1, h_{11}, h_{12}\}$ are set to be ∞, and so does h_2 and h_{21}. Other link weights are duplicated from $d_{1,2}$.

of the optimum route for TSP. Then, the corresponding min-sensor sweep-coverage still has optimum solution with one sensor. For this special min-sensor sweep-coverage problem, the number of sensors found by APPR will be at most $(2-\varepsilon)\cdot 1$. Notice that the number of sensors used must be an integer. It implies that the optimum solution for min-sensor coverage problem is 1, and this solution can be computed in polynomial time. This implies that the original TSP problem has a yes solution. Recall that, it is NP-hard to decide whether the decision TSP, with L as the length of the optimum route for TSP, has a yes solution. This finishes the proof. □

4.3 CSWEEP Algorithm

For the min-sensor sweep-coverage problem, global t-sweep coverage is a simplified case where $t_i = t$. For such case we design a centralized sweep algorithm (CSWEEP), which is derived from the approximation algorithm of the TSP problem.

For the euclidean TSP problem, there is a well-known polynomial time algorithm [4], PTAS, with the best approximation ratio $1+\varepsilon$. We begin with this algorithm. First, we create a weighted complete graph using the given POIs as vertices, and the link weights is just the distance between two POIs. We input this graph into PTAS for the TSP problem. Then, the output is a suboptimal route P for the corresponding TSP. Here every POI appears just once on P in the TSP problem. We partition route P into equal pieces with length $L_0 = v\cdot t/2$ as shown in Fig. 2a if L_0 is less than the length of P. Otherwise, one sensor is enough. Then, we let each mobile sensor move continuously on one individual piece of route back and forth as shown in Fig. 2b. As a result, each POI located on one piece of route will be visited at least once every $2\cdot L_0/v = t$ time units. Indeed, each mobile sensor does not necessarily need to move on the loose end of the route segment, resulting in a smaller sweep period for the POIs. By this way, every POI is t-sweep covered and the set of POIs are globally t-sweep covered. Thus, the total number of needed mobile sensors is exactly the number of subdivided pieces.

By Theorem 2, we further show that CSWEEP has an approximation ratio of $2+\varepsilon$.

Theorem 2. *For the min-sensor sweep-coverage problem, the approximation ratio of CSWEEP algorithm is at most $2+\varepsilon$ for arbitrary $\varepsilon > 0$ the time complexity of the CSWEEP algorithm depending on $1+\varepsilon$.*

Proof. First, taking the POIs of the min-sensor sweep-coverage as sites in TSP, we have the corresponding TSP problem. We assume that the length of optimal route for the TSP problem is L. Then the length of route P is at most $L' = L\cdot(1+\varepsilon)$ where ε is an arbitrarily small positive constant used by the PTAS, since PTAS has an approximation ratio $(1+\varepsilon)$. Thus, the route P shall be divided into $\lceil L'/L_0\rceil = \lceil 2L(1+\varepsilon)/(v\cdot t)\rceil$ pieces in CSWEEP. As shown above, in CSWEEP we assign each mobile sensor an individual piece of route. Then, the number of mobile sensors required in CSWEEP is $N_{cen} = 2L(1+\varepsilon)/(v\cdot t)$. Second, we assume the optimal solution of min-sensor sweep-coverage problem is N_{opt}. In other words, there is a coverage scheme F and according to scheme F, if we use N_{opt} sensors moving at constant speed v, each POI will be visited at least once in t time units. As L is the length of the shortest route for corresponding TSP problem, we get the following inequation $N_{opt}\cdot v\cdot t \geq L$, leading to $N_{opt} \geq \lceil L/(v\cdot t)\rceil$. Finally, the approximation ratio of CSWEEP is calculated $N_{cen}/N_{opt} = 2+\varepsilon$. This finishes the proof. □

4.4 GSWEEP Algorithm

For the general case of min-sensor sweep-coverage problem, the sweep periods of different POIs might be different. Therefore, the above approximation cannot apply to such case and we design a general approximation algorithm, GSWEEP, executed in three steps.

Step 1. Duplicating the POIs. For each POI *hi*, we calculate its monitoring frequency $f_i = 1/t_i$. If f_i is not an integer, we convert it to integers by ceiling. Then, we can compute the greatest common divisor of all the frequencies $f = \gcd(f_1, f_2, \ldots, f_m)$. For each POI h_i, we create $k(i) = f_i/f - 1$ virtual POIs for it, denoted by $H_i = \{h_{i1}, h_{i2}, \ldots, h_{ik(i)}\}$. As shown in Fig. 3a, two virtual POIs, h_{11} and h_{12}, are derived for h_1. One virtual POI is derived from h_2, denoted as h_{21}. For all POIs and their virtual POIs, we create a weighted complete graph. First, the link weight between h_i and h_j is set the same as their distance $d_{i,j}$. All the link weights of the clique among h_i and POIs in H_i are set to be ∞, which implies that those links with ∞ weight do not exist in practice in the following algorithms. Whereas, the link weights for members of H_i between h_j and members of H_j are just duplicated from link weight between h_i and h_j.

As depicted in Fig. 3b, the link weights of the clique among $\{h_1, h_{11}, h_{12}\}$ are set to be ∞, so does the link weight between h_2 and h_{21}. All the remaining link weights are set to be five, duplicated from link weight between h_1 and h_2. In the following steps, we consider the virtual POIs the same as POIs.

Step 2. Finding a TSP route P. Since the above weighted graph is not a geometric graph, we cannot use the approximation algorithm PTAS to address the TSP problem on this graph, but with the help of Christofides algorithm [13], we can find a route P for this problem with an approximation ratio 1.5, having a time complexity of $O(m^3)$ where m is the number of POIs. Notice that route P visits every POI just once and POI h_i has additional *k(i)* duplicates on route P.

Step 3. Partitioning the route P. Similar with CSWEEP, we partition route P into some equal pieces, which have the length $L_0 = v/2f$. Then, we assign each piece of route one sensor moving on back and forth. In result, we can guarantee that all POIs including the virtual POIs on the route can be visited at least once in $1/f$ time units. Since POI *hi* has additional *k(i)* duplicates on route P, then *hi* can be visited at least $k(i) + 1 = f_i/f$ times in $1/f$ time units. Therefore, during t_i time units, h_i is visited at least $f_i/f \cdot t_i \cdot f = 1$ times. Consequently, GSWEEP can guarantee the required sweep coverage.

Theorem 3. *GSWEEP algorithm has an approximation ratio at most 3.*

Proof. As shown in the GSWEEP algorithm, for the corresponding TSP problem on the complete graph we build, Christofides algorithm has an approximation ratio 1.5. This implies that route P derived by Christofides algorithm has a length $L' = 3L/2$, if the length of optimal route of the TSP problem is L. Then, the number of sensors required by GSWEEP is $N_{gs} = L'/L_0 = 3L \cdot f/v$. At the same time, the optimal solution of the min-sensor sweep-coverage problem is $N_{opt} \geq L/(v \cdot 1/f) = L \cdot f/v$. Therefore, the approximation ratio of GSWEEP is $N_{gs}/N_{opt} \leq 3$. This finishes the proof. □

4.5 Nonlocality of Sweep Coverage

In full coverage, it has been shown that sensors can locally determine whether a given region is not fully k-covered [17]. If any point on the perimeter of a sensor's sensing disk is covered by less than k sensors, then this sensor can locally conclude that the region is not fully k-covered.

In the case of sweep coverage, however, an individual mobile sensor cannot locally say "yes" or "no" to the question of whether a given set of POIs is globally sweep covered. We can explain this as follows.

In many applications, the number of POIs is large and the distance between them is long. One sensor is insufficient for many application requirements, and two or more mobile sensors are necessary. In such a mobile sensor network, if no centralized deterministic scheme like GSWEEP is provided, a sensor s_i cannot know the whole moving path of all other sensors. Then, s_i cannot determine whether the POIs not monitored by itself during each sweep period have been visited by any other sensor during corresponding time period. Therefore, a sensor cannot locally determine whether all POIs are t-sweep covered. Consequently, t-sweep coverage cannot be guaranteed by any deterministic scheme F without global information. In other words, none of the distributed local algorithms can guarantee the required t-sweep coverage.

Unfortunately, centralized global algorithms are not scalable for large-scale networks. In practice, the POIs to be sweep covered may change over time. Furthermore, the moving speed of mobile sensors might also vary and the mobile sensor may even fail during their trips. Therefore, both CSWEEP and GSWEEP are not scalable and adaptive to practical cases. To address these problems, we propose a distributed sweep algorithm, DSWEEP, using only local information to provide adaptive and reliable coverage with best effort of mobile sensors.

5 The DSweep Algorithm

As mentioned above, a distributed algorithm is necessary for manipulating large-scale networks. Without centralized scheduled moving route, each sensor only locally decide its moving path on runtime based on the knowledge exchanged with other sensors. Two questions need be answered before launching the algorithm. How does one sensor exchange the information with other sensors in the dynamic network? And, how does one sensor decide which POI to move toward based on the obtained information? In this section, we describe the principle of DSWEEP in detail and answer above two questions.

5.1 Assumptions

DSWEEP makes following assumptions. All sensors know their instant locations on the 2D plane, with the help of external location services such as GPS. Each POI has a globally unique position and ID. The positions and sweep period of all POIs are preknowledge for each sensor. All sensors keep moving with constant speed. The communication range of each sensor is assumed to be larger enough so that the sensors can exchange their coverage information with neighboring nodes. Also, all sensors are assumed to be roughly synchronized [30].

5.2 Epidemic Exchange

When a sensor arrives at one POI, it does the job of sampling and inspection. Then, it stores the coverage information, including the swept POI IDs and swept time. All the POI ID and swept time pairs form a sweep table which is locally stored at the sensor. For the same POI, only the latest swept time is saved. In order to precisely determine the next POI, each sensor needs the global coverage information of all sensors. However, in a dynamic and mostly disconnected network, there are few connected paths for sensors to flood their sweep table.

To address this problem, we use a variant of epidemic routing [32] to exchange sweep tables among sensor nodes. Epidemic routing adopts a "store-carry-forward" paradigm: a node receiving a packet buffers and carries that packet as it moves, passing the packet on to new nodes that it encounters. Newly infected nodes, in turn, behave similarly. The random pairwise exchanges of messages among mobile hosts ensure eventual message delivery.

In our case, every time a mobile sensor encounters another one, they immediately exchange their sweep tables. And afterwards both of them locally combine the

POI_ID	Swept_time	Sensor_ID
17	11,30	s_i
20	12,00	4
22	11,20	4
31	11,00	5
35	11,34	3
40	11,25	9

(a)

POI_ID	Swept_time	Sensor_ID
17	11,40	1
20	11,40	s_j
31	11,10	8
35	11,50	8
40	11,25	s_j
52	11,25	1

(b)

POI_ID	Swept_time	Sensor_ID
17	**11,40**	s_i
20	12,00	4
22	11,20	4
31	**11,10**	s_i
35	**11,50**	s_i
40	11,25	9
52	**11,25**	s_i

(c)

POI_ID	Swept_time	Sensor_ID
17	11,40	1
20	**12,00**	s_j
22	**11,20**	s_j
31	11,10	8
35	11,50	8
40	11,25	s_j
52	11,25	1

(d)

Fig. 4. An example of the filtered table exchange. (a) The original sweep table of sensor s_j before exchange. Only one entry comes from sensor s_i. (b) The original sweep table of s_i. Two entries come from s_j while s_i sends the shaded entries to s_j. (c) Sweep table of s_j after combining entries received from s_i while s_j sends the shaded entries to s_i. (d) Sweep table of s_i after combining entries received from s_j. The bold entries are new ones from s_j.

two sweep tables into a new table. The combining rules are as follows: If a new swept POI ID appears, the sensor just inserts it as a new entry in its own sweep table. If the same swept POI ID appears twice, the sensor only keeps the one with the latest swept time. Next time any two other sensors encounter, the same process is repeated, whereas exchanged tables are new ones. Therefore, the coverage information of a sensor can propagate quickly to the whole network. The ACK is used to guarantee reliable exchange process.

In fact, in above process, sensors do not need exchange the whole table with their neighbors. A sensor only needs those latest entries. For a sensor, however, it does not know what the neighbor has and what it needs before exchange. Therefore, we add a flag for each entry in the sweep table, including the POI ID, swept time, sensor ID. The column sensor ID means the ID of the sensor where the latest swept time information of the POI comes from. Further, a sensor needs not send the neighboring node those entries from the neighbor itself. For example in Fig. 4, when s_i and s_j encounter each other, during setting up the connection, they exchange the number of entries in which the sensor ID is equal to the other node. Therefore, s_i knows the number of entries in which the sensor ID is equal to s_i in the table of s_j, denoted by n_1, and so does s_j, denoted by n_2. If n_2 is larger than n_1, sensor s_i first sends s_j the entries in which the sensor ID is not s_j. Fig. 4a depicts the original sweep table of sensor s_j before the information exchange. Only one entry was from sensor s_i. Fig. 4b depicts the original sweep table of s_i. Two entries were from s_j. s_i first sends the shaded entries to s_j. After receiving the information from s_i, in Fig. 4c, sensor s_j combines the entries into its local sweep table according to aforementioned combination rules. It then sends s_i the shaded entries in which the sensor ID is not s_i. Fig. 4d shows the updated sweep table of s_i after the information exchange. Obviously, the latter sensor to send the table entries can save more unnecessary transmissions. The filtered table exchange can filter most redundant entries between two neighbors. Therefore, the transmission overhead is largely reduced. For example in Fig. 4, the number of exchanged entries is reduced from 12 to 7.

At the same time, the sensor periodically updates coverage information. Deleting outdated and useless information saves storage space and especially saves the energy consumption of data transmission. For each swept POI, if the time interval between its swept time and current time is no less than its sweep period, then it is outdated and deleted by the sensor.

5.3 Next-POI Decision

After a sensor finishes sweeping one POI, it needs decide the next POI to serve. The natural idea is that the nearest and most urgent POI should be first served. Considering the POIs in a planar graph, we can get the maximum distance between neighboring POIs, which is denoted as d_{max} and also referred to as one-hop distance. The moving speed is denoted as v. Therefore, the moving time of one-hop distance is $d_{max/v}$, which is also referred to as one-hop time. Similarly, $2 \cdot d_{max}$ is called as two-hop distance and $2d_{max/v}$ is two-hop time.

When sensor s_j finishes sweeping POI h_i, it first checks the set of POIs less than one-hop distance from h_i, denoted as H_i. Then for each POI in H_i, sensor s_j checks its sweep time locally in the sweep table. If the ID of one POI is not in the sweep table, there are three cases. One is POI h_j has never been covered. The second is that POI h_i was swept a long time ago, so the entry about it has been deleted with information updating. The third is sensor s_j has not obtained any coverage information of POI h_i. Both of the first two cases imply that POI h_j needs to be covered immediately. Therefore, the sensor marks those POIs as candidates. For all candidates, the mobile sensor chooses the closest one as the next POI. Otherwise, for each POI, its forthcoming sweep deadline is its last swept time added by its own sweep period. If the forthcoming deadline of any POI is within next one-hop time period, this POI is marked as an urgent POI. If multiple urgent POIs exist, the one with earliest sweep deadline is selected as next POI. If no POIs exist during the next one-hop time period, the sensor tries to find an urgent one during the next two-hop time period. Similarly, the sensor finds the POIs less than two-hop distance, and check whether their forthcoming sweep deadlines are within next two-hop time period. The same steps are repeated until its next POI is decided. The next-POI decision process is illustrated in Fig. 5. In Fig. 5a, the sensor finds one candidate POI within the one-hop distance, and then it selects such a POI as next-POI. In Fig. 5b, the sensor finds no urgent POIs within one-hop

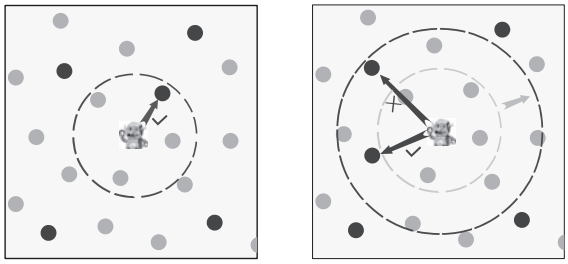


Fig. 5. An example of DSWEEP next-POI decision. (a) The sensor finds the next-POI with one-hop distance. Then, the decision is done. (b) The sensor finds two candidates with two-hop distance, and selects the more urgent one.

distance, so it continues to check the stations within two-hop distance. Finally it finds two urgent candidate POIs in the forthcoming two-hop time. Then, it selects the one with earliest deadline to move toward.

Algorithm 1. Event handlers

Handler $OnExchange(node\ j)$
1: **select** entries S_i from *table* i **where** sender is not j;
2: **send** entries S_i to j;
3: **receive** entries S_j from j;
4: **for** each entry $S_j(k)$
5: **if** POI exists in an entry E_i of *table* i
6: **if** swept time $S_j(k) \cdot t > E_i \cdot t$
7: **substitute** E_i with $S_j(k)$;
8: **else**
9: **drop** $S_j(k)$;
10: **else**
11: **append** $S_j(k)$ to *table i*;

Handler $OnUpdate(timer\ t)$
1: **for** each entry E_i of *table i*
2: **if** $E_i \cdot t +$ sweep period $<= t$
3: **drop** E_i;

Handler *OnDecide*(POI h, current time t)
1: **for** each entry E_i not in *table* i
2: **if** dist$(E_i \cdot \text{POI}, h) <= d_{max}$
3: **put** $E_i > H_i$;
4: **if** H_i is not empty
5: **choose** E_i with the least dist;
6: **else**
7: **for** each entry E_i of *table i*
8: **if** dist$(E_i \cdot \text{POI}, h) <= d_{max}$
9: & $E_i \cdot t +$ sweep period $< t + d_{max}/v$
10: **put** $E_i - > H_i$;
11: **if** H_i is not empty
12: **choose** E_i with the least dist;
13: **else**
14: **for** each entry E_i of *table i*
15: **if** dist$(E_i \cdot \text{POI}, h) <= 2 \cdot d_{max}$
16: & $E_i \cdot t +$ sweep period $< t + 2 \cdot d_{max}/v$
17: **put** $E_i - > H_i$;
18: **if** H_i is not empty
19: **choose** E_i with the least dist;

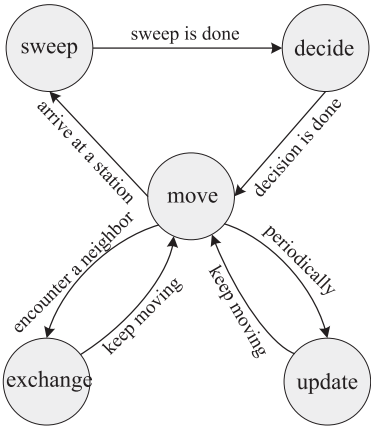


Fig. 6. State transition diagram of a mobile sensor.

5.4 State Transition of DSWEEP

To better describe the execution of DSWEEP, we analyze the state transition of DSWEEP in each sensor. As shown in the above, every sensor has five types of actions in DSWEEP.

- *Exchange.* The action of coverage information propagation described in Section 5.2.
- *Update.* The action of periodically checking the sweep table to delete outdated information described in Section 5.2.
- *Sweep.* The action of patrol inspection at a POI.
- *Decide.* The action of determining the next POI to move toward, which is detailed in Section 5.3.
- *Move.* The action of moving from one POI to another.

After deployment, all the sensors keep moving in the given region and perform the DSWEEP algorithm. The state transition of each sensor is shown in Fig. 6. In most of the time, the sensor keeps moving toward the targeted POI. When it arrives at the POI, it transits to the sweep state. The data sampling and inspection are performed, and then it starts to determine the next POI. After the next POI is determined, it moves toward it immediately.

There are three types of events that trigger the handlers at each mobile sensor. 1) When the current sensor S_i encounters another node S_j and receives the beacon of node S_j, it triggers an *OnExchange handler*, which executes sweep table exchange and POI entry updates. Node S_i first selects the entries from its local swept table which are not from S_j, and sends them to S_j (lines 1-2). Node S_i receives entries from S_j as well (line 3). After that, node S_i examines the received entries and updates the local swept table accordingly (lines 5-11). 2) The current sensor periodically fires a timer, and it triggers an *OnUpdate handler*, which updates the local swept table and drops outdated entries. Each entry from the swept table is picked up and compared with current time. If the time interval between its swept time and current time is no less than its sweep period, then it is outdated and dropped (lines 1-3). 3) When the current sensor finishes sweeping at a particular POI, it triggers an *OnDecide handler*, which decides the next POI the sensor should head for. The current sensor first selects from those POIs within one-hop distance but not included in the swept table (lines 1-5), then those POIs included in the swept table but with urgent sweep deadlines (lines 6-12), and finally those POIs within two-hop distance (lines 13-19).

6 Performance Evaluation

We conduct simulation experiments on the 3D robot simulator simbad [3] to test the performance of our algorithms. We present the simulation results in this section.

6.1 Simulation Setup

For the simulations, we implement a sweep coverage instance on simbad [3]. Hundred POIs are randomly deployed on a 10 by 10 meter square. The constant communication range of sensors is set to be 2 meters. The default moving velocity of mobile sensors is $0.3\ \mathrm{m/s}$. Since the proposed sweep coverage is a purely new coverage scenario, existing distributed algorithms for sensor coverage could not directly apply to this scenario. Therefore, we propose a straightforward randomized scheme for comparison with our DSWEEP algorithm described in Section 5. In the randomized scheme, each mobile sensor knows the positions of all POIs in advance. After the sensor arrives at a POI, it individually chooses a random neighboring POI as the next destination. For simplicity, we name this randomized scheme as RAND in the following.

6.2 Coverage Efficiency

We compare the coverage efficiency of DSWEEP and RAND under two different requirements of sweep coverage. One is all POIs require the same sweep period. The other is different POIs have different periods.

6.2.1 POIs with the Same Sweep Period Requirement

We set the same sweep period for all POIs in this section. The actual sweep period for each individual POI is the metric reflecting the coverage efficiency. Therefore, we first evaluate the cumulative distributed function (CDF) of the average sweep period for individual POIs. We also test the average sweep period of all POIs and the standard deviations. We set the number of sensors $n = 10$ and the moving speed of mobile sensors to be $v = 0.3\ \mathrm{m/s}$. Then, for different required sweep periods $t = 80, 120$, and $160\ \mathrm{s}$, we do the following experiments, respectively. We run the DSWEEP and RAND both for $100{,}000\ \mathrm{s}$ and compute the actual sweep period for each POI.

Fig. 7 shows how the sweep periods of the POIs vary with the required sweep period. Fig. 7a shows the CDF of different average periods of individual POIs when the required sweep period $t = 80\ \mathrm{s}$. It is obvious that DSWEEP significantly outperforms RAND. First, for the fraction of POIs with average period less than $80\ \mathrm{s}$, the required period, the result of DSWEEP is 78 percent much more than the 51 percent of RAND. This means, in DSWEEP more POIs meet their sweep period requirement. Furthermore, the CDF curve of DSWEEP reaches 100 percent more quickly than RAND which guarantees that for those POIs, which cannot meet their required sweep period, will not be delayed for too long. Fig. 7b presents the situation when the required sweep period $t = 120\ \mathrm{s}$. Similarly with the previous situation, first we can find that the sweep periods of POIs in DSWEEP concentrate around the required sweep period, $t = 120\ \mathrm{s}$, while those in RAND distribute along the entire span. Thus more POIs in DSWEEP fulfill the requirements and for those exceeding the required period they will not be delayed for too long as in RAND. Fig. 7c lifts the required sweep period to be $160\ \mathrm{s}$ and shows similar results. The main reason for above results is that the

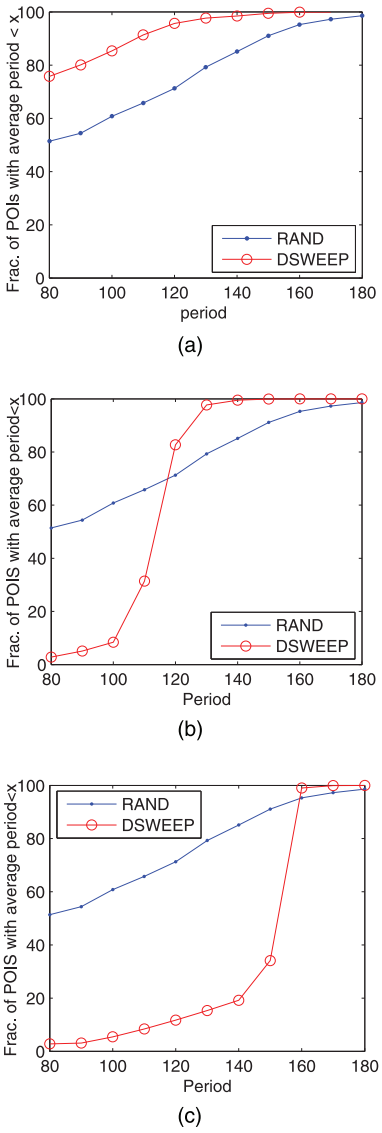
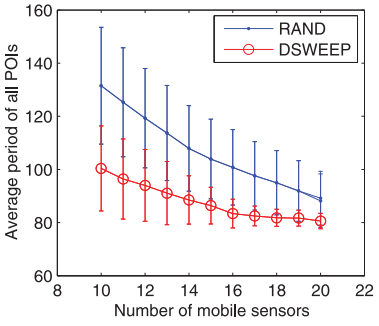


Fig. 7. The cumulative distribution function of the average monitoring period of POIs ($n = 10$ and $v = 0.3\ \mathrm{m/s}$). (a) The required sweep period $t = 80\ \mathrm{s}$. (b) The required sweep period $t = 120\ \mathrm{s}$. (c) The required sweep period $t = 160\ \mathrm{s}$.

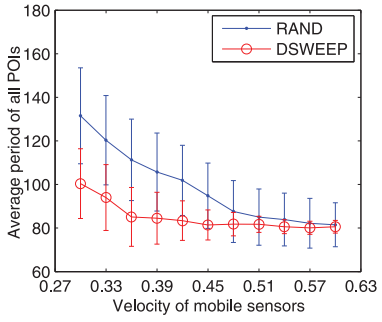
mobile sensor does not coordinate in the RAND scheme thus leading to the fact that some POIs might be visited frequently while other POIs might be visited rarely during a long time. In DSWEEP algorithm, however, if one POI *hi* is monitored by a sensor recently, the sensor will try to send out the information through epidemic exchange. Thereafter, other sensors obtaining this information will not sweep cover it until the next deadline of POI h_i comes.

We further measure the average period of all POIs and the standard deviation. We compute the average period of all POIs to see the global effectiveness and calculate the standard deviation to see the fluctuation on individual POIs. We do three groups of experiments to evaluate the performance of RAND and DSWEEP in Fig. 8.

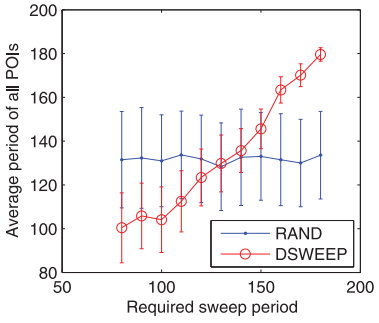
Fig. 8a varies the number of mobile sensors and plots the global average sweep period of all POIs. The moving speed $v = 0.3\ \mathrm{m/s}$ and the required sweep period $t = 80\ \mathrm{s}$. As expected, we see that both the global average period of DSWEEP and RAND decreases with the increase of the



(a)



(b)



(c)

Fig. 8. The global average period of all POIs and standard variation by DSWEEP and RAND scheme. (a) Average period versus the number of mobile sensors ($v = 0.3$ m/s and $t = 80$ s). (b) Average period versus the velocity of mobile sensors ($n = 10$ and $t = 80$ s). (c) Average period versus the required sweep period ($n = 10$ and $v = 0.3$ m/s).

number of mobile sensors. The curve of DSWEEP is much lower than that of RAND and decreases quickly to 80 s, which means DSWEEP can guarantee most of the POIs meet their sweep period with much fewer sensors. The standard deviation of DSWEEP is always much smaller than that of RAND. A small standard deviation is very important to guarantee that the average sweep periods of most POIs are close to the global average difference and thus can fulfill the requirements. Fig. 8b varies the sensor velocity and plots the global average sweep period of all POIs. The number of mobile sensors $n = 10$ and the sweep period $t = 80$ s. This result is similar with that in Fig. 8a. Both the global average period of DSWEEP and RAND decreases with the increase of the velocity of mobile sensors. And as expected, DSWEEP outperforms RAND in terms of either small average sweep periods or small deviations. Fig. 8c varies the required sweep period. The number of mobile sensors $n = 10$ and the moving velocity $v = 0.3$ m/s. As shown in the figure, apparently, the efficiency differs between RAND and DSWEEP. The average sweep period of RAND changes a little with the actual requirement while the average sweep period of DSWEEP is very sensitive to meet the varied requirement. Meanwhile, the standard deviations drop quickly which guarantees that the individual performance of most of the POIs are very close to the global capacity. Therefore, most of the POIs fulfill the required sweep period when the global capacity is adequate.

Through the above extensive simulations, compared with the randomized algorithm, DSWEEP provides required sweep coverage with fewer sensors under lower moving velocity.

6.2.2 POIs with Different Sweep Periods

When the POIs have different importance, their required sweep periods can be different. In this group of experiments, we divide the POIs into three types, the first type with sweep period $t = 80$ s, the second with $t = 120$ s and the third with $t = 160$ s. Each type has equal number of POIs. Then, varied number of sensors and velocities are tested to evaluate their impact on the individual average period of POIs. We call the POIs which fulfill the required sweep period as reliable POIs. Fig. 9 shows the fraction of reliable POIs for all three types, respectively.

Figs. 9a and 9b compare DSWEEP and RAND with different number of mobile sensors. The moving velocity of mobile sensors is set to be $v = 0.3$ m/s. Apparently DSWEEP outperforms RAND with a much larger number of reliable POIs. Moreover, in DSWEEP all three types of POIs have similar fraction of reliable POIs which shows the DSWEEP is adaptive to the hybrid sweep period requirements. In RAND, however, the three different types of POIs differ much with each other. The POIs with loose requirement ($t = 160$ s) has a large fraction of reliable POIs but those with strict requirements ($t = 80$ s) has only a small faction of reliable POIs. Similar results are shown in Figs. 9c and 9d, where we vary the velocities of sensors. Therefore, according to above results, DSWEEP appears to be more adaptive and versatile to the hybrid sweep coverage requirements.

6.3 The Number of Required Sensors

We investigate the effectiveness on the min-sensor sweep-coverage problem in this section. The goal of the min-sensor sweep-coverage problem is to provide the required sweep coverage with the least number of mobile sensors. As mentioned above, no distributed local algorithms guarantee that every POI meets the sweep period requirement, neither does DSWEEP. Thus, we test the actual average sweep period and compare it with the sweep period requirement. If with a relative error less than 10 percent, we consider the mobile sensors are eligible on providing the required sweep coverage. Fig. 10a shows the required number of mobile sensors by RAND, DSWEEP, and CSWEEP under the identical sweep period requirement for all POIs $t = 80$ s. Fig. 10b shows the required number of mobile sensors under three different sweep period requirements for the POIs, i.e., $t = 80, 120$, and 160 s, by RAND, DSWEEP, and GSWEEP. As the velocity of mobile sensors increases, all algorithms need fewer sensors. The CSWEEP and GSWEEP as global centralized algorithms set the lower bounds for DSWEEP, whereas, DSWEEP always outperforms RAND.

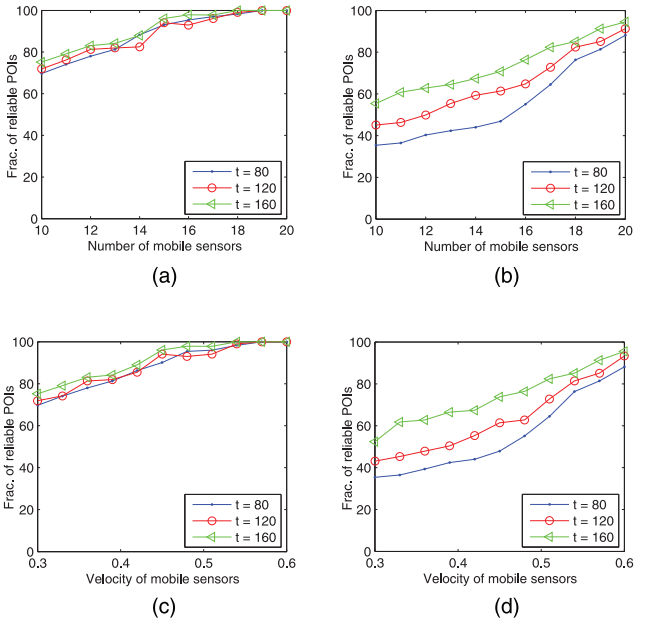


Fig. 9. The fraction of reliable POIs by DSWEEP and RAND scheme. (a) Fraction of reliable POIs versus the number of mobile sensors by DSWEEP ($v = 0.3$ m/s). (b) Fraction of reliable POIs versus the number of mobile sensors by RAND ($v = 0.3$ m/s). (c) Fraction of reliable POIs versus the velocity of mobile sensors by DSWEEP ($n = 10$). (d) Fraction of reliable POIs versus the velocity of mobile sensors by RAND ($n = 10$).

All the above experiments show that the proposed distributed algorithm DSWEEP outperforms the randomized scheme in both effectiveness and efficiency, whereas the proposed centralized algorithms outperforms DSWEEP in the number of required sensors.

7 Conclusion

Patrol inspection with mobile sensors is an efficient scheme for many environments surveillance applications with specified delay bounds. We define the concept of sweep coverage to model the requirements of periodically monitoring a set of POIs in such applications. We discuss the problem of determining the minimum number of required sensors for given sweep coverage requirements. We prove that this min-sensor sweep-coverage problem is NP-hard and it cannot be approximated within a factor of 2. Accordingly we propose a general centralized algorithm, GSWEEP, with constant approximation ratio 3 for this problem. We further design a distributed sweep algorithm, DSWEEP, which cooperates sensors to provide efficient sweep coverage for given POIs and their sweep period requirements with the best effort. The simulation results show that DSWEEP outperforms a straightforward randomized scheme in both effectiveness and efficiency.

Sweep coverage is a purely new concept for sensor network monitoring. There are still many interesting problems not discussed in this paper. One significant extension of this problem is that for a given area rather than a set of discrete POIs, how to determine the metric of sweep coverage and study the applicability? How to work toward a bounded distributed algorithm and reduce the communication cost in a practical protocol for sweep coverage is also challenging. In our future work, we plan to study these problems and obtain more useful results.

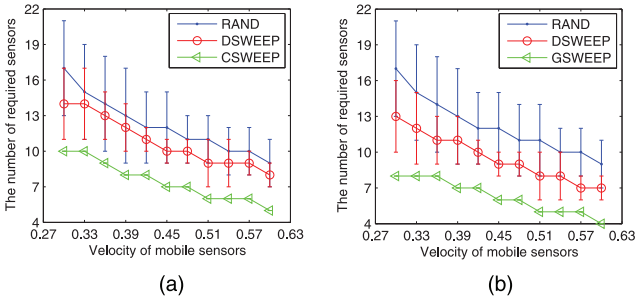


Fig. 10. The number of required sensors versus various moving velocities by different algorithms. (a) The number of required sensors versus the velocity with identical sweep period requirement $t = 80$ s. (b) The number of required sensors versus the velocity with three types of sweep periods, $t = 80, 120$, and 160 s.

Acknowledgments

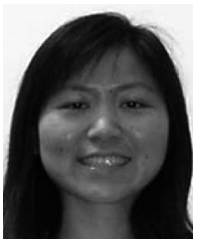
This work was partially supported by SUG COE_SUG/RSS_20Aug2010_13/14 from the Nanyang Technological University of Singapore, US National Science Foundation grants CNS-0832120 and CNS-1035894, the National Natural Science Foundation of China under Grant Nos. 60828003 and 60903224, a program for the Zhejiang Provincial Key Innovative Research Team, a program for the Zhejiang Provincial Overseas High-Level Talents (One-Hundred Talents Program), and the Tsinghua National Laboratory for Information Science and Technology (TNList).

References

[1] "GreenOrbs," http://www.greenorbs.org, 2011.
[2] "Robocar," http://www.esi.ust.hk/projects.html, 2011.
[3] "Simbad," http://simbad.sourceforge.net, 2011.
[4] S. Arora, "Polynomial-Time Approximation Schemes for Euclidean TSP and Other Geometric Problems," *Proc. IEEE 37th Ann. Symp. Foundations of Computer Science (FOCS '96)*, 1996.
[5] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar, "Reliable Density Estimates for Achieving Coverage and Connectivity in Thin Strips of Finite Length," *Proc. ACM MobiCom*, 2007.
[6] M.A. Batalin and G.S. Sukhatme, "Multi-Robot Dynamic Coverage of a Planar Bounded Environment," *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems*, 2002.
[7] M.A. Batalin and G.S. Sukhatme, "Multi-Robot Dynamic Coverage of a Planar Bounded Environment," Univ. of Southern California, CRES-03-011, 2003.
[8] M. Bennewitz, W. Burgard, and S. Thrun, "Optimizing Schedules for Prioritized Path Planning of Multi-Robot Systems," *Proc. IEEE Int'l Conf. Robotics and Automation (ICRA '01)*, 2001.
[9] M. Cardei and D.-Z. Du, "Improving Wireless Sensor Network Lifetime through Power Aware Organization," *ACM Wireless Networks*, vol. 11, pp. 333-340, 2005.
[10] M. Cardei and J. Wu, "Energy-Efficient Coverage Problems in Wireless Ad Hoc Sensor Networks," *J. Computer Comm. Sensor Networks*, vol. 29, pp. 413-420, 2005.
[11] A. Chen, S. Kumar, and T.H. Lai, "Designing Localized Algorithms for Barrier Coverage," *Proc. ACM MobiCom*, 2007.
[12] H. Choset, "Coverage for Robotics—A Survey of Recent Results," *Annals of Math. and Artificial Intelligence*, vol. 31, pp. 113-126, 2001.
[13] N. Christofides, "Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem," Technical Report 388, Carnegie Mellon Univ., 1976.
[14] P. Fazli, "On Multi-Robot Area Coverage," *Proc. Ninth Int'l Conf. Autonomous Agents and Multiagent Systems (AAMAS '10)*, 2010.
[15] D.W. Gage, "Command Control for Many-Robot Systems," *Proc. AUVS Technical Symp.*, 1992.
[16] A. Howard, M.J. Mataric, and G.S. Sukhatme, "Mobile Sensor Network Deployment Using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem," *Proc. Int'l Conf. Distributed Autonomous Robotic Systems 5 (DARS '02)*, 2002.

[17] C.F. Huang and Y.C. Tseng, "The Coverage Problem in a Wireless Sensor Network," *Proc. Second ACM Int'l Conf. Wireless Sensor Networks and Applications (WSNA '03)*, 2003.
[18] F.A. Kolushev and A.A. Bogdanov, "Multi-Agent Optimal Path Planning for Mobile Robots in Environment with Obstacles," *Proc. Third Int'l Andrei Ershov Memorial Conf. Perspectives of System Informatics (PSI '99)*, 1999.
[19] C. Kong, A. New, and I. Rekleitis, "Distributed Coverage with Multi-Robot System," *Proc. IEEE Int'l Conf. Robotics and Automation (ICRA '06)*, 2006.
[20] S. Kumar, T.H. Lai, and A. Arora, "Barrier Coverage with Wireless Sensors," *Proc. ACM MobiCom*, 2005.
[21] S. Kumar, T.H. Lai, and J. Balogh, "On K-Coverage in a Mostly Sleeping Sensor Network," *Proc. ACM MobiCom*, 2004.
[22] D. Latimer, S. Srinivasa, V. Shue, S. Sonne, and H. Choset, "Towards Sensor Based Coverage with Robot Teams," *Proc. IEEE Int'l Conf. Robotics and Automation (ICRA '02)*, 2002.
[23] L. Lin and H. Lee, "Distributed Algorithms for Dynamic Coverage in Sensor Networks," *Proc. 26th Ann. ACM Symp. Principles of Distributed Computing (PODC '07)*, 2007.
[24] B. Liu, P. Brass, and O. Dousse, "Mobility Improves Coverages of Sensor Networks," *Proc. ACM MobiHoc*, 2005.
[25] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M.B. Srivastava, "Coverage Problems in Wireless Ad-Hoc Sensor Networks," *Proc. IEEE INFOCOM*, 2001.
[26] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, and G. Dai, "Canopy Closure Estimates with GreenOrbs: Sustainable Sensing in the Forest," *Proc. Seventh ACM Conf. Embedded Networked Sensor Systems (SenSys '09)*, 2009.
[27] M. Ryan, "Graph Decomposition for Efficient Multi-Robot Path Planning," *Proc. 20th Int'l Joint Conf. Artifical Intelligence (IJCAI '07)*, 2007.
[28] M.R.K. Ryan, "Exploiting Subgraph Structure in Multi-Robot Path Planning," *J. Artificial Intelligence Research*, vol. 31, pp. 497-542, 2008.
[29] S. Slijepcevic and M. Potkonjak, "Power Efficient Organization of Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Comm. (ICC '01)*, 2001.
[30] B. Sundararaman, U. Buy, and A.D. Kshemkalyani, "Clock Synchronization in Wireless Sensor Networks: A Survey," *Ad-Hoc Networks*, vol. 3, pp. 281-323, May 2005.
[31] P. Svestka and M.H. Overmars, "Coordinated Path Planning for Multiple Robots," *Robotics and Autonomous Systems*, vol. 23, pp. 125-152, 1998.
[32] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected Ad hoc Networks," Technical Report CS-200006, Duke Univ., 2000.
[33] D. Wang, J. Liu, and Q. Zhang, "Field Coverage Using a Hybrid Network of Static and Mobile Sensors," *Proc. IEEE Int'l Workshop Quality of Service (IWQoS '07)*, 2007.
[34] G. Wang, G. Cao, and T.L. Porta, "Sensor Deployment and Target Localization Based on Virtual Forces," *Proc. IEEE INFOCOM*, 2003.
[35] G. Wang, G. Cao, and T.L. Porta, "Movement-Assisted Sensor Deployment," *Proc. IEEE INFOCOM*, 2004.
[36] W. Wang, V. Srinivasan, and K.C. Chua, "Trade-Offs between Mobility and Density for Coverage in Wireless Sensor Networks," *Proc. ACM MobiCom*, 2007.

Mo Li received the BS degree from the Department of Computer Science and Technology from Tsinghua University, China, in 2004 and the PhD degree from the Department of Computer Science and Engineering from the Hong Kong University of Science and Technology. He is currently an assistant professor in the School of Computer Engineering at Nanyang Technological University, Singapore. He won the ACM Hong Kong Chapter Professor Francis Chin Research Award in 2009 and the Hong Kong ICT Award—Best Innovation and Research Grand Award in 2007. His research interests include wireless sensor networking, pervasive computing, mobile and wireless computing, etc. He is a member of the IEEE and the ACM.



Weifang Cheng received the BS, MS, and PhD degrees in 2002, 2004, and 2008, respectively, all from the School of Computer from the National University of Defense Technology, China. Her research interests include wireless sensor networks and information security.

Kebin Liu received the BS degree from the Department of Computer Science at Tongji University and the MS and PhD degrees from Shanghai Jiaotong University, China. He is a member of the Tsinghua National Lab for Information Science and Technology. He currently works as a postdoctoral research fellow at the Hong Kong University of Science and Technology. His research interests include sensor networks, pervasive computing, and distributed systems. He is a member of the IEEE.

Yuan He received the BE degree from the University of Science and Technology of China in 2003, the ME degree from the Institute of Software, Chinese Academy of Sciences, in 2006, and the PhD degree from the Hong Kong University of Science and Technology. He is a member of the Tsinghua National Lab for Information Science and Technology. He now works as a postdoctoral fellow in the Department of Computer Science and Engineering at the Hong Kong University of Science and Technology. His research interests include sensor networks, peer-to-peer computing, and pervasive computing. He is a member of the IEEE, the IEEE Computer Society, and the ACM.

Xiang-Yang Li received both the BEng degree in computer science and the bachelor's degree in business management from Tsinghua University, PR China, in 1995, and the MS degree in 2000 and the PhD degree in 2001 in computer science from the University of Illinois at Urbana-Champaign. He has been an associate professor since 2006 and was an assistant professor of computer science at the Illinois Institute of Technology from 2000 to 2006. He has served as an editor of the *IEEE Transactions on Parallel and Distributed Systems (TPDS)* since 2010, an editor of *Networks: An International Journal* since 2009, an advisory board member of *Ad Hoc and Sensor Wireless Networks: An International Journal* since 2005, and a steering committee member of the AAIM conference. He was a guest editor of special issues for *ACM Mobile Networks and Applications*, the *IEEE Journal on Selected Areas in Communications*, and several other journals. His research interests span wireless sensor networks, computational geometry, and algorithms. He has been a senior member of the IEEE since 2008.

Xiangke Liao received the BS and MS degrees in computer science from Tsinghua University and the National University of Defense Technology (NUDT), China, in 1985 and 1988, respectively. He is now a professor and dean in the School of Computer, NUDT, China. His research interests include parallel and distributed computing, high performance computer systems, operating systems, and networked embedded systems.