Vector Method based Coverage Hole Recovery in Wireless Sensor Networks

Prasan Kumar Sahoo
Dept. of Information Management
Vanung University
Chungli, 32061, Taiwan
Email: pksahoo@mail.vnu.edu.tw

Jang-Zern Tsai, Hong-Lin Ke
Dept. of Electrical Engineering
National Central University
Chungli, 32001, Taiwan
Email: jztsai@ee.ncu.edu.tw

***Abstract*—In Wireless Sensor Networks (WSN), sensors form the network dynamically without help of any infrastructure. The accidental death of the nodes due to technical failures or death due to power exhaustion may disturb the existing coverage and connectivity of the network. In this paper, distributed coverage hole recovery algorithms for the wireless sensor networks are designed that use the vector methods to decide the magnitude and direction of the mobile nodes. In the post deployment scenario, coverage holes of the network are repaired by moving the nodes in a self organized manner. To minimize the energy consumption of the nodes due to mobility, algorithms are designed in such a way that the mobility is limited within only one-hop of the nodes and highest coverage (*k*-coverage) of a node is not increased after its mobility. Performance evaluation of the proposed algorithms show that cent percent of coverage recovery could be possible by moving the nodes within their communication range. Besides, the average mobility distance of the nodes is very small to recover the coverage holes by our algorithms.**

I. INTRODUCTION

Wireless Sensor Network (WSN) consists of hundreds to thousands of low-cost sensing nodes, which are capable of performing limited sensing and communication tasks. In order to monitor the region of interest, sensor nodes should be deployed appropriately with sufficient number of sensors to ensure a certain degree of redundancy. Recently, several researchers have investigated the techniques of mobile sensors such as MICAbot [1], and Robomote [2] to obtain better solution for many issues. Several experiments have done that provide convenient platforms for investigating related algorithms and applications of the mobile sensor networks. Due to mobility of sensor nodes, mobile sensors can change their position depending on the requirement of the missions. When sensors are deployed in a disaster environment, where human interference is not possible, we need mobile sensors to accomplish the tasks such as coverage and connectivity compensation, location assignment and node replacement.

In a post deployment scenario, it is possible that some nodes deployed over certain region are destroyed due to intrusion, explosion or due to environmental factors like heat, vibration, failure of electronic components or software bugs in the network. In another scenario, power sources of the nodes may lead death of the nodes, thereby affecting the coverage and connectivity of the original network. Hence, it is essential to reconfigure the network in order to maintain the connectivity and coverage and to avoid the network partitions. Furthermore, sensors are normally deployed in remote or hostile environments, such as a battlefield or desert and is impossible to recharge or replace the battery. But, the data sensed by the sensors is generally highly critical, and may be of scientific or strategic importance. Hence, coverage provided by the sensor networks is a very important criterion of their effectiveness and its maintenance is highly essential to form a robust network.

It is essential to maintain the network coverage and connectivity as a small un-monitored area can spoil the whole purpose of the network, if it goes undetected. Besides, density of the nodes throughout the network may not be uniform due to random deployment of the sensors. Though, some researchers have proposed coverage and connectivity maintenance algorithms, to the best of our knowledge, none of the work proposes how to recover the coverage holes with limited mobility of the nodes and without increasing the highest coverage of a mobile node after mobility. In this paper, we propose the distributed coverage hole recovery algorithms and main contributions of our work can be summarized as follows.

- We propose distributed algorithms to recover the holes that use the laws of vectors as the tool to decide the magnitude and direction of the mobile nodes.
- Since mobility of the nodes consumes more energy, mobility of nodes in our protocol is limited within only one-hop.
- Our algorithms can recover coverage holes of irregular shapes without increasing the highest coverage of the mobile nodes.
- In our protocol, decision of mobility requires limited computation and therefore more efficient as compared to other hole recovery protocols.

The rest of the paper is organized as follows. The related work are given in Section II. Problem formulation with few definitions related to our work are given in Section III. Our

---

This work is supported by the National Science Council of Taiwan partly under the grant NSC 98-2221-E-238-010, and partly under the grant NSC 97-2218-E-008-001.

978-1-4244-5489-1/10/ $26.00 © 2010 IEEE

limited mobility hole recovery algorithms are described in Section IV. Performance analysis of our algorithms and future work are presented in Section V. Concluding remarks are made in Section VI.

II. RELATED WORK

A distributed algorithm for the coordinated coverage fidelity (Co-Fi) maintenance [3] in sensor networks is proposed, where mobile nodes are used to repair the coverage loss of the area being monitored by it. The dying node notifies the network of its death, which is not practical for some factors causing death such as program failure and malicious damage. A Dynamic Coverage Maintenance (DCM) scheme is proposed in [4], which exploits the limited mobility of the nodes. Considering transmission range is twice of the coverage range. The paper proposes a set of mobility schemes, which can be executed on individual sensors and the proposed algorithms decide which neighbors to migrate, and to what distance. In [5], a potential field based approach is proposed for the self-deployment of mobile sensor network, in which nodes use their sensed information in making the decision to move. It is a cost effective solution to the coverage problem and experimental studies have not been conducted to test the sensitivity of the changes in transmission and sensing ranges for different network sizes. The concept of Voronoi diagram is used in [6] to discover the existence of coverage holes, in which a node compares its sensing disk with the area of its Voronoi polygon to estimate any local coverage hole.

As mentioned in [7], the communication range (R_c) and sensing range (R_s) of each sensor can be controlled or adjusted by itself and the coverage and connectivity can be guaranteed. Because of the redundant nodes, the WSNs may not work efficiently even if we maintain the coverage and connectivity. To reduce the number of redundant nodes, the authors in [8] propose a redundant elimination algorithm which is developed by voronoi diagram. To make the WSNs work more efficient, there is another point of view to decrease the redundant nodes by controlling the density of WSNs. A probing-based algorithm is proposed in [9]. It chooses the smallest covering set of nodes by the probing range and any pair of working nodes is less than or equal to a probing range. An optimal geographical density control protocol is proposed in [10] that minimizes the overlapping area between each working node without disturbing the connectivity and coverage in the view of three sensors. The authors in [11] converts the density control to be the flow control problem in a grid-based network to decide the maximum mobile distance of a node and provides k-coverage for a monitoring area.

In [12], algorithms are proposed to maintain the coverage and connectivity of the WSNs, where a node has to calculate the the required and available moving distance before deciding the magnitude and direction of the mobility. Based on the available mobility distance of a node, it can move to recover the coverage hole of the network. The authors [13] design distributed algorithms to maximize the sensing coverage and to minimize the moving distance of a mobile sensor. The proposed scheme can work for any irregular shape of the obstacles or coverage hole. Positioning of a given number of mobile stations in a given region of wireless networks to cover the largest possible area is proposed in [14]. Based on this scheme, a configuration of node positions corresponding to a maximal covered area can be identified as long as nodes continuously move in directions that guarantee increasing coverage. A probabilistic approach is proposed in [15] to compute the covered area fraction and a correlated disk model is developed for percolation in WSNs. However, the paper does not mention how to recover the coverage holes.

An analytical method for the coverage problems of WSN is proposed in [16] to improve the network lifetime of the nodes. The authors propose a two-dimensional Gaussian distribution and analyze with different parameters such as standard deviation and dispersion. Though, the paper studies the coverage problems, the work is meant for designing the deployment strategy based on the analytical models of coverage and lifetime and it does not propose any hole recovery method. Three sensor relocation algorithms are proposed in [17] based on mobility degree of sensors, particle swarm optimization based algorithm and energy efficient fuzzy optimization algorithm. In their limited mobility based algorithm, mobility distance of a node is bounded by a threshold. Besides, they propose static topology control or scheduling schemes to further reduce energy consumption of WSN. However, the work is far behind the coverage recovery algorithms, though it analyzes threshold based mobility of nodes.

From the survey of several literature, it is observed that most of the research are related to coverage and connectivity analysis either to maintain the network or to propose the deployment strategy. Though, few of the work design algorithms for coverage hole recovery or hole detection, to the best of our knowledge none of the work propose to recover hole using mobile sensors, where mobility is limited within only one hop. Hence, in this work we propose the distributed vector algebra based hole recovery algorithms that use few mobile sensors without disturbing the existing network coverage.

III. PROBLEM FORMULATION

It is considered that the sensors are distributed randomly and densely with higher degree of neighbors over the monitoring region of a wireless sensor network. Each node is aware of its own location and coordinates of the boundaries of the deployed region through location services [18]. Each node is capable of moving, as mentioned in [1, 2]. It is assumed that each node has a unique ID and is equipped with homogeneous sensing and transmission ability. The communication range (R_c is equal to twice of the sensing (R_s) of a node. At the time of deployment, it is assumed that there are multiple coverage holes present in the network. However, the nodes around the coverage holes are connected and are aware of the location of their nearest coverage holes. Due to random deployment of nodes, it is assumed that probability of intersection of sensing discs of any three sensors at a single point is zero. Besides, no two sensors are located at a single point and each sensor has at least two neighbors in the monitoring region.

A. Definitions

Definition 1: ***CLOSE HOLE*** If a coverage hole is completely enclosed by the sensing discs of the sensors, the hole is called a close hole. As shown in Fig. 1, H_1 and H_4 are close holes.

Definition 2: ***OPEN HOLE*** If a coverage hole is enclosed by the boundary of the monitoring region and sensing discs of the sensors, the hole is called an open hole. As shown in Fig. 1, H_2, H_3 and H_5 are open holes.

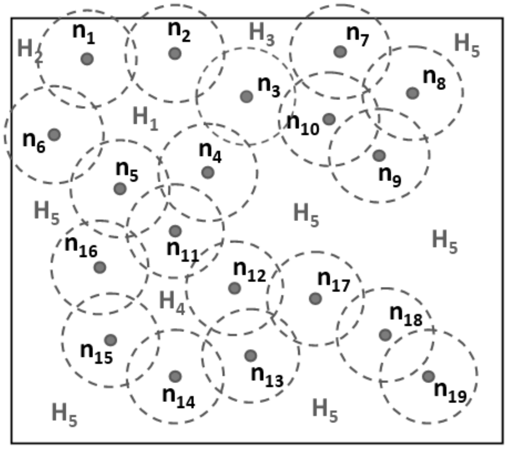


Fig. 1. Example of open and close holes. H_1 and H_4 are close holes, whereas H_2, H_3 and H_5 are open holes.

Definition 3: ***CLOSE-WORKER*** and ***CO-WORKER*** For any two nodes S_i and S_j, if $0 < d_{ij} \leq 2R_s$, S_i and S_j are Close-workers. For any two nodes S_i and S_j, if $2R_s < d_{ij} \leq 3R_s$, S_i and S_j are Co-workers, where d_{ij} is the physical distance between those two nodes. As shown in Fig. 2, n_1 and n_2 are close-workers of S, whereas n_3 and n_4 are co-workers of S.

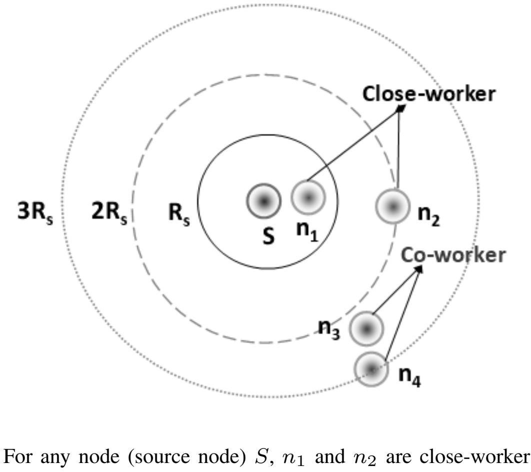


Fig. 2. For any node (source node) S, n_1 and n_2 are close-workers of S. n_3 and n_4 are co-workers of S.

Definition 4: ***CRITICAL POINT*** (P^C) The points of intersection of sensing disc of a source node (S) with sensing disc of each of its connecting neighbors (close-workers) OR with boundary of the monitoring region, are called critical points of source node S.

If S is a source node and S_j and S_k are close-workers of S, such that $d_{jk} \leq 2R_s$, the point of intersection p_{jk} between sensing disc of S_j and S_k OR between the boundary of the monitoring region with any close-workers of S that satisfies the following conditions is called a critical point of S.
1. $p_{jk} \in$ sensing disk of S.
2. Distance between location of source node and that point of intersection $< R_s$.

As shown in Fig. 3, p_1, p_2, p_3, and p_4, are critical points of source node S_2. Similarly, the white balls located within sensing disc of S_1 are also its critical points.

Definition 5: ***BOUNDARY GAP POINT*** (p_{gp}) The point of intersection of sensing disc of a source node (S) with sensing disc of its connecting neighbor (close-worker) S_i OR with boundary of the monitoring region, which lies only on the sensing disc of S_i and S OR on the boundary of the monitoring region is called boundary gap point (p_{gp}). The arc that is formed by joining the consecutive boundary gap points is known as *Boundary gap*. As shown in Fig. 3, S_1 has no boundary gap points, since all of its points of intersection with its close-workers either lies outside of the monitoring region or lies within sensing disc of another close-worker. p_7, and p_8 are boundary gap points of S_2, since p_7 lies on the sensing disc of S_2 and n_6 and p_8 lies on the sensing disc of S_2 and n_{10}. Similarly, nodes n_1 to n_{10} have boundary gap points marked in bigger oval points. The arc connecting to each boundary gap points p_5 to p_{10} is a boundary gap.

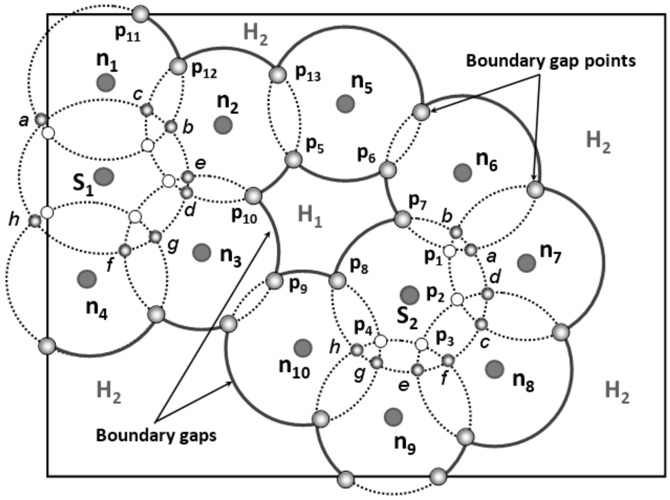


Fig. 3. Example of critical points, boundary gap points and boundary gap.

B. Boundary Gap Construction Algorithm

Based on the definitions of boundary gap points and boundary gap given in subsection A, we design here an algorithm to construct the boundary gap of each sensor. It is to be noted

TABLE I
Boundary Gap Construction Algorithm

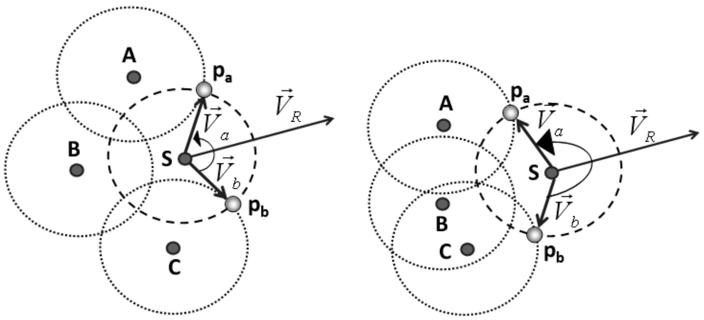
```
Step 1: Select any node (source node) $S$;
Step 2: Create a list of close-workers of $S$;
Step 3: Select a close-worker $S_i$ of $S$ randomly from the list;
Step 4: Sort the list in clockwise starting from $S_i$;
Step 5: Find points of intersection ($p_i$) of $S$ with each of its sorted close-workers;
Step 6: Verify if each $p_i$ is a gap point ($p_g$) or not;
Step 7: Create a list of gap points ($p_g$) of $S$;
Step 8: Sort the list of gap points ($p_g$) in clockwise;
Step 9: Connect each consecutive $p_g$ by constructing an arc (boundary gap) along the border of the nodes having $p_g$;
Step 10: Continue the construction of boundary gap till the starting boundary gap point $p_g$ is revisited
OR it touches the border of the monitoring region;
Step 11: Continue this procedure for all nodes of the network;
```

that the sensors that enclose any close or open hole can have boundary gaps. As per the definition of boundary gap points, the sensors that are not along the boundary of any coverage holes cannot have any boundary gap point and therefore do not need to execute the boundary gap construction algorithm. The detail procedure of our algorithm is given in Table I.

An example of constructing boundary gap is shown in Fig. 3. As shown in the figure, select a source node S_1 randomly. Now, n_1, n_2, n_3 and n_4 are its close-workers of S, which are listed in clockwise. Here, source node S_1 finds its point of intersection with each of its close-workers. Let, a & b, c & d, e & f and g & h are points of intersection of source node S_1 with its close-workers n_1, n_2, n_3 and n_4, respectively. As per definition of gap points, points a and h are not gap points, since they lie outside of the monitoring region. Points b, c, d, e, f and g are not the gap points as they lie more then three sensors including the source node S_1. Hence, there is no boundary gap for the source node S_1. Let us consider another source node S_2, as shown in Fig. 3. Though p_7 & a, b & c, d & e, f & h and g & p_8 are the points of intersection of S_2 with its close-workers n_6, n_7, n_8, n_9 and n_{10}, respectively, only p_7 and p_8 can satisfy the definition of gap points. Listing those points in clockwise, an arc can be drawn along the boundary of S_2 from p_8 through p_7, which represents the boundary gap. Similarly, considering n_1, n_2, n_5 etc as the source nodes, the boundary gap can be drawn along the border of open hole H_2. Taking, n_2, n_5, S_2, n_{10}, and n_3 as source nodes, the boundary gap can be drawn along the border of close hole H_1.

IV. Vector based Hole Recovery (VHR) Protocol

In this section we develop algorithms for the mobility direction and mobility distance of the sensors to recover the coverage holes. We use vector algebra as a tool to determine the direction of coverage holes and accordingly plan the mobility of the nodes to recover them. Our Vector based Hole Recovery (VHR) algorithms are distributed by nature and can decide the direction and magnitude of mobility without increasing highest coverage degree of a node.



(a) Angle of boundary gap $< \pi$ (b) Angle of boundary gap $> \pi$

Fig. 4. Example of angle of boundary gap.

TABLE II
Resultant Vector Construction Algorithm

```
Step 1: Select any node (source node) $S$;
Step 2: Select a pair of consecutive boundary gap points $p_1$ and $p_2$ of $S$;
Let, $(x, y)$, $(x_1, y_1)$, and $(x_2, y_2)$ be coordinates of source node $S$, boundary gap point $p_1$ and $p_2$, respectively;
Step 3: Construct the vector $\vec{V_1} = (x_1 - x)\vec{i} + (y_1 - y)\vec{j}$;
Step 4: Construct the vector $\vec{V_2} = (x_2 - x)\vec{i} + (y_2 - y)\vec{j}$;
Step 5: Continue to construct vectors with rest of the boundary gap points same as Step 3;
Step 6: Let, $\vec{V_1}$, $\vec{V_2}$, ..., $\vec{V_n}$ be the vectors formed by boundary gap points $p_1$, $p_2$, ...,$p_n$, respectively;
Step 7: Construct the resultant vector $\vec{V_R} = \sum_{i=1}^{n}(\vec{V_i})$;
```

A. Construction of Vectors

As discussed in the previous section, each source node can find its boundary gap points, which are ultimately connected with each other to form the boundary gaps. After construction of the boundary gaps, each sensor can know if it encloses a coverage hole or not. It is obvious that a sensor must enclose the hole if it has boundary gap points. Hence, a sensor should estimate its mobility direction and magnitude to recover the coverage hole that it encloses. In our protocol, we suggest that the sensor that encloses the hole should use the laws of vectors as a tool to calculate its mobility direction. To achieve this, each source node connects its own location to the boundary gap points such that head of the vector points toward the boundary gap point. After constructing vectors from the location of the source node to each of the boundary gap points, the result vector is calculated using polygon laws of vector addition. For example as shown in Fig. 4, let A, B, and C are close-workers and p_a and p_b are boundary gap points of S. Now vectors $\vec{V_a}$ and $\vec{V_b}$ can be constructed from the location of source node S with p_a and p_b, respectively. Based on the location of p_a and p_b, the angle of boundary gap (θ) may be $< \pi$ or $> \pi$, as shown in Figs. 4a and 4b, respectively. Taking the vectors $\vec{V_a}$ and $\vec{V_b}$, and using triangle law of vector addition, the resultant vector $\vec{V_R}$ could be constructed as shown in Fig. 4. The detail procedure of finding the resultant vector is given in Table II.

TABLE III
ALGORITHM TO CALCULATE MOBILITY DISTANCE (D)

```
Step 1: Select any node (source node) $S$;
Let $p_S$ be the location of $S$;
Step 2: List the close and co-workers of $S$;
Step 3: List all critical points of $S$;
Step 4: Find the resultant vector $\vec{V_R}$ of $S$;
Step 5: Find equation of the line $L_M$
that passes through location of $S$ and along
the direction of $\vec{V_R}$;
Step 6: Find distance between each critical points
and location of $S$;
Step 7: Find the longest distance among them;
Let it be $d_{max}$ from the critical point $p_c$;
Step 8: Find the distance between each co-workers
of $S$ and itself;
Step 9: Find the shortest distance among them;
Let it be $d_{min}$ from the co-worker $X$;
Step 10: Put a point $p_m$ on $L_M$ such that
$|p_m p_c|$ = $R_s$ units;
Step 11: Put a point $p_n$ on $L_M$ such that
$|p_n p_X|$ = $2R_s$ units, where $p_X$ is location of $X$;
Step 12: Find $|p_m p_S|$ = $d_m$,
where $p_S$ is location of $S$;
Step 13: Find $|p_n p_S|$ = $d_n$;
Step 14: Output the mobility distance $D$ = min ($d_m d_n$);
```

B. Mobility Distance Calculation Algorithm

The goal of this part of the algorithm is to find the maximum mobility distance of a mobile node so that the existing coverage of the node is not lost. Besides, the mobile node should not increase the highest *k*-coverage of the sensors after its movement to a new position. It is assumed that the mobile node S, knows its list of close-workers and co-workers before it decides to move to a new location.

An example of finding mobility distance of a node is shown in Fig. 5. Let S be a mobile node that wants to calculate its mobility distance. Let A, B, and C be the close-workers and D, E, F and G be co-workers of S. a and b are critical points of S and $\vec{V_R}$ is the direction of the resultant vector, which is determined using resultant vector construction algorithm given in Table II. Let L_M be the equation of straight line along the resultant vector $\vec{V_R}$, which passes through position of S. From S, find all possible distances between S and its critical points. Let, a be the critical point from which distance between S and a is longest one. Then from a, put a point c on L_M such that $|ac|$ = R_s units. Similarly, find the co-worker of S that is nearest to it. Let, co-worker D be the nearest one. From the location of D, put a point d on L_M such that $|Dd| = 2R_s$ units. Now, calculate $|Sc|$ and $|Sd|$ and find the minimum value between them. The minimum of ($|Sc|$, $|Sd|$) is considered to be mobility distance of S. As shown in Fig. 5, $|Sc|$ is the mobility distance of the node.

It is to be noted that in our algorithms, mobility of the nodes is limited within only one-hop. Besides, the mobility of a node does not affect the existing coverage and initial *k*-coverage degree of the nodes that enclose the holes is not increased due to mobility of a node to the hole area, which are verified by the following lemmas.

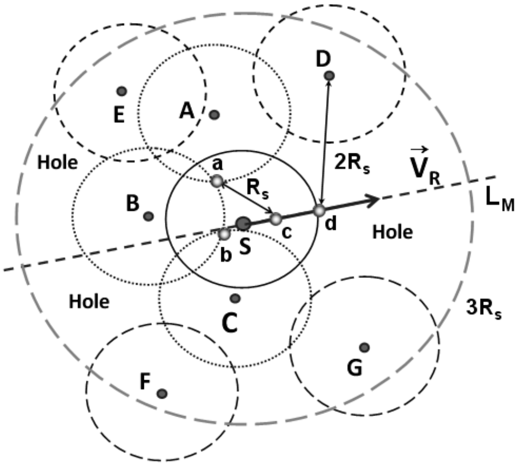


Fig. 5. Example of finding mobility distance of a mobile node S

LEMMA 1: *Mobility distance of a node is limited within its one-hop.*

Proof: Let, c be the distance between new position and critical point (p_c) of S having longest distance.
$\Longrightarrow$ As per our algorithm, this distance (c) must be R_s.
Let, a be the mobility distance, b be the distance between S and p_c and θ be the angle of boundary gap, which must be $< \pi$.
Based on the cosine law, $c^2 = a^2 + b^2 - 2ab \times \cos\theta$.
Since, $\theta < \pi$, c and a must be $\leq R_s$
$\Longrightarrow$ Substituting value of θ, c and a in cosine law, mobility distance a must be $\leq 2R_S$.
$\Longrightarrow$ Mobility of a node is limited within one-hop. ■

LEMMA 2: *The connectivity of a node (source node) with its close-workers is not lost after its mobility.*

Proof: Let, S be a source node, and S_j, S_k are its connecting neighbors (Close-worker).
Let a critical point p_{jk} of any close-worker be within the sensing disk of source node S.
$\Longleftrightarrow$ The maximum distance between p_{jk} and S must be equal to R_s after mobility of S.
$\Longleftrightarrow$ p_{jk} is a critical point of S.
$\Longleftrightarrow$ Distances d_{Sj}, d_{Sk} and d_{jk} must be less than $2R_s$.
$\Longleftrightarrow$ $R_c = 2R_s$ is larger than d_{Sj} or d_{Sk}.
$\Longleftrightarrow$ Connectivity of S is not lost with its close-workers S_j, S_k. ■

LEMMA 3: *The initial k-coverage of the nodes that encloses a hole is not increased after mobility of another node.*

Proof: Let, a node S be a moving one and p_c be a farthest critical point of S.
$\Longleftrightarrow$ Distance (d_m) between p_c and S should be at most R_s units.
$\Longleftrightarrow$ If mobility distance $D \leq d_m$, the existing area is not lost after mobility of S.
As per our algorithm, if S is moving, the distance d_n between S and its closest co-worker should be at most $2R_s$ units
$\Longleftrightarrow$ If $D \leq d_n$, then the k-coverage would not be increased after mobility of S. ■

C. The VHR Algorithm

In this subsection, we summarize the Vector based Hole Recovery (VHR) algorithm. The mobility distance calculation, vector construction method and other necessary procedures are combined to give the complete form of VHR algorithm, as given in Table IV. However, the theoretical basis for the conditions of mobility are verified by the following lemmas.

LEMMA 4: *If a node has at least two boundary gaps, it cannot move.*

Proof: Let us assume that S has at least two boundary gaps.
$\Longleftrightarrow$ Number of boundary gaps is equal to the number of resultant vectors.
$\Longleftrightarrow$ If S moves along the direction of one of the resultant vectors
$\Longleftrightarrow$ The existing coverage must be lost.
$\Longleftrightarrow$ The node cannot move. ■

LEMMA 5: *If angle of boundary gap θ of a node is $< \pi$, it can move.*

Proof: Let angle of boundary gap θ of a source node (S) be $< \pi$
$\Longleftrightarrow$ There must be fully covered area to one side of those boundary gap points due to presence of close-workers of S.
$\Longleftrightarrow$ There must be uncovered area along the direction of $\vec{V_R}$
$\Longleftrightarrow$ $\exists$ Coverage holes along the direction of $\vec{V_R}$
$\Longleftrightarrow$ Node S can move along the direction of $\vec{V_R}$. ■

LEMMA 6: *If angle of boundary gap θ of a node is $> \pi$, mobility direction of that node should be along $-\vec{V_R}$.*

Proof: Let, A and B be two boundary gap points. $\vec{V_A}$ and $\vec{V_B}$ are vectors due to boundary gap points A and B, respectively. Let, the angle of boundary gap between vectors $\vec{V_A}$ and $\vec{V_B}$ be θ_{AB}.
If $\theta_{AB} < \pi$, based on lemma 5
$\Longleftrightarrow$ After mobility of S, the total sensing area should be larger than before
$\Longleftrightarrow$ The coverage overlapping of S with its connecting neighbors should be less than before.
If $\theta_{AB} > \pi$
$\Longleftrightarrow$ The coverage overlapping of S with its connecting neighbors is not less than before
$\Longleftrightarrow$ Node S needs to correct its mobility direction to $-\vec{V_R}$. ■

LEMMA 7: *If a node has single boundary gap with angle of boundary gap $\theta < \pi$, direction of its mobility, i.e. direction of $\vec{V_R}$ cannot be toward location of its close-workers.*

Proof: Let, $\vec{V_A}$ and $\vec{V_B}$ be the vectors of S due to its boundary gap points A and B, respectively and $\theta_{AB} < \pi$.
If direction of $\vec{V_R}$ is opposite to $\vec{V_A}$ or $\vec{V_B}$.
$\Longleftrightarrow$ θ_{AR}, θ_{BR} or $\theta_{RR} = \pi$
$\Longleftrightarrow$ Angle between $\vec{V_A}$ and $\vec{V_B}$, i.e. $\theta_{AB} = \pi$
$\Longleftrightarrow$ Direction of $\vec{V_R}$ cannot be opposite to direction of $\vec{V_A}$ or $\vec{V_B}$.
$\Longleftrightarrow$ Mobility direction of S cannot be toward location of its close-workers. ■

The detail procedure of our VHR algorithm is given in Table IV followed by an explanation with examples as shown in Fig. 5. From the different steps of our proposed algorithms, it is to be noted that each node determines its critical points and boundary gap points from its one-hop neighbor's information and without help of the sink. Hence, our protocol is distributed by nature.

TABLE IV
VHR ALGORITHM

```
Step 1: Select a node randomly as source node $S$;
Step 2: Find close-workers and co-workers of $S$;
Step 3: Find the boundary gap of $S$ given in Table I;
Step 4: If $S$ has at least two boundary gaps, it cannot move;
Step 5: Go to Step I to select another node as source node;
Step 6: If the angle of boundary gap of $S$
i.e. $\theta > \pi$, it cannot move;
Step 7: Go to Step I to select another node as source node;
Step 8: Find the resultant vector $\vec{V_R}$ of $S$ given in Table II;
Direction of $\vec{V_R}$ is the direction of mobility of $S$;
Step 9: Find the mobility distance $D$ given in Table III;
Step 10: Continue the procedure until all nodes are visited;
```

V. PERFORMANCE EVALUATION

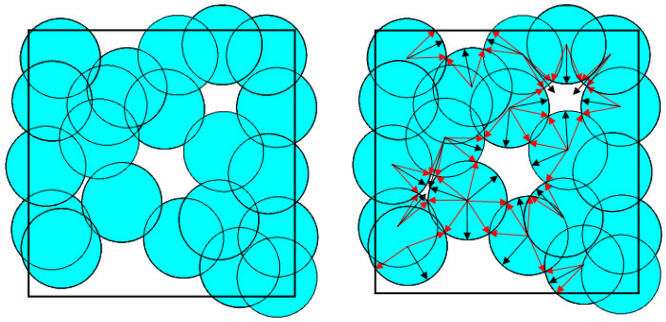
The performance evaluation of our VHR algorithms are analyzed in this section. The algorithms are evaluated in terms of average mobility distance, percentage of hole recovered area and number of recovered holes as described below.

A. Simulation Setup

Our algorithms are simulated using Ns-2.33 for different number of nodes that are deployed randomly over $500m \times 500m$. The number of deployed nodes varies from 200 to 1200. In our simulation the TwoRayGround propagation model and omni-directional antenna are considered and each simulation is run about 30 rounds to get the average of the data. The IEEE 802.15.4 MAC is considered for the channel access mechanism. For each sensor node, a fixed amount of 100J reserved energy is assumed. The sensing range varies from 15m to 30m with communication range is twice of the sensing range, i.e. $R_c = 2R_s$ and a homogenous network environment is considered. The mobility cost of each sensor is taken 8.267J/m. Each sensor, initially broadcasts message limited with in $3R_s$ to get the neighbor information and classifies those neighbors into close and co-workers, which is compatible with our algorithm.

B. Simulation Result

In order to see the effect of vector and resultant vectors, we have considered the simulation environment with and without using the vectors, as shown in Fig. 6. Initially, we deployed the sensors randomly over the monitoring region to create open and close holes, as shown in Fig. 6 (a). Then we run our boundary gap construction and vector construction algorithms, given in Table I and II, respectively to verify if vectors are drawn correctly or not. As shown in Fig 6. (b), vectors and resultant vectors are drawn correctly based on each boundary gap points. In Fig. 6 (b), the red lines represent each individual vectors and the resultant vector is shown in black lines. From



(a) Simulated environment with holes and without using vector methods.

(b) Simulated environment after using vectors. methods.

Fig. 6. Simulation of network environment using vectors.

the figure, it is observed that direction of each resultant vector, whose angle of boundary gap (θ) is $< \pi$, points toward the coverage holes instead of toward the close-workers of a node, which certainly justify the correctness of our theoretical analysis given in different lemmas.

As shown in Fig. 7, the average percentage of recovery of hole area is simulated for different number of nodes to analyze the recovery area before and after the mobility of nodes. Initially, simulation is done to recover the holes without moving the nodes, but by increasing the sensing range. However, it is observed that higher percentage of coverage holes is reduced due to mobility of the nodes. The percentage of hole area is improved by increasing the number of nodes. The average percentage of hole recovered area is simulated for different average number of mobile nodes as shown in Fig. 8. For fixed number of deployed nodes and fixed number of holes, it is first simulated to study the required number of mobile nodes per hole to recover the holes. It is observed that average percentage of hole recovery area is improved if number of deployed nodes is increased. For fixed number of nodes, the average percentage of coverage area is improved if average number of mobile nodes per hole is increased.

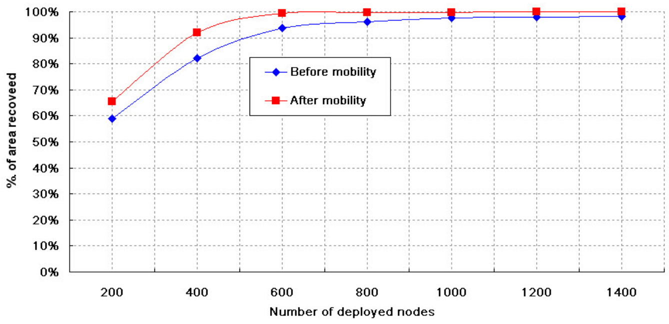


Fig. 7. Average percentage of area recovered before and after mobility of nodes, when sensing range of each node is fixed at $R_s = 20m$.

Mobility is an important parameter in coverage hole recovery protocols as longer mobility distance consumes more energy and therefore affect the network lifetime. It is to be noted that mobility of nodes in our protocol is limited within one hop only. In order to analyze the average mobility distance to recover the number of holes, we simulated our algorithms for different number of nodes. As shown in Fig. 9, it is observed that for less number of deployed nodes, the average mobility distance is increased to recover the same number of coverage holes. However, the average mobility distance decreases drastically if higher number of nodes are deployed. Besides, average mobility distance also increases if we want to recover more number of nodes, which is obvious. As shown in Fig. 10, it is analyzed that for fixed number of deployed nodes, 100 % of hole recovery could be possible if average mobility distance is increased. However, the same 100 % of hole recovery could be achieved and average mobility distance can be reduced, if number of deployed nodes are increased. However, average mobility distance is always less for fixed number of deployed nodes, if we want to recover smaller percentage of hole recovery. From this simulation, it is inferred that it is simply a tradeoff among the average mobility distance, percentage of hole recovery and number of deployed nodes.

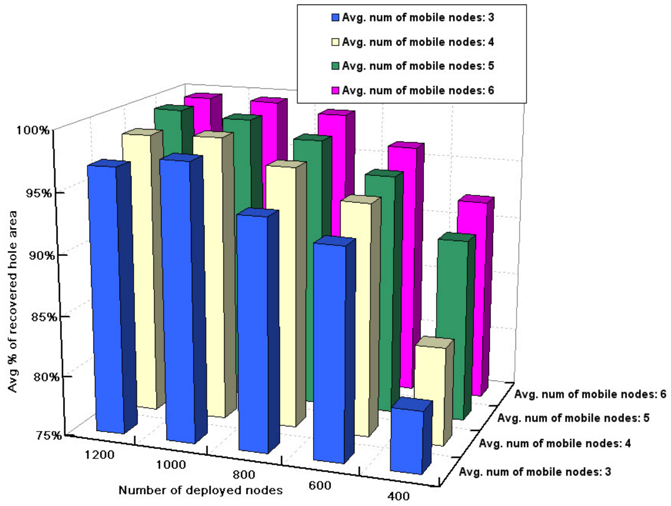


Fig. 8. Average percentage of area recovered due to different average number of mobile nodes per hole against different number of deployed nodes, when sensing range of each node is fixed at $R_s = 20m$.

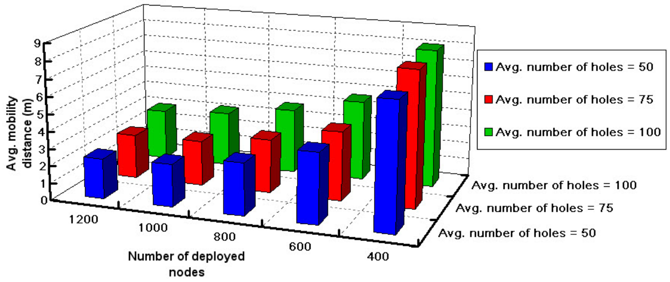


Fig. 9. Average mobility distance for recovering different number of coverage holes against different number of deployed nodes. The sensing range of each node is fixed at $R_s = 20m$.

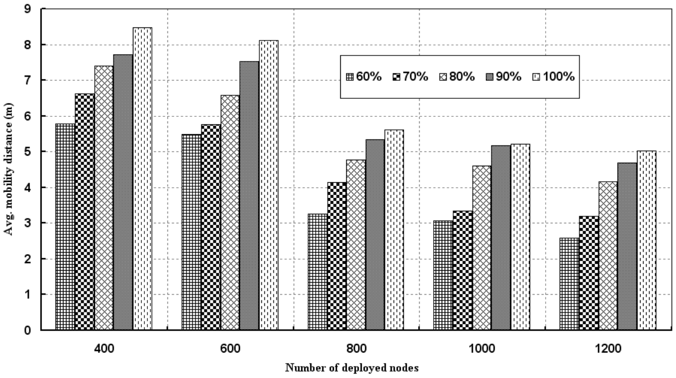


Fig. 10. Average mobility distance for recovering different percentage of coverage hole area against different number of deployed nodes. The sensing range of each node is fixed at $R_s = 20m$.

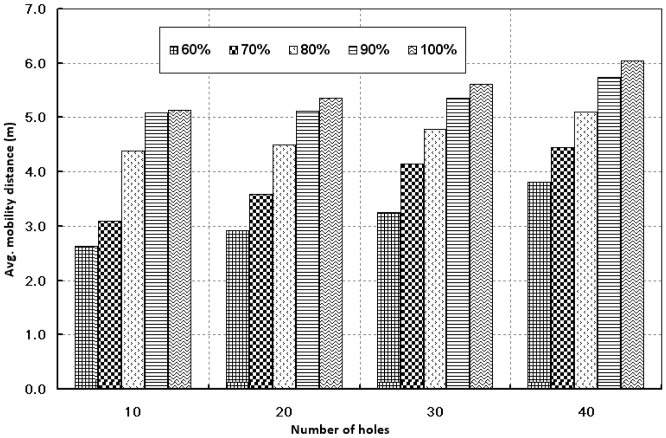


Fig. 11. Average mobility distance for recovering different percentage of coverage hole area against different number of coverage holes. The sensing range of each node is fixed at $R_s = 20m$.

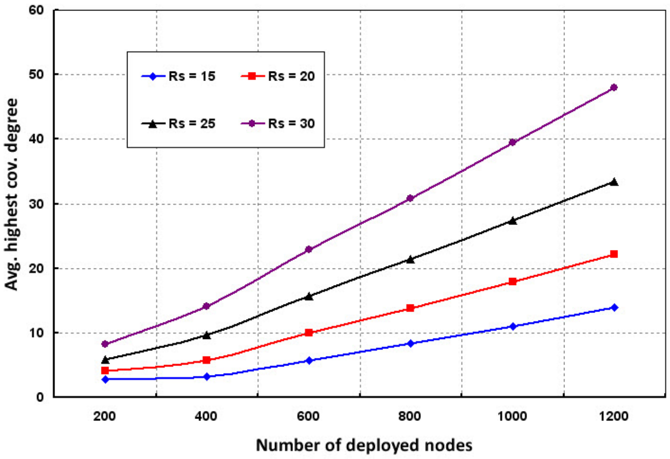


Fig. 12. Average highest coverage degree for different sensing range of each sensor against different number of deployed nodes.

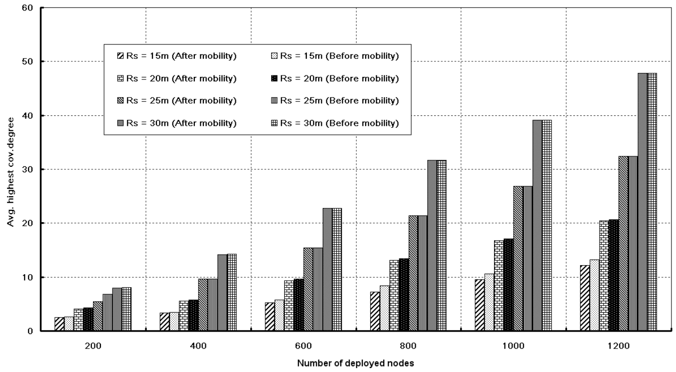


Fig. 13. Change in average highest coverage degree for different sensing range of each sensor before and after the mobility against different number of deployed nodes.

The average mobility distance for different numbers of coverage hole is simulated and the result is shown in Fig. 11. As expected, it is noticed that the the average mobility distance is increased if number of coverage holes are increased. However, to recover the coverage of smaller number of holes, the average mobility distance is less. In order to get higher percentage of coverage recovery, the average mobility distance is increased for fixed number of coverage holes. From Figs. 10 and 11, it is observed that average mobility distance in our protocol is limited within only 10 meters, which is about one-fourth of the one-hop distance.

The average highest coverage degree (*k*-coverage) of a node is defined as the highest *k*-coverage due to sensing range overlapping with its neighbors. As long as a node is static, its highest coverage degree remains unchanged unless any of its one-hop neighbor is not dead. However, if a sensor moves from one location to another, it may happen that the mobile node may increase the *k*-coverage of the nodes located in the target area. However, in our protocol, we are careful not to increase the highest coverage degree of a node by the mobile sensor. As shown in Fig. 12, we have simulated our protocol with different number of static nodes. It is observed that average highest coverage degree increase if sensing range of each is increased uniformly. Besides, the average coverage degree is also increased with increase in number of deployed nodes. In order to verify the correctness of our algorithm, we have simulated our protocol for different sensing ranges with different number of nodes and have verified the average highest coverage degree before and after mobility of the nodes, as shown in Fig. 13. It is noticed that though the average highest coverage degree increases with increase in number of deployed nodes, it remains almost same before and after mobility of the nodes irrespective of increase in sensing range.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we design a low mobility coverage hole recovery protocol using vector algebra as a virtual method to decide the magnitude and direction of a mobile node. Vectors and resultant vectors are drawn from the points of intersection of sensing discs of different sensors. Decision of mobility direction and magnitude is based on the direction of the resultant vectors drawn from the network. Besides, decision of mobility is done by a sensor from its local information obtained from its one-hop neighbors and therefore complexity of making decision is limited. Since, limited nodes those enclose the coverage hole may move, energy consumption due to mobility is also limited. It is observe from our performance evaluation that the mobility of the nodes is limited within their one-hop, which may be maximum up to $10m$. In future, we will simulate our protocol to get more performance evaluation and to compare them with similar hole recovery protocols.

REFERENCES

[1] M. B. McMickell, B. Goodwine, and L. A. Montestruque, ”MICAbot: A Robotic Platform for Large-Scale Distributed Robotics”, in *Proc. IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1600-1605, Taipei, Taiwan, Sep. 2003.

[2] G. T. Sibley, M. H. Rahimi, G. S. Sukhatme, ”Robomote: A Tiny Mobile Robot Platform for Large-Scale Sensor Networks”, in *Proc. IEEE International Conference on Robotics and Automation*, pp. 1143-1148, USA, Sep., 2002.

[3] S. Ganeriwal, A. Kansal, and M. B. Srivastava, ”Self aware actuation for fault repair in sensor networks”, in *Proc. IEEE International Conference on Robotics and Automation*, vol 5, pp. 5244-5249, Spain, May, 2004.

[4] A. Sekhar, B. S. Manoj, and C. S. R. Murthy, ”Dynamic Coverage Maintenance Algorithms for Sensor Networks with Limited Mobility”, in *Proc. IEEE International Conference on Pervasive Computing and Communications*, pp. 51-60, USA, Mar, 2005.

[5] 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”, in *Proc. International Symposium on Distributed Autonomous Robotic Systems*, pp. 299-308, Japan, Jun., 2002.

[6] G. Wang, G. Cao, and T. L. Porta, ”Movement-assisted sensor deployment”, *IEEE Transactions on Mobile Computing*, vol. 5(6), pp. 640-652, Jun., 2006.

[7] C. H. Wu, K. C. Lee, and Y. C. Chung, “A Delaunay Triangulation Based Method for Wireless Sensor Network Deployment,” *IEEE Parallel and Distributed Systems*, vol. 1, July, 2006.

[8] B. Cărbunar, A. Grama, J. Vitek, and O. Cărbunar, “Redundancy and Coverage Detection in Sensor Networks,” *ACM Transactions on Sensor Networks)*, vol. 2, Feb., 2006.

[9] F. Ye, G. Zhong, J. Cheng, S. Lu, L. Zhang, “PEAS: a robust energy conserving protocol for long-lived sensor networks,” *IEEE Distributed Computing Systems*, pp. 28-37, May, 2003.

[10] H. Zhang and J. C. Hou, “Maintaining Sensing Coverage and Connectivity in Large Sensor Networks,” *Journal on Ad Hoc & Sensor Wireless Networks*, vol. 1, pp. 89-124, March, 2005.

[11] W. Wang, V. Srinivasan, and K. C. Chua, “Trade-offs Between Mobility and Density for Coverage in Wireless Sensor Networks,” *ACM International Conference on Mobile Computing and Networking*, pp. 39-50, 2007.

[12] P. K. Sahoo, J.-P. Sheu, and W.-S. Lin, “Dynamic Coverage and Connectivity Maintenance Algorithm for Wireless Sensor Networks,” *IEEE Communication Systems Software and Middleware*, pp. 1-9, Jan., 2007.

[13] G. Tan, S. A. Jarvis and A. M. Kermarrec, ”Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks”, *IEEE Transactions on Mobile Computing*, vol. 8(6), pp. 836-848, 2009.

[14] W. Wang and S. Lafortune, ”An algorithm for maximising covered area”, *International Journal of Control*, vol. 81(9), pp. 1493-1505, Sep., 2008.

[15] H. M. Ammari and S. K. Das, ”Integrated coverage and connectivity in wireless sensor networks: A two-dimensional percolation problem”, *Ieee Transactions on Computers* , vol. 57(10), pp. 1423-1434, 2008.

[16] D. Wang, B. Xie and D. P. Agrawal, ”Coverage and lifetime optimization of wireless sensor networks with gaussian distribution”, *IEEE Transactions on Mobile Computing*, vol. 7(12), pp. 1444-1458, 2008.

[17] X. Wu, J. Cho, B. J. D’Auriol and S. Lee, ”Mobility-assisted relocation for self-deployment in wireless sensor networks”, *IEICE Transactions on Communications*, vol. E90b(8), pp. 2056-2069, 2007.

[18] B. Karp, and H. T. Kung, ”Greedy Perimeter Stateless Routing for Wireless Networks”, in *Proc. ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 243-254, USA, Aug., 2000.