Covering Points of Interest with Mobile Sensors

Milan Erdelj, Tahiry Razafindralambo, David Simplot-Ryl

Covering Points of Interest with Mobile Sensors

Milan Erdelj INRIA Lille, France
milan.erdelj@lifl.fr Tahiry Razafindralambo INRIA Lille, France
tahiry.razafindralambo@inria.fr David Simplot-Ryl Univ. Lille 1, France
David.Simplot-Ryl@inria.fr

Abstract—The coverage of Points of Interest (PoI) is a classical requirement in mobile wireless sensor applications. Optimizing the sensors self-deployment over a PoI while maintaining the connectivity between the sensors and the sink is thus a fundamental issue. This article addresses the problem of autonomous deployment of mobile sensors that need to cover a predefined PoI with a connectivity constraint. In our algorithm, each sensor moves toward a PoI but has also to maintain the connectivity with a subset of its neighboring sensors that are part of the Relative Neighborhood Graph (RNG). The Relative Neighborhood Graph reduction is chosen so that global connectivity can be provided locally. Our deployment scheme minimizes the number of sensors used for connectivity thus increasing the number of monitoring sensors. Analytical results, simulation results and real implementation are provided to show the efficiency of our algorithm.

1 Introduction

Wireless sensor networks have received a lot of attention in recent years due to their potential applications in various areas such as environment monitoring [11], [2]. Covering and monitoring events from the environment in a given area are difficult tasks. Indeed, sensors have to be correctly placed to monitor the events and a connection between the monitoring sensors and a base station (sink) have to be kept to report data.

In this context, sensor placement can be divided into off-line and online schemes. Although off-line deployments can provide optimal placement of sensors, they require precise knowledge of the events' locations. Online deployments can cope this drawback but are only feasible when sensors have motion capabilities. However, the main advantage of online deployments is the possibility to obtain particular topologies which can provide properties such as connectivity.

In classical wireless sensor deployment, communications follow a N to 1 paradigm, that is, all the sensors have to report the sensed data to a sink or a base station. Unlike ad hoc networks, communication between two sensors is not considered. However, a sensor can play a forwarding role for other sensors but all the data packets have only one destination (the base station). While considering this communication paradigm, most of the sensor deployment schemes proposed in the literature can be optimized. Indeed, in these deployments, network connectivity is evaluated based on a N to N communication paradigm. Mobile sensor deployment allows to control the resulting connectivity graph of the network and thus can strongly increase the quality of such deployment.

The placement of sensors related to coverage issues is intensively studied in the literature, and can be divided into three categories. *The full coverage* problem aims at covering the whole area. Sensors are deployed to maximize the covered area [4]. *The barrier coverage* problem aims at detecting intrusion on a given area. Sensors have to form a dense barrier in order to detect each event that crosses the barrier [6]. *Point of Interest coverage* aims at monitoring specific points in the field of interest [7]. Different examples and results related to the deployment of sensors can be found in [22]. These coverage requirements can be either provided using offline or online deployment.

Previous works on Points of Interest (PoI) coverage using mobile sensors, such as [7], do not consider the use of a base station where sensors have to report data and to which a sensor have to be permanently connected either directly or in a multi-hop fashion. The use of a base station in PoI coverage increases the deployment complexity since a connectivity constraint is added.

In this article, we report a solution that solves the PoI coverage problem. We consider a network composed by mobile sensors and a base station (data sink). We also assume that at the beginning of the deployment the sensors are connected to the base station. In our deployment solution, connectivity is the main constraint and, therefore, is maintained all along the deployment procedure by a local control of the topology.

In the proposed solution, each sensor moves toward a PoI but has also to maintain the connectivity with a subset of its neighboring sensors. Depending on the chosen subset of neighbors, keeping these local connections can provide a global connectivity of the network. Such a subset is chosen based on results from the literature of graph theory. Relative Neighborhood Graphs (RNG) or Gabriel Graphs (GG) are examples of such graphs. Once global connectivity can be provided locally, we want the sensors to deploy in such a way that the number of sensors used for connectivity is minimized and the number of sensors that covers the PoI is maximized.

The main contribution of this paper is a deployment

algorithm that has the following properties:

- Our algorithm achieves *PoI coverage*. Examples of static, moving and multiple PoI coverage are provided.
- *Connectivity* between each sensor and the base station is kept all along the deployment procedure.
- Our algorithm is *local* i.e., every decision taken is based on local neighborhood information only and does not require synchronization.
- It is *efficient*, since it minimizes the number of connectivity sensors and maximizes the number of covering sensors.

The rest of this paper is organized as follows: Section 2 provides some backgrounds which include state of the art, assumptions, definitions and a problem statement. Section 3 describes our deployment algorithm with its properties. Simulation results are given in Sections 4, 5 and 6 in which we consider static PoI, moving PoI and multiple PoIs respectively. Real implementation of our algorithm using Wifibots[1] is presented in Section 7 and conclusions are drawn in Section 8.

2 Background and assumptions

2.1 State of the Art

In this section, papers about deployment and self-deployment of wireless sensor networks are reviewed and we shortly extend this state of the art to mobile robots deployment. As our main focus is Point of Interest coverage with connectivity constraint, we only cite papers that consider these two properties. Moreover, we consider the deployment of mobile sensors but more interested readers can refer to [4], [16] for static random deployment strategies, to [8], [14], [1] for off-line computation of sensor placement and to [22] or [19] for complete surveys. There are mainly three ways to optimize the deployment or the placement of mobile sensors that were previously described in [19].

The coverage pattern based movement [20]. In this category, target locations of the sensors are computed based on a predefined regular pattern such as hexagons. The final positions of the sensors can be given at the beginning of the deployment (global coverage). Or, a particular sensor plays a specific role and helps the other neighboring sensors to find their final positions based on the seed's position. With this strategy, connectivity is not provided all along the deployment procedure. Moreover, the coverage pattern based movement is not suitable for PoI coverage.

Grid quorum based movement [5]. In this category, the sensors' field is partitioned into many small grid cells, and the number of sensors in each cell is considered as the load of the cell. Coverage and connectivity requirements depend on the grid size. The sensor's mobility is viewed as a classical load balancing problem of each cell. As in coverage pattern based movement, this deployment strategy cannot guarantee connectivity and cannot provide PoI coverage.

Virtual force based movement [3]. In this category, sensors are repelled or attracted each other by using virtual forces like electromagnetic particles. The sensors move step by step. The virtual forces are computed based on the set or a subset of neighboring sensors and allow the computation of the sensor's next movement. The sensor can undergo attractive forces, for preferential coverage areas, repulsive forces for obstacle avoidance and forces exerted by another sensor. With this deployment strategy connectivity and PoI coverage can be provided. The work proposed in this paper belongs to this category.

Movement is the way the sensor moves depending on the application. The coverage requirement is the primary aim that describes how the sensors have to be deployed over the field. Even if some ways of moving are strongly related to the coverage requirements, it is important to notice that movement and coverage are independent. From our point of view, these two aspects must be decorrelated in order to have simple deployment algorithms. Coverage requirements can be divided into three categories:

1) In the full coverage problem, sensors have to maximize the covered area. The work proposed in [17] and [12] uses virtual force based movement to increase the covered area. The main difference of these two works is the connectivity consideration. In [17], a connectivity checking procedure is implemented. That is, a specific sensor regularly floods the networks, and a sensor that does not receive the flooding message considers itself as disconnected from the rest of the network. Thus, the disconnected sensor moves back to its previous position. In [12], authors use local geometry and potential field theory to maximize the area covered by mobile robots. They use a Neighbor-Every-Theta (NET) graph to compute the robot's movements. The authors apply the forces described in [13]. By using a combination of mutually opposing forces, each sensor maximizes its coverage and maintains the NET condition of having at least one neighbor in every θ sector.

2) In barrier coverage problem, sensors must form a barrier that detects any event crossing the barrier. A barrier is defined as a segment between two points of the sensor field between which the sensors have to be evenly distributed. In the work proposed in [15], authors use virtual forces to relocate the sensors. The repulsive forces are used to have an uniform distribution of the sensors. On the other hand, attractive forces are used to gather sensors into the same horizon. It is important to notice here that, when the number of sensors is sufficiently large, connectivity can be provided at the end of the deployment. However, it is hard to guarantee connectivity all along the deployment procedure.

3) In the PoI coverage, only some specific points of the sensor field need to be monitored. Surprisingly, very few works consider the problem of PoI coverage. To the best of our knowledge, the only work that consider

1. http://www.wifibot.com

PoI coverage is [7]. In [7], authors propose an algorithm to periodically monitor some specific points (instead of all along). Unlike the work presented in this paper, results from [7] do not consider connectivity issue. In [9], authors developed an algorithm to deploy the sensors around a PoI following a triangle tesselation. In this work, the PoI is not covered by all the sensors and is only used as a focus point.

In this paper, we consider single and multiple PoI coverage where connectivity has to be kept between the sensors that cover the PoIs and a base station (or sink). Moreover, we increase the connectivity constraint and provide an algorithm in which connectivity is kept all along the deployment procedure.

2.2 Motivating application

A typical scenario in wireless sensor networks is environment monitoring. The sensors have to be deployed and placed on strategic locations to monitor the area of interest. In many cases, monitoring the whole area might be unnecessary. Therefore, monitoring some points of interest increases the sensing performance and reduces the deployment cost since the number of sensors that monitor the area can be increased by a given fixed number of sensors. When sensors have motion capabilities, monitoring only some PoIs instead of the whole area also allows time dependent coverage.

In this article, we consider an environmental monitoring application. We assume that an event is detected by an external entity and not by the mobile wireless sensor network itself. Moreover, we assume that this external entity can precisely define the event's location. When the event is detected, it's position is sent to the mobile sensors through a fixed base station. The mobile sensors, then, self-deploy to monitor this event and report data (such as temperature, humidity, video, etc.) to the sink in a multi-hop fashion.

In this application, it is possible to have more than one event and these events can be mobile. In this context, the deployment has to adapt it's behavior depending on evolving requirements. In order to dynamically adapt to the changing requirements, the deployment algorithm must provide properties such as connectivity all along the deployment procedure. This enables the base station to change the position and/or to add another PoI even during the deployment procedure.

2.3 Preliminaries

We use the following definitions and notations for the network model.

Definition 1: Let $G(V, E)$ be the graph representing the sensor network. V is the set of vertices each one representing a sensor. $E \subseteq V^2$ is the set of edges; $E = \{(u, v) \in V^2 \mid u \neq v \ \wedge \ d(u, v) \leq R\}$, where $d(u, v)$ is the euclidean distance between sensors u and v and R is the communication range. $G(V, E)$ is our model of the sensor network.

Definition 2: $N(u) = \{v \in E \mid d(u, v) \leq R\}$. $N(u)$ is the set of 1-hop neighbors of sensor u.

Assumption 1: We assume that each sensor has its position denoted by $(x(u), y(u))$ for sensor u. This position can be provided by any internal mechanisms or external entities such as GPS.

Assumption 2: We assume that at the beginning of the deployment the sensors are randomly spread out around the base station at a maximum distance of $d < R/4$ from the base station. This condition ensures that the network is connected.

2.4 Relative Neighborhood Graph

The Relative Neighborhood Graph (RNG) [18] is a graph reduction method. Given an initial graph G, the RNG graph extracted from G is a graph with a reduced number of edges but the same number of vertices. Let the sensors be the vertices of the initial graph and that there exists an edge between two vertices if the two sensors can communicate directly. We assume here that the communication between two sensors is possible only if the distance between them is less than a given communication range. To build an RNG from an initial graph G, an edge that connects two sensors is removed if there exists another sensor that is at a lower distance from both sensors. The formal definition of the RNG graph is as follows:

Definition 3: Let $RNG(G)$ be the relative neighborhood graph extracted from $G(V, E)$. $RNG(G) = (V, E^{rng})$, where $E^{rng} = \{(u, v) \in E \mid \nexists w \in (N(u) \cap N(v)) \wedge d(u, w) < d(u, v) \wedge d(v, w) < d(u, v)\}$.

Definition 4: $RNG(u)$ is the set of u's neighbors which are part of the $RNG(G)$ graph $RNG(u) = \{v | v \in N(u) \cap RNG(G)\}$. We denote by $|RNG(u)|$ is the number of sensor in $RNG(u)$.

Definition 5: $RNG^+(u)$ (resp. $RNG^-(u)$) is the furthest sensor that is part of $RNG(u)$, the distance between u and $RNG^+(u)$ (resp. $RNG^-(u)$) is denoted by $d^+(u)$ (resp. $d^-(u)$).

Using the RNG reduction has two main advantages. First, the RNG reduction can be computed locally by each sensor since sensors only need the distances with its neighbors [18]. Second, given that the initial graph is connected, the RNG reduction is also connected. These two properties are important for scalability and connectivity preservation. Indeed, to preserve the connectivity of the whole network, each sensor has to preserve the connectivity with its neighbors that are part of the RNG graph. In our algorithm, we use these properties to preserve connectivity and to ease the movement computation.

3 Deployment algorithm for PoI coverage

3.1 Basic idea

At the beginning of the deployment, all the sensors are in the communication range of the base station and

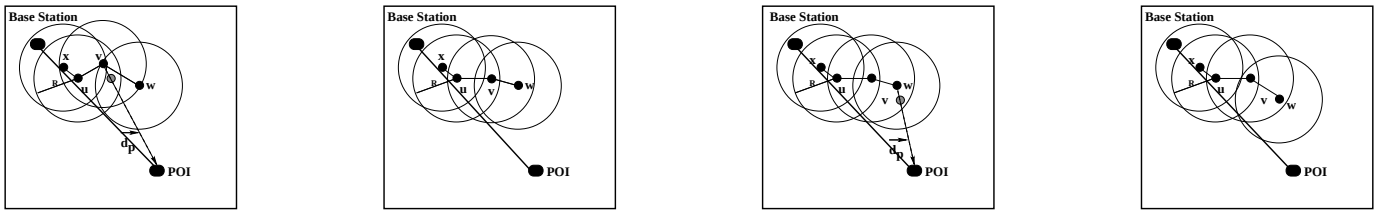


Fig. 1. Example of sensor movement.

all the sensors are also within communication range of each other. Each sensor moves independently from the other sensors. The sensors are not synchronized, motion decisions are taken individually and all the sensors run the same algorithm. It is important to notice here that the base station can compute an optimal placement and can provide this location to each sensor which can move toward this optimal position. However by doing so, it is hard to ensure that the network is connected all along the deployment procedure. Therefore, when tracking a moving PoI, some sensors may not have an up-to-date position and placement.

In order to cover the PoI, the sensors move toward one predefined point that may be chosen randomly within the set of PoIs. This movement toward the PoI is constrained by the connectivity requirements. While they are moving, the sensors must keep connected with their neighbors that are part of the RNG reduction (also called RNG neighbors) of the dynamic graph. Indeed, even if a sensor does not cover the PoI, it must stop moving to keep the connection with its RNG neighbors. It is worth noting that, when a sensor covers the PoI, it also stops its movement. The direction of a sensor is thus given by the following unit vector : $\overrightarrow{\Delta} = \overrightarrow{d_p}/||\overrightarrow{d_p}||$, where $\overrightarrow{d_p}$ is the vector toward the PoI. When a sensor has computed its direction, it will move toward this direction. However, the distance covered by the mobile sensor is constrained by the connection with its RNG neighbors. The movement vector of a sensor is thus $\overrightarrow{m} = d \cdot \overrightarrow{\Delta}$, where d is the maximum distance that the sensor covers while maintaining connectivity with its RNG neighbors. If $d^+(u)$ is the distance of sensor u and its farthest RNG neighbor, $d \leq (R - d^+(u))/2$, where R is the communication range. This condition ensures that when considering worst case movements u and $RNG^+(u)$ remain connected.

The Figure 1 shows an example of movements. We can see how sensors move toward the PoI and how connectivity is preserved by maintaining the connectivity with the RNG neighbors. It is worth noting that x is a neighbor, but not an RNG neighbors, of sensor v. We can also notice in this figure that sensor v does not move at distance R of sensor u. This is due to the upper bound on the distance $d \leq (R - d^+(u))/2$. The fact that sensor v is not at distance R from sensor u also helps to have a straight line deployment between the base station and the PoI since after each iteration the sensor v moves toward the PoI and toward the segment between the PoI and the base station.

In order to avoid an infinite small movements of sensors, we add a condition on d. That is : If $d < \epsilon_1$, with $\epsilon_1 > 0$ then we set $d = 0$. Note also that for termination purpose, if the distance between a sensor and the PoI is lower than a given threshold ϵ_2, with $\epsilon_2 > 0$ (related to sensing range), the sensor stops moving.

3.2 Algorithm sketch

Algorithm 1 formally describes our deployment process. Our algorithm is divided into three parts:

Algorithm 1 Deployment process

Part 1 — Direction computation on sensor u:

1: $\overrightarrow{\Delta} = \dfrac{\overrightarrow{d_p}}{||\overrightarrow{d_p}||}$

Part 2 — Distance and speed computation on sensor u:

1: $d = (R - d^+(u))/2$
2: $\nu = \frac{d}{\delta}$, δ is the periodicity of movement decision, ν the speed

Part 3 — Motion of sensor u:

1: move to u_{new} using :
2: -speed ν.
3: -direction $\overrightarrow{\Delta}$.
4: -distance d.
5: -Take obstacle into account.

Part 1: In this part, the sensor computes its direction based on its own position and the coordinate of the PoI. At the end of Part 1, the sensor knows its direction $\overrightarrow{\Delta}$.
Part 2: In this part, the sensor computes the distance it has to cover. This distance must take into account connectivity constraint by considering the worst case movement of the RNG neighbors of the sensor. Since connectivity with farthest RNG neighbor must be kept, moving distance should always be $d \leq (R - d^+(u))/2$. Recall that sensors run the same algorithm. Therefore, if a sensor $v = RNG^+(u)$, $d(u, v) \leq d^+(v)$. This inequality ensures that if a sensor v is the farthest RNG neighbor of a sensor u, and sensor u is not the farthest RNG neighbor of a sensor v, connectivity is still kept between these two sensors. Note that using virtual force based movement implies a step by step computation of sensor's movements. In our algorithm, δ is the movement decision

period given in seconds. Based on the value of δ and d, we can easily compute the speed of the sensor. At the end of Part 2, the sensor knows it's direction, it's speed and the distance it has to cover.

Part 3: In this part, the path planning of the sensor is considered. With a direction, speed and distance being known, any path planning algorithm can be used for sensor movement. The distinction between Part 3 and Part 2 allows us to use a path planning and obstacle avoidance algorithm from robotics such as the one described in [10]. The development and the integration of an efficient obstacle avoidance scheme (from the literature) in our algorithm is left to future work.

The three parts of our algorithm are related to three important aspects of deploying a fleet of mobile sensors. The first part is related to the coverage requirements. The second part is related to the connectivity preservation and the third part is related to sensor's movement. Since our algorithm is divided into three independent parts, it is simple to modify each part independently from the other parts. It is thus easy to modify the direction computation while maintaining connectivity and by using the same path planning algorithm.

3.3 Algorithm properties

Theorem 1: Connectivity. If at time $t = T$ the graph is connected, $\forall t = i, i > T$ the resulting graph at time $t = i$ is connected.

Proof: In an asynchronous environment, sensors can run algorithm 1 at any time. Let u and v be two sensors and u and v are connected a time $t = T$. Let $u \in RNG(v)$, $v \in RNG(u)$ and $d(u, v) = d^+(u)$. Let us assume that two sensors run Algorithm 1 at the same time and that they are moving in the opposite direction of each other. The maximum distance covered by sensor v depends on $d(u, v)$. Since $d(u, v) \leq d^+(v)$ the maximum distance covered by sensor v is $d_v = (R - d^+(v))/2 \leq (R - d^+(u))/2$. Therefore, the maximum distance between sensor u and v after their respective movement is $d(u, v) + (R - d^+(u))/2 + (R - d^+(v))/2 \leq R$. Thus, after their respective movement, sensors u and v are still connected. If the connection to the farthest RNG neighbor is maintained, the connection to closer RNG neighbors is also maintained and if the connectivity with RNG neighbors is kept, network connectivity is thus also kept [18]. □

Theorem 2: Termination. There exists a time $t > T$ when all the sensors stop moving.

Proof: Let us observe a case where sensor deployment is composed of the base station b, the PoI p and the mobile sensor u. At the beginning of the deployment, at $t = 0$, $d(u^{(0)}, b) < R$. After the first iteration $d(u^{(1)}, b) = d(u^{(0)}, b) + (R - d(u^{(0)}, b))/2$ after the i^{th} iteration,

$$d(u^{(i)}, b) = \frac{1}{2^i}\left((2^i - 1)R + d(u^{(0)}, b)\right), \quad (1)$$

we thus have:

$$\lim_{i \to \infty}\left(R - d(u^{(i)}, b)\right) = 0 \quad (2)$$

Therefore, there exists a $t > T$ such that at time T sensor u runs its i^{th} iteration and thus $R - d(u^{(i)}, b) < \epsilon_1$ and sensor u stops moving.

The same proof holds recursively for an arbitrary number of sensors. Indeed, we have to consider the time T where the sensor u_1 that is closer to b stops moving and apply the proof while considering that u_1 plays the role of the base station. Moreover, the T needed to stop deployment can be reduced since when $d(u, p) < \epsilon_2$, the sensor u also stops moving. □

Theorem 3: Straight line deployment. Let b be the base station, p be the PoI and let us assume that sensor u is not on the segment $[b, p]$. The distance h between a sensor and the segment $[b, p]$ is strictly decreasing.

Proof: At each step of the deployment, sensor u moves toward the PoI. Since the direction of the sensor is $\overrightarrow{up}$, where u is the sensor's position and the covered distance is $d \geq 0$, the distance between a sensor and the PoI is strictly decreasing. As a consequence, the distance between the sensor and the segment $[b, p]$ is also decreasing. It is worth noting that when the sensor $u \in [b, p]$, it remains in the segment during movement and $h = 0$. □

Theorem 4: Minimizing number of connectivity sensors. If the PoI p is at a distance $d = \infty$, each sensor has at most two RNG neighbors at the end of the deployment.

Proof: We consider in this theorem, without loss of generality, that the PoI is at distance $d = \infty$ for two reasons. First, this assumes that sensors are thus moving in a parallel to the segment $[b, p]$. Second, this ensures that no sensor reaches the PoI and thus that all sensors are connectivity sensors.

If the deployment reaches termination, this means that at least the distance between a sensor u and one of its neighbors v is $d(u, v) > R - \epsilon_1$. To better understand the proof, let us consider the configuration depicted in Figure 2.

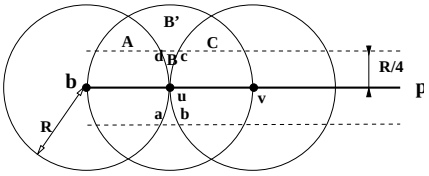


Fig. 2. Straight line deployment.

In this configuration, the sensors u and v cannot move anymore since they are at distance $R - \epsilon_1$ of b and u respectively. It is also important to notice here that due to Theorem 3 the sensors stay at most at a distance $R/4$ of the segment $[b, p]$. Let us now assume that sensor u can have more than two RNG neighbors and that the deployment reaches termination. In this case, we need to place a sensor w in such way that $w \in RNG(u)$, and $\exists x \in N | d(x, w) > R - \epsilon_1$. In the configuration depicted in

Figure 2, w must fall in the surface indicated by A, B, B' or C.

Case A or C: If w falls into surface A (or C), $w \in RNG(u)$, but $b \notin RNG(u)$ (or $v \notin RNG(u)$). Therefore, $d(b,w) \leq R - \epsilon_1$ (or $d(v,w) \leq R - \epsilon_1$) and w can move. Which is contrary to our assumption that the deployment reaches termination.

Case B': sensor w cannot fall into surface B' due to Theorem 3, since we assume that sensors are at the maximum distance $d = R/4$ of the base station at the beginning of the deployment.

Case B: If sensor w falls into surface B, $w \in RNG(u)$ and Theorem 3 is verified. However, if $w \in B$, $d(u,w) \leq R - \epsilon_1$ and thus w can move, which is contrary to our assumption that the deployment reaches termination. This proof can be extended to any configuration since the maximum distance between u and the set of intersection points a, b, c and d is $\max_{i=\{a,b,c,d\}} d(u,i) = R\sqrt{2-\sqrt{3}}$. This is the case if we observe intersection d when b and u are located on the bottom dashed line. Therefore, if $w \in B$, then $d(u,w) \leq R\sqrt{2-\sqrt{3}} < R - \epsilon_1$, $\forall \epsilon_1 < R(1-\sqrt{2-\sqrt{3}})$ □

Theorems above show that our algorithm preserves connectivity all along the deployment procedure. Furthermore, we bring proofs that the deployment will eventually terminate and that, at the end of the deployment, sensors used for connectivity are more likely to form a straight line and to be at distance $R - \epsilon_1$ from their neighbors. We also show that, at the end of the deployment, each sensor used for connectivity has at most two RNG neighbors, which proves that the number of connectivity sensors is minimized.

4 Static PoI

This section shows the performance evaluation results of our algorithm. Simulations were performed using WSNet[2]. In the simulations, we set the communication range to be equal to the sensing range but this assumption can be easily modified without affecting the behavior of the deployment. In this paper, we mainly focus on connectivity for PoI coverage. Therefore, comparisons with other works are hard to provide since literature lacks similar algorithms. Note that in the simulations δ is set to $5s$.

4.1 Deployment example

Figure 3 shows an example of the deployment's evolution where the PoI is located at position $[70, 100]$. After $180s$, the deployment is finished. In the simulation setup, the sensors move during five seconds and compute a new direction after their movements. This figure shows that the sensors form a straight line between the base station and the PoI which reduces the number of sensors used for connectivity preservation and therefore increases the number of sensors involved in coverage.

2. http://wsnet.gforge.inria.fr

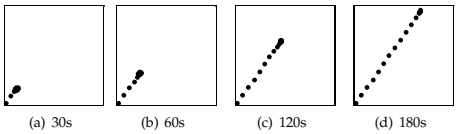


Fig. 3. Evolution of sensors' positions depending on time. In this simulation there are 20 sensors with a range of 10 on a square of 100×100. The PoI is located at $[70, 100]$

4.2 Coverage quality

The Figure 4(a) presents the number of covering sensors w.r.t. the distance between the PoI and the base station. In the simulation, the base station is considered as a sensor which is not mobile. That is, we consider 20 sensors including the base station. This figure shows that the number of sensors used for connectivity is minimized and that the number of covering sensors is maximized. For example, when the PoI is at distance 40, we need 3 sensors for connectivity at distances 10, 20, 30 and the base station at distance 0, which means that 4 sensors are needed for connectivity and 16 sensors can cover the PoI for a total of 20 sensors.

4.3 Deployment speed

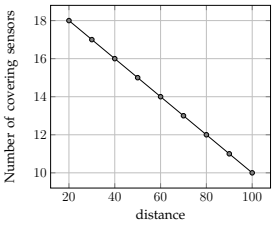
The Figure 4(b) plots the number of covering sensors depending on time. In this simulation, PoI is at distance 100 and 20 mobile sensors are considered. A movement decision is taken every $\delta = 5s$. This figure shows that the first PoI is covered by at least one sensor after $120s$. Note here that we check the coverage every $1s$. This means that the first covering sensor has a mean speed of $0.75m/s$ ($90m$ covered distance after $120s$).

Note that in an ideal case, the distance a sensor can cover at each step is R meters (the communication range) and each step lasts for $\delta = 5s$ (motion decision). This means that the maximum (in the best case) speed of a sensor is : $\nu = 10/5 = 2m/s$. Note here that since we want to ensure connectivity, a sensor must consider the worst case movement of it's neighbors since we assume that the sensors are not synchronized. Therefore, the maximum speed is reduced to $\nu = 2/2 = 1m/s$ to take the connectivity constraint into account. These results are very close to the results obtained above. One way to increase deployment speed is to reduce the motion decision period or to increase the communication range.

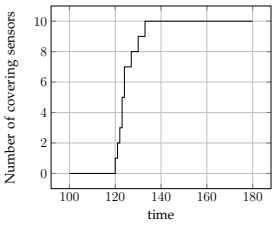
4.4 Energy consumption

To evaluate the energy consumed by each sensor during the deployment, we consider a simple energy model where the energy consumed by a sensor u is: $E(u) = d\alpha + \beta$, where d is the covered distance and α and β are constants (here, $\alpha = 1$, $\beta = 1$). This simple energy model considers the distance covered by a sensor but also penalizes multiple small movements.

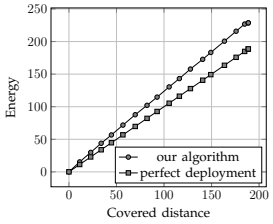
Figure 4(c) shows the energy consumption of each sensor for a deployment of 20 sensors and a PoI at



(a) Number of covering sensors w.r.t distance.



(b) Number of covering sensors w.r.t time.



(c) Energy consumed w.r.t distance.

Fig. 4. Coverage quality, deployment speed and energy consumption.

$[100, 100]$. This figure shows that the energy consumption is linear depending on the covered distance. Moreover, our scheme consumes small amount of energy since (for example) for a covered distance of $105m$, 130 energy units are needed. We can notice that a sensor can cover $R/2 = 5m$ in every movement decision period since it has to maintain connectivity with its neighbors. Therefore, the sensor needs at least $105/5 = 21$ iterations to cover $105m$. The energy consumed by the sensor is at least $E(u) = 105 \times 1 + 1 \times 21 = 126$ which is very close to 130.

The Figure 4(c) shows that the energy consumed by each sensor is related to the covered distance and that the energy overhead is mainly due to the periodic motion decision. In order to reduce energy consumption by removing this periodicity, each sensor can be given its final destination at the beginning of the deployment. However this deployment cannot guarantee connectivity during the deployment, is not robust against obstacles, and is not suitable for the coverage of moving PoI.

5 Moving PoI

In this section, we consider a single moving PoI. Indeed, when the sensors are deployed over a given PoI, the sensing application may require the sensor to move to another location. This scenario is possible with our algorithm since it maintains connectivity all along the deployment procedure. Note here that we consider only one PoI.

5.1 Tracking strategies

There are three different strategies for covering a new PoI when the sensors are already deployed. In the first strategy, hereafter referred to as `STR1`, the sensors first move back to the base station before deploying toward the new location of the PoI. This first strategy provides a high coverage quality but increases the deployment duration and the amount of energy consumed. In the second strategy, hereafter referred to as `STR2`, the sensors try to move directly toward the location of the PoI without going back to the base station. This second strategy reduces the time needed to cover the new PoI but also reduces the coverage quality since an increasing number of sensor is needed to preserve connectivity. The third strategy is a mix of `STR1` and `STR2` and is called `STR3`. In this strategy, sensors move toward the segment $[b, p]$ and when the distance between the particular sensor and the segment is lower than $R/4$, the sensor moves toward the PoI. This strategy combines the advantages of `STR1` and `STR2`.

5.2 Example of deployment for moving PoI

The Figure 5 shows the example of deployment for different tracking strategies. Figures 5(a) to 5(e) show the deployment using `STR1`. We can see from this set of figures that after $450s$ the deployment reaches it's end and that the first covering sensor reaches the PoI between $[350 - 450]s$. Figures 5(f) to 5(j) show the deployment using `STR2`. This set of figures shows that the deployment terminates after 7 hours but that the PoI is reached after $300s$. The long termination time is mainly due to the fact that sensors can only make small movements since they are at a distance close to R of each other. Figures 5(k) to 5(o) show that the deployment using `STR3` terminates after $900s$ and that the PoI is first reached between $[350 - 900]s$, which is a consequence of this deployment strategy being a trade-off between `STR1` and `STR2`.

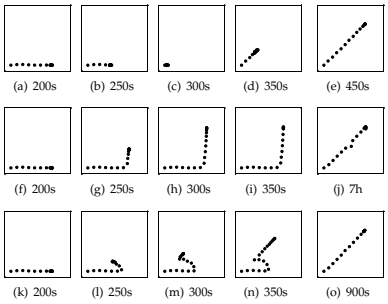


Fig. 5. Evolution of sensors' position depending on time. In this simulation there is 20 sensors with a range of 10 on a square of 100×100. The PoI is first located at $[70, 0]$ and then at $[70, 70]$ after $200s$.

5.3 Coverage quality and deployment speed

In this section we evaluate the coverage quality and the deployment speed of each strategy. We run a simulation

of $3000s$ with 20 sensors and move the PoI at a random location every $500s$. The Figure 6 plots the number of covering sensors depending on time, coverage quality and (re)deployment speed for these three strategies.

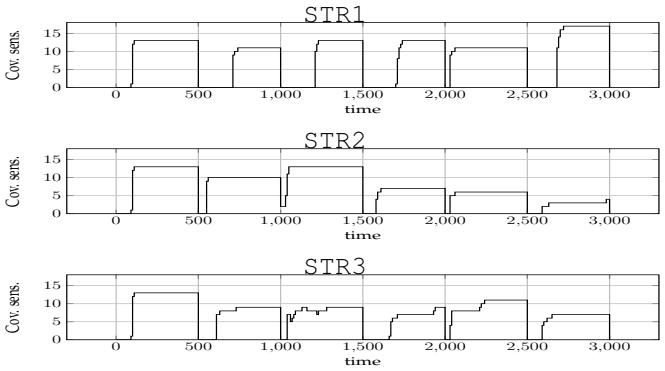


Fig. 6. Number of covering sensors w.r.t time. Simulation parameters: $R = 10$, $Sensing = 10$, 20 sensors including the base station. The simulation lasts $3000s$. A new location of the PoI is chosen every $500s$.

We can see from Figure 6 that each new PoI location is covered by at least one sensor for each strategy. We can also see that from the coverage quality point of view, `STR1` shows very good performances compared to other strategies. Actually, if we consider the coverage of the last PoI (between $[2500-3000]s$), `STR1` has more than 15 covering sensors, `STR2` has less than 5 covering sensors and `STR3` has around 7 covering sensors. More generally, when using `STR1` the coverage quality depends only on the distance between the base station and the PoI which is not the case for `STR2` and `STR3`. From the redeployment speed point of view, `STR2` shows very good performances. We can see for example that between $[1000-1500]s$ the PoI is covered at most after $10s$ (we sample the number of covering sensors every $10s$). For `STR1`, $200s$ are needed and for `STR3`, $30s$ are needed. We can notice here that at time between $[500-1000]$ the PoI is located at $[93, 27]$ and between $[1000-1500]s$ it is at $[75, 1]$.

This section shows that when the PoI is moving or when sensor redeployment is needed, our three proposed strategies have their advantages and drawbacks but they keep the properties described in Section 3.3 such as connectivity and termination. Note that the trade-off proposed with `STR3` can be optimized depending on the application requirements. Moreover, it could be of interest to use `STR1` or `STR2` depending on the distance between the old and the new location of the PoI or any other metric, such as angle. This study is left for future works.

6 Multiple PoIs

In this section, we give some results regarding the coverage of multiple PoIs. We limit our assessment to two static PoIs. However, our algorithm can be applied to more PoIs without modifications.

6.1 Multiple PoI coverage strategy

We have developed two families of deployment strategies for the coverage of multiple PoIs. In our first family, later referred to as F1, we consider each PoI independently. In this case, the base station is responsible of dividing and assigning the set of sensors to each particular PoI. The assignment of a subset of sensors to a given PoI can be done regarding the distance between the PoI and the base station or based on some other criteria. Second family, F2, considers the set of PoIs as a whole. The base station defines some intermediate points that have to be reached by the sensors before effectively covering a PoI. For instance, when two PoIs have to be covered, an intermediate point could be the gravity center of the two PoIs and the base station. Note that we can also consider the Steiner tree to choose intermediate points [21].

6.2 Example of deployment for multiple PoIs

The Figure 7 shows the example of deployment for F1 and F2 respectively. In these simulations, we consider two PoIs at $[90, 50]$ and $[50, 90]$ and 30 sensors. For F1, we consider that the set of sensors is divided into two subsets and each subset is assigned to one PoI. The Figure 7 shows that for F1 the deployment terminates after $180s$ and that the PoIs are considered independently. For F2, we choose the gravity center of the two PoIs and the base station as an intermediate point. In F2 (as in F1), each sensor is also assigned to a given PoI by the base station. However, before effectively moving toward it's PoI, the sensors need to reach the intermediate point. The Figure 7 shows the two steps of the deployment for F2.

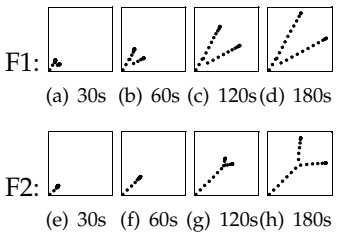


Fig. 7. Sensors' positions with multiple PoIs depending on time.

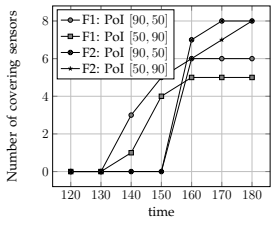


Fig. 8. Number of covering sensors w.r.t time for F1 and F2.

6.3 Coverage quality and deployment speed

Figure 8 plots the number of covering sensors depending on time for the two families of deployment strategies. This figure shows the trade-off performance between deployment speed and coverage quality. Indeed, F1 outperforms F2 regarding deployment speed since the two PoIs are covered by at least one sensor at $140s$ for F1 and this value is $160s$ for F2. However, the coverage quality provided by F2 is better than the coverage provided by

F1 since the maximum number of covering sensors is 6 for F1 and 8 for F2. Note that for F1, the number of covering sensors is not equal for the two PoIs since in our simulation setup 30 sensors are considered, including the base station. Therefore, 14 sensors are dedicated to one PoI and 15 sensors are dedicated to the other. This is not the case for F2 since a subset of sensors is used in common for connectivity.

7 Implementation

This section shows real example of deployment and implementation of our algorithm in order to prove the proposed concept. We also show in this section that even with real radio condition, obstacle condition and without accurate position information our algorithm performs well. Mobile sensors used in this work are Wifibot mobile robotic platforms (visit Wifibot website[3] for more details).

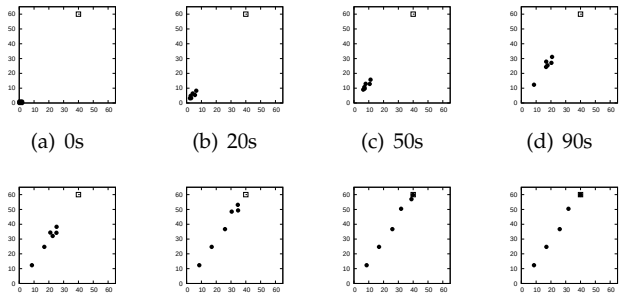
Our experiments were ran indoor and we used dead reckoning technique using motor encoders to get robots' positions during the deployment. The Wifibot has an 802.11b interface which is used in our experiments to send periodic messages containing position data to surrounding robots. It is important to notice that due to indoor conditions and radio instability it was hard to evaluate the real communication range of the robots. Therefore, we fixed the robot communication range arbitrary and discarded messages received from robots that are out of communication range.

In first deployment example, point of interest is at location $[40, 60]$ and communication range of all robots is set to $15m$, target is covered after approximately $160s$ and deployment frames are shown in Figure 9. It can be clearly seen in these examples how deployment actually works: all the robots are moving to a known target with constraints regarding movement forward and constantly preserving connection with the base station. As the group of robots advances towards the target, after reaching the boundary of the communication range, the robot closest to the base station stops with it's movement, thus creating a communication path back to the base station.

Figures 10 and 11 present our deployment examples where target is located at $[70, 100]$, communication ranges are set to $20m$ and $15m$ respectively and with different number of robots. In both examples target is covered after approximately $260s$ which shows that the deployment time is not related to the number of robots or communication range, but just to robots' maximal speed (of course, if a group of robots can reach the target at all).

The situation where group of robots cannot reach the target is shown in Figure 12. Five robots are trying to reach the target with their communication range set to $8m$, but after approximately $90s$ they all reach the end

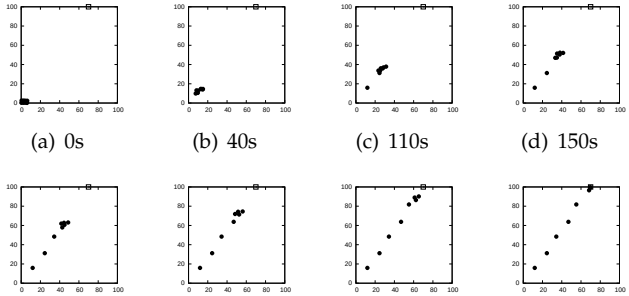
3. http://www.wifibot.com



(a) 0s (b) 20s (c) 50s (d) 90s

(e) 110s (f) 150s (g) 160s (h) 177s

Fig. 9. Single target at $[40, 60]$ with comm. range of $15m$ (6 robots).



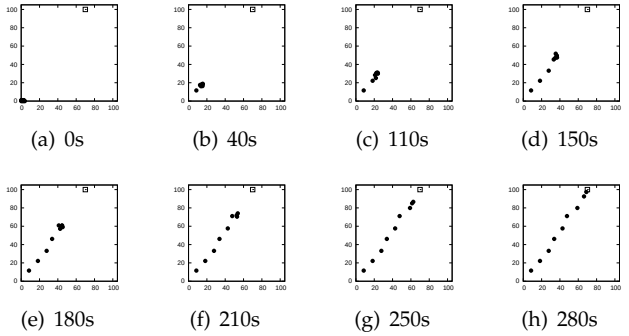
(a) 0s (b) 40s (c) 110s (d) 150s

(e) 180s (f) 210s (g) 250s (h) 280s

Fig. 10. Single target at $[70, 100]$ with comm. range of $20m$ (6 robots).

of their communication range and therefore complete deployments stops.

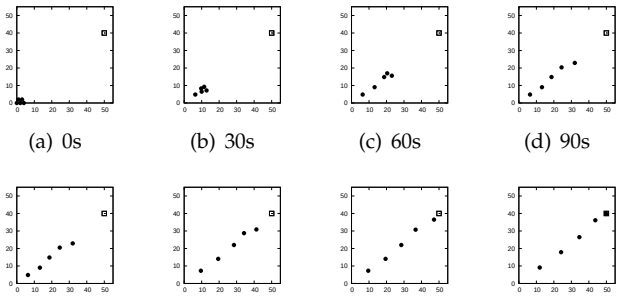
We have also implemented a simple obstacle avoidance since Wifibots are equipped with two IR proximity sensors. When a robot encounters an obstacle it stops moving and considers the obstacle in it's next direction computation. The right (left) hand rule is used to bypass the obstacle depending on the value from each IR sensor. The integration of an efficient obstacle avoidance scheme (from the literature) in our algorithm is left for future work. It is also important to notice that in order to alleviate the effect of messages losses, we increase the



(a) 0s (b) 40s (c) 110s (d) 150s

(e) 180s (f) 210s (g) 250s (h) 280s

Fig. 11. Single target at $[70, 100]$ with comm. range of $15m$ (9 robots).



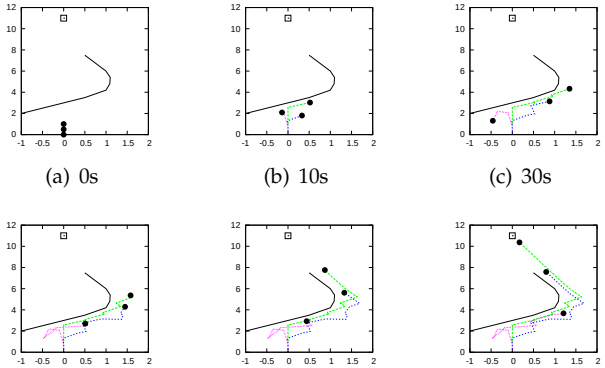
(a) 0s (b) 30s (c) 60s (d) 90s

(e) 0s (f) 30s (g) 50s (h) 80s

Fig. 12. Single target at $[50, 40]$ with changing the comm. range from $8m$ to $15m$ (5 robots).

frequency of Hello messages. Investigation related to optimal Hello messages frequency is left to future works.

Figure 13 shows the example of deployment for a single PoI as presented in Section 4 with an obstacle. In this experimentation, we have 3 Wifibots and a base station at $[0, 0]$. The PoI is at $[0, 11]$ and the communication range is set to $4.5m$ while all other parameters are the same as in simulations.

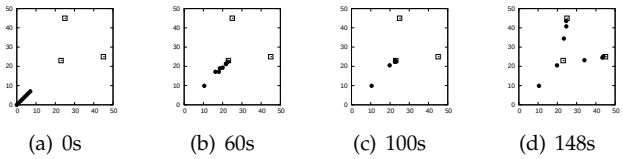


(a) 0s (b) 10s (c) 30s

(d) 40s (e) 50s (f) 60s

Fig. 13. Wifibot deployment with an obstacle

Figure 14 shows results regarding multiple PoI coverage as presented in Section 6. In this experimentation we used 8 Wifibots with a range of $15m$ and two PoIs at $[25, 45]$ and $[45, 25]$ with an intermediate point at $[23, 23]$.



(a) 0s (b) 60s (c) 100s (d) 148s

Fig. 14. Wifibot deployment with multiple PoIs at $[25, 45]$ and $[45, 25]$ with an intermediate point at $[23, 23]$.

8 Conclusion

We present an algorithm for Point of Interest (PoI) coverage with mobile wireless sensors. In our algorithm, the sensors must cover the PoI while maintaining the connectivity with a fixed base station. The algorithm is distributed, needs only local information at each sensor, does not require synchronization and is divided into three parts. In the first part, the sensor computes its direction. In the second part, the distance that has to be covered by the sensor and its speed is computed. The third part is devoted to sensor's motion. Unlike other algorithms described in the literature, our algorithm maintains the connectivity all along the deployment procedure and therefore allows the tracking of mobile PoI. The connectivity maintenance of our algorithm is done by using the properties of the Relative Neighborhood Graph (RNG). Indeed, if a graph G is connected, the Relative Neighborhood Graph extracted from G is also connected. Hence, during their movements, the sensors have only to keep the connection with their RNG neighbors to keep the whole graph connected. Moreover, the computation of the RNG uses only local information.

We evaluate the performances of our algorithm regarding the number of sensors that covers the PoI, the deployment speed, and the energy consumption. We also provide some proofs about the connectivity preservation, the algorithm's termination and the shape of the resulting graph (straight line). We provide some results regarding the coverage of moving PoI and multiple PoIs. Moreover, we implement our algorithm on Wifibots and show that our algorithm can be easily implemented and can work in real conditions by using a simple collision avoidance scheme rule and by alleviating message losses. The next step of this work is to consider the coverage of multiple moving PoIs and to consider the effect of having more than one base station.

References

[1] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *MobiHoc '06: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 131–142, New York, NY, USA, 2006. ACM.

[2] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange. Sensorscope: Out-of-the-box environmental monitoring. *IPSN*, 0:332–343, 2008.

[3] M. A. Batalin and G. S. Sukhatme. Spreading out: A local approach to multi-robot coverage. In *in Proc. of 6th International Symposium on Distributed Autonomous Robotic Systems*, pages 373–382, 2002.

[4] J. Carle and D. Simplot-Ryl. Energy-efficient area monitoring for sensor networks. *Computer*, 37(2):40–46, 2004.

[5] S. Chellappan, W. Gu, X. Bai, D. Xuan, B. Ma, and K. Zhang. Deploying wireless sensor networks under limited mobility constraints. *Mobile Computing, IEEE Transactions on*, 6(10):1142–1157, Oct. 2007.

[6] A. Chen, S. Kumar, and T. H. Lai. Designing localized algorithms for barrier coverage. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 63–74, New York, NY, USA, 2007. ACM.

[7] W. Cheng, M. Li, K. Liu, Y. Liu, X. Li, and X. Liao. Sweep coverage with mobile sensors. In *IPDPS*, pages 1–9, 2008.

[8] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. *Information Processing in Sensor Networks, 2006. IPSN 2006. The Fifth International Conference on*, pages 2–10, 0-0 2006.

[9] X. Li, H. Frey, N. Santoro, and I. Stojmenovic. Focused-coverage by mobile sensor networks. In *Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on*, pages 466 –475, oct. 2009.

[10] V. J. Lumelsky and A. A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.

[11] K. Martinez, R. Ong, and J. Hart. Glacsweb: a sensor network for hostile environments. In *IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks. (IEEE SECON 2004)*, pages 81–87, Oct. 2004.

[12] S. Poduri, S. Pattem, B. Krishnamachari, and G. S. Sukhatme. Using local geometry for tunable topology control in sensor networks. *IEEE Transactions on Mobile Computing*, 8(2):218–230, 2009.

[13] S. Poduri and G. Sukhatme. Constrained coverage for mobile sensor networks. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, 1:165–171 Vol.1, April-1 May 2004.

[14] D. Pompili, T. Melodia, and I. F. Akyildiz. Deployment analysis in underwater acoustic wireless sensor networks. In *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, pages 48–55, New York, NY, USA, 2006. ACM.

[15] C. Shen, W. Cheng, X. Liao, and S. Peng. Barrier coverage with mobile sensors. In *ISPAN '08: Proceedings of the The International Symposium on Parallel Architectures, Algorithms, and Networks*, pages 99–104, Washington, DC, USA, 2008. IEEE Computer Society.

[16] D. Simplot-Ryl, I. Stojmenovic, and J. Wu. Energy-efficient backbone construction, broadcasting, and area coverage in sensor networks. *Handbook of Sensor Networks*, pages 43–380343–380, 2005.

[17] 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*, 8:836–848, 2009.

[18] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4):261 – 268, 1980.

[19] B. Wang, H. B. Lim, and D. Ma. A survey of movement strategies for improving network coverage in wireless sensor networks. *Computer Communications*, 32(13-14):1427 – 1436, 2009.

[20] Y.-C. Wang and C.-C. Hu. Efficient placement and dispatch of sensors in a wireless sensor network. *IEEE Transactions on Mobile Computing*, 7(2):262–274, 2008. Senior Member-Tseng, Yu-Chee.

[21] P. Winter. Steiner problem in networks: a survey. *Netw.*, 17(2):129–167, 1987.

[22] M. Younis and K. Akkaya. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Networks*, 6(4):621 – 655, 2008.