Connectivity Preservation and Coverage Schemes for Wireless Sensor Networks

Tahiry Razafindralambo, David Simplot-Ryl

Connectivity Preservation and Coverage Schemes for Wireless Sensor Networks

Tahiry Razafindralambo, David Simplot-Ryl

Abstract—In this paper, we consider the self-deployment of wireless sensor networks. We present a mechanism which allows to preserve network connectivity during the deployment of mobile wireless sensors. Our algorithm is localized and is based on a subset of neighbors for motion decision. Our algorithm maintains a connected topology regardless of the direction chosen by each sensor. To preserve connectivity, the distance covered by the mobile nodes is constrained by the connectivity of the node to its neighbors in a connected subgraph like the relative neighborhood graph (RNG). We show the connectivity preservation property of our algorithm through analysis and present some simulation results on different deployment schemes such as full coverage, point of interest coverage or barrier coverage.

Index Terms—Wireless Sensor Networks, Autonomous systems, Coverage, Deployment, Connectivity

1 Introduction

Monitoring areas in wireless sensor networks has received a lot of attention these last years. In this context, the optimization of sensor placement is a difficult problem, even for deterministic and static deployments. Although deterministic deployments can provide optimal solutions, they are not always feasible since they require precise knowledge of the monitored area. Controlled deployments, or on-demand deployments are only feasible when the sensors' positions are available, and when sensors have motion capabilities. However, the main advantage of on-demand deployment is the possibility to obtain particular topologies which can reduce energy consumption, optimize routing scheme or flooding, etc [1]. Moreover, different coverage schemes such as full coverage (sensors try to cover the whole area of interest), barrier coverage [2] (sensors try to form a barrier for intrusion detection) or sweep coverage [3][4][5] (sensors try to cover only some specific points) can be obtained with on-demand deployment.

During the on-demand deployment, the evolving graph can have different properties. Controlling the dynamic graph of mobile sensors is a fundamental issue in order to keep wireless sensor application properties even during the deployment. The most important property of the dynamic graph is connectivity. Indeed, in the area of mobile communications and especially for wireless sensor networks, the main challenge is to maintain connectivity from any sensors to the sink (or base station) if we consider a N to 1 communication paradigm. This connectivity allows an external entity to update the behavior of all the sensors and to gather information from them by using multi-hop communication. For example, the sensor deployment may start with a full coverage requirements and may change to a point of interest coverage.

In this paper we consider the problem of deploying and controlling a fleet of mobile sensors for different coverage schemes[1] while keeping the graph of mobile sensors connected at each step of the deployment. Our first assumption is: all sensors are within communication range of each other (complete graph) at the beginning of the deployment. From this initial configuration, a sensor node may choose any direction to expand the network. This direction is linked to a deployment requirement. In order to avoid disconnection, the node has to maintain its connections to its neighbors that are parts of the Reletive Neighborhood Graph (RNG) reduction (see Section 3.1). After the first expanding process, some connections between the sensor and its previous neighbors are lost and deployment process continues in the same way. Analytical and simulation results show that our deployment algorithm maintains the graph's connectivity all along the deployment procedure, and that different coverage schemes can be provided by our algorithm.

Our algorithm is localized *i.e.*, every decision taken by sensors is based on local neighborhood information only, asynchronous and simple enough to take into account obstacles, or specific fields constraints. Moreover, since the directions and the movements of a given node are only constrained by the connections to its RNG neighbors, the node's direction can be governed by any requirement which allows our algorithm to adapt to different coverage schemes.

Summary and main results.

- We provide a *localized* deployment algorithm for mobile sensor networks.
- Our algorithm *preserves connectivity* all along the deployment procedure. We use the Relative Neighborhood Graph [6] to preserve connectivity and prove that the network is connected at each step of the deployment.
- We split our algorithm into 2 major parts. The *connectivity preservation* part and the *deployment* part. This distinction allows us to provide different deployment schemes while keeping connectivity. In this paper, results on area *coverage maximization*,*Point of Interests* (POI) coverage and *barrier coverage* are provided.

The rest of this paper is organized as follow. In Section 2,

- *T. Razafindralambo is with the INRIA Lille - Nord Europe center, France. Tahiry.Razafindralambo@inria.fr*
- *D. Simplot-Ryl is with the University of Lille 1, France. David.Simplot-Ryl@inria.fr*

1. The deployment of mobile sensors can be either used to maximize area coverage, to provide a point of interest coverage or to provide a barrier coverage

we give a state of the art on deployment of mobile sensors. In Section 3, we give an overview of assumptions and notations used in this paper. Section 4 details our deployment algorithm. Section 5 provides an analysis of this algorithm and simulation results are given in Sections 7 and 8. The conclusion and some possible future works are provided in Section 9.

2 Related Work

Papers about deployment and self-deployment of wireless sensor networks are reviewed in this section and we shortly extend this state of the art to mobile robots deployment. As our main focus is connectivity preservation for different deployment schemes, this section cites papers that consider connectivity during the deployment and briefly review some different coverage schemes. Moreover, we mainly focus on mobile sensors deployments but the interested readers can refer to [7], [8] for static random deployment strategies, to [9], [10], [11] for off-line computation of sensors placement and to [12], [13] or [14] for a complete surveys.

2.1 Connectivity

The local dispersion of multiple mobile sensors (or mobile robots which embed sensors) was first developed in [15] to achieve a better coverage of the whole sensing field. In [15], Batalin and Sukhatme argue that the local dispersion is the basis for increasing global coverage. In this approach, sensors are mutually repelled by each other within their communication range. Their approach is inspired by the diffusive motion of fluid particles. However, connectivity is not considered as essential during the deployment and the authors mainly focus on increasing the covered area.

The artificial potential field concept was first proposed by Khatib in [16]. Potential field theory was used to compute path planning algorithms for mobile robots in [17], [18]. In [19], the coverage maximization strategy is constraint by the fact that each node must have at least K neighbors. A repelling force is computed to increase the coverage of the network while an attracting force tries to maintain the node degree. However, this simple scheme cannot guarantee connectivity during the deployment since having K neighbors does not guarantee global connectivity.

In [20], the authors also use potential field concepts for deploying mobile robots. The proposed deployment tries to preserve the graph connectivity by checking at any time whether the graph is connected. To do so, a message is regularly flooded in the network by a central entity. Note that this regular flooding is resource consuming in wireless sensor networks. If a mobile sensor does not receive this message, it assumes that it is disconnected and moves toward its last position or some predefined intermediate destinations. In [20], the network is not connected all along the deployment procedure. This approach for maintaining connectivity during deployment is very simple but induces a very high message overhead, requires a central entity and a efficient flooding algorithm.

The closest work to ours are proposed in [21] and [22]. The authors of [21] use local geometry combined with potential field theory to maximize the area coverage of mobile robots. They use a Neighbor-Every-Theta (NET) graph to constrain the node movements. The authors apply the same forces as described in [19]. By using a combination of mutually opposing forces, each node maximizes its coverage while maintaining the NET condition of having at least one neighbor in every θ sector. It is worth noting that in [21] for a specific $\theta \leq \frac{2\pi}{3}$ the graph can embed an RNG. However, the distributed algorithm deployment proposed by the authors cannot guarantee connectivity since a node in a sector that provides the connectivity may not be chosen as a NET neighbor. Unlike [21] and the work presented in this paper, the authors of [22] use spanning tree to maintain connectivity. They introduce a novel state dependent graph, called the double-integrator disk graph to add a connectivity constraint to the agent's movements. The proposed scheme maintains connectivity but needs high message overhead for the spanning tree construction. Yet, the resulting controls may be too restrictive to allow different coverage schemes. Moreover, the connectivity constraint computation relies on linearized Jacobi methods which is resource (processor, memory) consuming to be run on sensors.

The work proposed in [23] addresses connectivity in the context of formation control. The authors present an algorithm to perform a graph rearrangements. With this rearrangements, different coverage schemes can be obtained. However, the starting and the target configuration must be known, which is not always feasible.

In algebraic graph theory, the properties of a graph are characterized through the study of the eigenvalues of the Laplacian. In [24], a decentralized estimation of the algebraic connectivity can be found. A technique based on a decentralized power iteration algorithm is given to control a fleet of mobile sensors. However, this technique cannot guarantee connectivity all along the deployment procedure.

In [25] the authors focus on network connectivity and do not properly consider coverage. They translate the connectivity conditions into differentiable constraints on individual node motions. In this deployment, a central entity has to be aware of the connections between all robots, computes the possible movements of each robots and sends movements constraints to each robots using a feedback procedure. This approach guarantees connectivity during the deployment but is not scalable since it need a central entity.

2.2 Coverage schemes

The coverage requirement is the primary aim that describes how the sensors have to be deploy over the field. Even if some ways of moving are strongly related to the coverage requirement, it is important to note that movement and coverage are independent. From our point of view, these two aspects must be decorrelated in order to have simple deployment algorithms. In this section, we recall the previous cited papers regarding coverage schemes. The coverage requirements can be divided into three categories:

- In the full coverage problem, sensors have to maximize the covered area. The work proposed in [20] and [21] uses virtual force based movement to increase the covered area. The main difference of these two works is the connectivity consideration. In [20], a connectivity checking procedure is implemented. In [21], the authors use the Neighbor-Every-Theta (NET) graph properties.
- In barrier coverage problem, sensors must form a barrier that detects any events crossing the barrier. In the work proposed in [26], the 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. In [26], when the number of sensors is sufficiently large, connectivity can be provided at the end of the deployment. However, [26] does not guarantee connectivity all along the deployment procedure.
- In the PoI coverage, only some specific points of the sensor field need monitoring. Surprisingly, very few works consider the problem of PoI coverage. In [3], the authors propose an algorithm to periodically monitor some specific points. Unlike the work presented in this paper, results from [3] do not consider connectivity issue.

In our work, we use the property of the Relative Neighborhood Graph to maintain connectivity. To the best of our knowledge, our work is the first one that can simply provide any coverage scheme while preserving connectivity at each step of the deployment because we separate connectivity preservation and coverage scheme. Moreover, unlike most of the work cited above, our work is robust against message losses during deployment.

3 Preliminaries and Basic Idea of Our Approach

Notations and assumptions are introduced in the following section. This section also outlines the basic idea of our approach and motivates the choice of the RNG.

3.1 Relative Neighborhood Graph

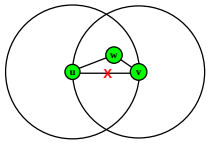


Fig. 1. Example of RNG edge removal. To obtain a RNG reduction all triangles from the initial graph are suppressed by removing the longest edge of each triangle.

The Relative Neighborhood Graph (RNG) [6] is a graph reduction method. Given an initial graph G, the RNG 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 a RNG from an initial graph G, an edge that connects two sensors is removed if there exists another node that is at a lower distance from both sensors. Figure 1 shows an example of edge removal, where edge between sensors u and v is removed since there exists a sensors w that is closer to both u and v.

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

3.2 Notations and assumptions

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 defined by $E = \{(u, v) \in V^2 \mid u \neq v \wedge d(u, v) \leq R\}$, where $d(.)$ is the Euclidean distance between nodes u and v and R is the communication range.

The graph G is known as the Unit Disk Graph (UDG) model [27] and is commonly used to model sensor networks. Note that if we use the UDG in the description of our algorithm and in simulation, the protocol also works with a realistic graph, *i.e.* graph observed by a real node. The only condition is to be able to compute, in a localized way, the subgraph of nodes we want to maintain.

Definition 2: The set of 1-hop neighbors of node u is denoted by $N(u)$ and is defined by $N(u) = \{v \in V \mid (v, u) \in E\}$.

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

Assumption 1: We assume that each sensor u is aware of its position denoted by $(x(u), y(u))$. This position can be provided by any internal mechanism or external entity such as GPS.

Definition 4: $RNG(u)$ is the set of neighbors of node u in $RNG(G)$ graph. $RNG(u) = \{v \in N(u) \mid (u, v) \in E^{rng}\}$. We denote by $|RNG(u)|$ the number of nodes in $RNG(u)$.

Definition 5: $RNG^+(u)$ (resp. $RNG^-(u)$) represents the furthest (resp. the closest) node from u in $RNG(u)$ ($RNG^+(u)$ such that $d(u, RNG^+(u)) = max_{v \in RNG(u)} d(u, v)$, $RNG^-(u)$ such that $d(u, RNG^-(u)) = min_{v \in RNG(u)} d(u, v)$). We note $d^+(u) = d(RNG^+(u), u)$ and $d^-(u) = d(RNG^-(u), u)$. Note that more than one node can be at the same furthest distance from node u. In this case, one of these nodes is chosen randomly.

Definition 6: $\nu \in [0, \nu_{max}]$ is the speed of a given node.

Assumption 2: Each node periodically computes its next position based on its neighborhood every δ. We also assume that each node regularly sends a HELLO message containing its ID and position with a frequency higher than $\delta/2$ for computation accuracy.

3.3 Basic idea

Our deployment algorithm is localized and is based on potential field theory. Each node is considered as a particle and its movements are governed by the interactions with some of its neighboring nodes. The subset of neighbors with which a node u interacts together with the direction u has to choose only rely on the kind of coverage required while the distance to be covered by u is constraint by the preservation of its connection to nodes in $RNG(u)$. The RNG [6] is a suitable solution since its computation only requires local information. Moreover, the use of the Euclidean distance for computing the RNG strongly reduces the mean degree of the graph. The mean degree of an RNG is ~ 3 [6]. In addition, one of the most interesting feature of an RNG is that, while removing some edges from the initial graph, the graph $RNG(G) \subseteq G$ also preserves the connectivity, provided that the initial graph G is connected.

In our algorithm, any graph that reduces the neighborhood size while keeping connectivity can be used such as Gabriel Graph (GG) or Spanning Trees to maintain connectivity. The use GG or spanning trees may change the deployment results compared to RNG since Spanning tree in include in RNG and RNG is included in GG. The main difference would be the

average degree of sensors at the end of the deployment that impacts the final deployment results especially for the full coverage scheme. We can also try to increase the connectivity by using graphs such as k-RNG, k-GG, etc., [28]. Note here that we choose the RNG because of its simplicity regarding construction that can be done in a localized way and since most of the cited graphs are included in RNG.

Our algorithm is broken down into two distinct parts: the **direction computation scheme** and the **connectivity preservation scheme**. This partition allows providing different deployment schemes while preserving connectivity. Therefore with only some simple modifications on the deployment scheme, our algorithm fits different requirements of mobile sensor deployment applications.

4 Protocol description

The protocol is split into two parts formally described in Algorithm 1. Part 1 provides the direction computation scheme while part 2 ensures the connectivity preservation.

Part 1: Direction computation The direction to be chosen by node u depends on the deployment requirements. We will go back on the details on this part in Section 7 for each of the following schemes: maximising coverage, point of interest coverage of barrier coverage. Part 1 provides a normalized vector $\overrightarrow{\Delta}$ giving the direction of node u. Different algorithms for Part 1 are described in the simulation section.

Part 2: Connectivity preservation. This part is divided in several steps: (a) Speed computation, (b) Distance computation and (c) Destination and movement.

Part 2.a: Speed computation. Based on the information gathered from its neighborhood, a node u computes its movement speed. This speed has to be as fast as possible (to allow a fast deployment of nodes) while preserving connectivity. This maximum speed is thus divided by two to consider the worst case movement of $RNG^+(u)$. It can not be null to still allow some small movements of node u. This also allows nodes that are at distance R to move toward each other. At the end of Part 2.a, node u knows its movement speed ν.

Algorithm 1 MSD (Mobile Sensor Deployment) protocol

Part 1 — Direction computation on node u:
1: return $\overrightarrow{\Delta}$; // unit vector that gives the direction of the node

Part 2.a — Speed computation on node u:
1: $\nu = \frac{R-d^+(u)}{\delta \times 2}$;
2: $\nu = \min(\nu, \nu_{max})$;

Part 2.b — Distance computation for node u:
1: $d_{max} = \max(\epsilon, R - d^+(u))$;
2: $d_{opt} = \{d \in [0, d_{max}] \mid \forall v \in RNG(u), d(u_{new}, v) + \nu_{oth}(v) \times \delta < R\}$
3: where $\nu_{oth}(v) = \frac{R-d(u,v)}{\delta \times 2}$
4: and u_{new} is the new position of node u based on:
5: -direction $\overrightarrow{\Delta}$ from Part 1.
6: -speed ν from Part 2.a.
7: -distance d.

Part 2.c — Node u destination and movement:
1: move to u_{new} using :
2: -direction $\overrightarrow{\Delta}$ from Part 1.
3: -speed ν from Part 2.a.
4: -distance d_{opt} from Part 2.b.
5: -Take field border into account

Part 2.b: Distance computation In this part, node u computes the distance it has to cover before running Algorithm 1 again. This distance is chosen in a given range $[\epsilon, d_{max}]$ (Line 1 of Part 2.b) where ϵ is a parameter of our algorithm and is used to allow some small movements when needed. To do so, it computes the maximal speed of its RNG neighbors ν_{oth} in such a way that even in the worst case, *i.e.* if its further RNG neighbor $RNG^+(u)$ goes in the opposite direction, they both remain connected. This means that in the worst case, they can both travel a distance equal to $d = \frac{R-d^+(u)}{2}$ before being disconnected. Indeed, $RNG^+(u)$ can not chose a higher distance than u since, as links are bidirectional $d^+(RNG^+(u)) \geq d^+(u)$. This distance has to be covered between two checking, *i.e.* in δ. Based on these information, node u computes the maximum possible distance d_{opt} it can travel given that in its new position, it still remains connected to its $RNG(u)$ nodes.

Part 2.c: Destination and movement In this part, the path planning of the sensor is considered. Given a direction, a speed and a distance, 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 [29]. This allows unmodified motion planning algorithms to control the trajectories of each mobile sensors.

It is important to note here that Part 1 of Algorithm 1 is completely independent from the other parts. This is an important property since the direction of nodes can be easily modified to fit some other requirements as we will see in more details in Section 7.

5 Algorithm properties

In this section we demonstrate that at each step of our algorithm, the graph is connected. We assume that each link is symmetric and sensors are equipped with omni-directional antenna. We also assume that the transmission power of each node is fixed and thus that the graph can be modelled as a UDG (unit disk graph). These assumptions are taken for the sake of tractabilty and are discussed later.

In the sequel, nodes are running the same algorithm (Algorithm 1) with the same parameters. We assume a discretized time indexed by $i \in \mathbb{N}$. At $t = 0$, all nodes are connected. Nodes are uniquely identified and are called $n_i, i \in [0, ..., N-1]$ where N is the number of nodes. Note that in the sequel, node positions are identified by n_i^j where i is the node identifier and j the time index. We use n_0 and n_0^j to identify the node and its position.

Lemma 1: If at time $t = T$ the graph is connected, and all nodes are synchronized and run Algorithm 1 in a sequential way that is in a given interval $[t, t+1[$, only one node runs the algorithm and reaches its new position during this interval; then $\forall i > T$ the resulting graph at time $t = i$ is connected.

Proof: The proof of this theorem only relies on the distance covered by the node. This distance is computed in Part 2.b of Algorithm 1. We know that at $t = 0$, the network is connected (initial condition). At $t = 0$, n_0^0 runs Algorithm 1. The maximum distance covered by n_0^0 is $d_{opt} = \{d \in [0, d_{max}] \mid \forall v \in RNG(n_0^0), d(n_0^1, v) + \nu_{oth}(v) \times \delta < R\}$. At time $t = 1$ the new position of n_0^0 is n_0^1. The condition on d_{opt}, avoids n_0 being disconnected from its $RNG(n_0^0)$ neighbors. Indeed, at time $t = 1$ $\forall u \in RNG(n_0^0), d(n_0^1, v) < R$ (note that $\nu \times \delta > 0$ and $\nu_{oth}(v) \times \delta \geq 0$) since the nodes run algorithm in a sequential way, during the time $[t, t+1[$ no other node is moving. Thus n_0^1 is still connected to its $RNG(n_0^0)$. Based on the properties of the RNG, the graph at time $t = 1$ is therefore still connected. We know that if the graph is connected at $t = 0$, it is also

connected at $t = 1$. We can state that if at any $t = i, i > 0$ the graph is connected, at $t = i + 1$ the graph is still connected and we can also state that if at a given time $t = T$ the graph is connected at any time $t > T$ the graph is still connected.

□

Theorem 2: If at time $t = T$ the graph is connected, $\forall t = i, i > T$ the resulting graph at time $t = i$ is connected.

Proof: Lemma 1 shows that in a synchronized environment, if the initial graph is connected, the graph remains connected. In an asynchronous environment, nodes can run Algorithm 1 at any time. Let n_i^T and n_j^T be two nodes and n_i^T and n_j^T are connected a time $t = T$. $n_i^T \in RNG(n_j^T)$, $n_j^T \in RNG(n_i^T)$ and $d(n_i^T, n_j^T) = d^+(n_i^T)$.

CASE 1: Let us assume that both nodes run Algorithm1 at the same time, and that both nodes are moving to the opposite direction of each other. The maximum distance covered by node n_j^T depends on its speed and the sampling time δ. Since $d(n_i^T, n_j^T) \leq d^+(n_j^T)$ the maximum speed of node n_j^T computed in Line 1, Part 2.a of Algorithm 1 is $\nu_j = \frac{R-d^+(n_j^T)}{\delta \times 2} \leq \frac{R-d^+(n_i^T)}{\delta \times 2} = \nu_i$ (the speed of node n_i^T). Therefore, the maximum distance covered by node n_j^T during δ is at most $\nu_i \times \delta$. Note that the position of node n_i^T is $n_i^{T+\delta}$ after its movement. The computation of d_{opt} for node n_i^T in Line 3, Part 2.b of Algorithm 1 includes the worst case movement of the n_j^T which still maintains connection to all its $RNG(n_i^T)$ nodes after its movement.

CASE 2: Let us now assume that $d(n_i^T, n_j^T) = R = d^+(n_i^T)$. In that case, $\nu_i = \nu_j = \epsilon$ and $\nu_{oth} = 0$ for both nodes. If both nodes are moving to the opposite direction, condition in Line 3, Part 2.b of Algorithm 1 is not satisfied for a value of $d > 0$. If one of the nodes, let say n_i, is moving toward the other node and (n_j) moves to the opposite direction of n_i, $d_{opt} = 0$ for node n_j, for the same reason, and $d_{opt} \geq 0$ for node n_i since condition in Line 3, Part 1 of Algorithm 1 can be verified.

If the connection to the furthest RNG neighbor is maintained, the connection to close RNG neighbors is also maintained. For that, let us assume node $n_k^T \in RNG(n_i^T)$ with $d(n_i^T, n_k^T) \leq d(n_i^T, n_j^T)$, the speed of node n_k^T is at most $\nu_k = \frac{R-d^+(n_k^T)}{\delta \times 2} \leq \frac{R-d(n_i^T, n_k^T)}{\delta \times 2}$. Therefore, the maximum distance covered by node n_k^T is at most $\nu_k \times \delta$. The worst case movement for the two nodes n_i^T and n_k^T leads to a distance $d(n_i^{T+\delta}, n_k^{T+\delta}) = d(n_i^T, n_k^T) + \nu_k \times \delta + \nu_i \times \delta \leq R$ by replacing ν_k and ν_i by their value (or upper bounds) and since $d(n_i^T, n_k^T) \leq d(n_i^T, n_j^T)$. Therefore, the movement of node n_i^T does not disconnect it from its RNG neighbors. Network connectivity is thus kept.

□

6 Discussion and Implementation Issues

Section 4 provides the basis of our algorithm. However, we do not consider all possible special cases that may arise during a real deployment. This section discusses these issues and the possible solutions.

As stated earlier, connectivity is very important in wireless sensor networks since data reporting from the sensor to a sink may be interrupted due to the lack of connectivity, to message losses due to collision at the Medium Access Control layer, to the instability of wireless channels or to physical obstacles can occur. These losses may happen during deployment and then modify the behaviour of the deployment.

A simple way to cope with this issue is to use historical evolution of the RNG neighbors. Let us assume that nodes u and v are neighbors and that $v \in RNG(u)$ at time $t = 0$. Let, for example, node u be repelled by node v. At $t = 1$ (after u's movement), the set $RNG(u)$ can change. However, node v is still a neighbor of u since our algorithm preserves connectivity with the RNG neighbors. If node v does not appear to be part of $N(u^1)$ (set of u's neighbors at $t = 1$) this means that messages from node v were lost. In this case, node u stops moving until it gets a message from node v.

It is important to note that building an RNG needs symmetrical links. Indeed, if node $u \in RNG(v)$ but $v \notin RNG(u)$, node v's movement may disconnect the graph. In order to keep symmetrical links, we use the same neighborhood table exchanges. If node u receives a neighbor discovery message from node v but node u id is not in node v message, this means that node v did not receive node u message and thus asymmetry is identified. In case of asymmetry, nodes wait until symmetrical neighborhood table is build. Note here that after some timeout, for example $3 \times (\delta/2)$, if a node does not receive a message from a neighbor, this means that the neighbors has failed and a procedure for partition detection and reconnection can be used such as the ones described in [30] and [31].

Here, when message losses are considered, it is not possible to guarantee connectivity because two neighboring nodes may never receive message one from the other one and thus may not be aware of their respective existence. However, the effect of message losses can be alleviated by increasing the number of HELLO messages sent during δ. It is also worth noting that deployment performance regarding speed and coverage requirements are reduced due to message loss.

7 Simulation Results

Simulation results are divided two parts. In the first part (Section 7.1), we present the simulation results of different deployment schemes such as maximizing area coverage, Point of Interest (POI) coverage and barrier coverage. These different schemes are related to the implementation of Part 1 of Algorithm 1. In the second part (Section 7.2), we consider the impact of message loss and show how it can affect our algorithm. The simulation were performed using WSNet[2]. The simulation parameters are summarized in Table 1. Note that we do not present simulation results with obstacles due to space limitations. However, in our implementation we use a simple obstacle avoidance scheme. Indeed, a node stops moving when it encounters an obstacle, and considers the obstacle when computing its next direction. In the simulation, we set the communication range to be equal to the sensing range but this assumption can be easily modified without affecting the behaviour of the deployment. It is worth noting that since we focus both on connectivity preservation during the deployment and different deployment schemes, comparisons with work from the literature are hard to provide.

7.1 Different deployment schemes

7.1.1 Maximizing area coverage

In order to maximize area coverage, we use a repelling force between RNG neighbors that allows coverage expansion. Part 1 of Algorithm 1 is described in Algorithm 2. Here, node u goes further from all $RNG(u)$ nodes except those at distance R by using the resulting vector $\overrightarrow{\Delta}$. Moreover, we use a weighted sum to compute the resulting vector and increase the impact

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

Field size	$100m \times 100m$
Sensing range	$10m$
Max Communication range	$10m$
ν_{max}	$10ms^{-1}$
δ	$5s$
ϵ	0.1
Simulation time	$5000s$

TABLE 1
Summary of the simulation parameters.

of closer nodes in the direction of node u (Line 2 of Part 1). In Lines 3 and 4 of Part 1 we only compute the normalized direction. Here, P (Line 5) is the destination point of node u and can be used to reduce the distance computed in Part 2.b of Algorithm 1.

Algorithm 2 Maximal coverage requirements

Part 1 — Direction computation on node u:

1: $\overrightarrow{\Delta}(x(\Delta), y(\Delta))$ is the direction vector of u
2: $\overrightarrow{\Delta} = \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} (R - d(u,v)) \times ||\overrightarrow{vu}||$
3: $x(\Delta) = \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} (x(u) - x(v)) \times (R - d(u,v))/R;$
4: $y(\Delta) = \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} (y(u) - y(v)) \times (R - d(u,v))/R;$
5: $P(x(u) + x(\Delta), y(u) + y(\Delta));$
6: $\overrightarrow{\Delta} = \frac{\overrightarrow{\Delta}}{||\overrightarrow{\Delta}||};$

Figure 2 shows the evolution of node position along time. We can see from Figure 2 that, as proved by Theorem 2, the graph is connected. We can also note from this figure that graph is expanding, remains connected and that the algorithm reaches a stability point (means that no node is moving anymore).

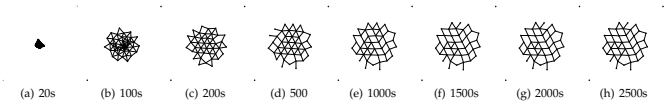
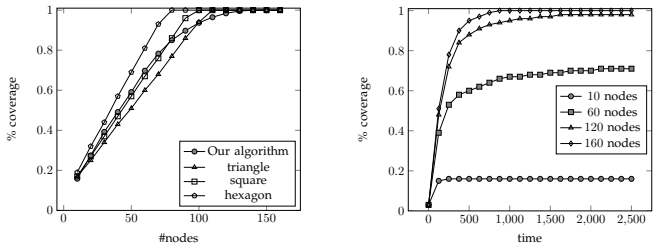


Fig. 2. Evolution of nodes position over time. In this simulation, there is 40 nodes with a range of 10 on a square of 100×100.

We also evaluate the coverage performance, of this deployment scheme. We use a discrete way to evaluate the coverage. Area is divided into a grid and we compute the number of points that are covered over the total number of points in the grid. We can see from Figure 3(a) that the resulting coverage of our protocol is very close to a coverage provided by a square pattern until 90 nodes since the area cannot be fully covered. When the number of nodes is enough to cover the whole area, our algorithm, because of its dynamic behaviour suffers from border effects which may prematurely stop node movements. In Figure 3(b) we can see the coverage results depending on time. This result shows the expansion property of our algorithm. We can see that the curves are increasing. Note here that this second figure is a sample from one simulation which explains the coverage difference with Figure 3(a) that is the average of many simulations.

7.1.2 Point of Interest coverage (POI)

Since whole area coverage may not be necessary, the previous algorithms can be modified to monitor some point of interest



(a) Coverage vs. Nodes (b) Coverage vs. Time

Fig. 3. Coverage results. In Fig. 3(a) we compare our algorithm to off-line deployment following different regular patterns (triangle, square and hexagon). In Fig. 3(b) we plot the coverage evolution vs. Time for different node numbers.

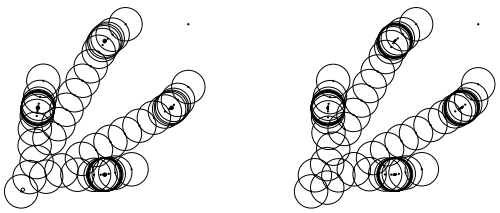
in the field [3]. In this section we present the modifications applied to the Part 1 of Algorithm 1 to provide POI coverage. The modification is presented in Algorithm 3. In this algorithm, each node moves toward the center of gravity B of its RNG neighbors and I the point of interest. B is the destination point of node u and can be used to reduce the distance computed in Part 2.b of Algorithm 1. This simple modification can strongly change the shape of the resulting graph since I becomes an attractor. Lines 2 and 3 of Algorithm 3 show the computation of B. Because Part 1 is independent from Part 2.b, connectivity is preserved.

In the simulation results presented in this section we define four POIs in a field of 100×100 at coordinate $(10, 50)$, $(50, 10)$, $(90, 50)$ and $(50, 90)$. We also define a node at position $(0, 0)$ as a base station. The starting point of all nodes is in the communication range of the base station. All the nodes have a predefined point of interest randomly chosen at the beginning of the simulation. The base station node is only defined to fit deployment where POIs have to be covered and connectivity to a fixed based station need to be kept.

Algorithm 3 Point of Interest coverage

Part 1 — Direction computation on node u:

1: $B(x(B), y(B))$ center of gravity of $RNG(u)$
2: $x(B) = \frac{1}{|RNG(u)|+1} \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} x(v) + x(I)$
3: $y(B) = \frac{1}{|RNG(u)|+1} \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} y(v) + y(I)$
4: $\overrightarrow{\Delta} = \frac{\overrightarrow{uB}}{||\overrightarrow{uB}||};$



(a) 60 nodes (b) 80 nodes

Fig. 4. POI deployments using the center of gravity as the direction. This figure plots the resulting graph after 5000s for different number of nodes with a range of 10 on a square of 100×100

Figure 4 plots the resulting graphs for 60 and 80 nodes. Since connectivity is kept, we do not plot edges, instead we plot the sensing range of each node. Here, the sensing range is equal to the communication range. We can see from these figures that the four points of interest are covered by more than 6 sensors (for 60 nodes). Figure 5 shows that the number of covering nodes is increasing with the total number of nodes. We can see in Figure 5 that the coverage of a given POI depends on its distance from the starting point of the deployment (here $(0, 0)$). This is due to the fact that $1/4$ of nodes are affected to a given POI and for example $5 = 20/4$ nodes are not enough to reach the point $(50, 90)$. We can also see that the number of covering nodes is linearly increasing for each POI. This shows that the number of nodes connecting the base station and a POI (if possible) is independent from the total number of nodes. This property is confirmed by the example of resulting POI coverage in Figure 4.

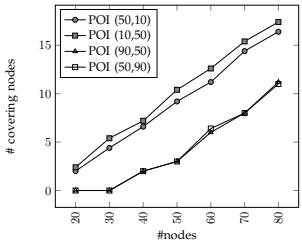


Fig. 5. Results for POI coverage. This figure plots the number of nodes covering a given POI depending on the total number of nodes in the network.

7.1.3 Barrier coverage

Barrier coverage [2], [32] is an important way of covering area especially when considering intrusion detection. Since in Algorithm 1 connectivity is independent from the direction computation, we can easily modify the direction while keeping the properties of our algorithm such as connectivity. Unlike in [2] our deployment is focused on maintaining the graph connectivity while providing barrier coverage. In this section we present the modifications applied to Part 1 of Algorithm 1 in order to provide self deployment of sensor network for barrier coverage. In our algorithm we consider the barrier coverage as a special case of POI coverage. In barrier coverage, POI are used to define a barrier between the starting points and the POI. Unlike in POI coverage, our aim is not to cover the POI, instead, we want the nodes to be regularly spread out between the starting point and the POI. The modification of Part 1 (in Algorithm 1) is presented in Algorithm 4. Where, like in Algorithm 3, I is the POI, and B the center of gravity of the RNG neighbors. In this algorithm, if a node, say v has already covered the POI, the POI is removed and the other nodes behave like without an attractor.

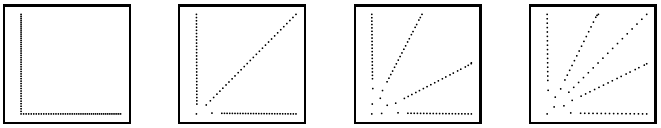
Node v becomes fixed, the other nodes deploy themselves to be above the center of gravity of their RNG neighbors. This behaviour of our algorithm provides a dense barrier since sensors will form a line between POIs and the base station. We obtain a straight line deployment since RNG reduction removes the triangles of the original graph (see Figure 1 in Section 3.1) and sensors move toward the center of gravity of its RNG neighbors (in Figure 1, node w will move to the center of gravity of u and v). Figure 6 plots some examples of deployments where 2 to 5 POIs are defined. In this simulation,

Algorithm 4 Barrier coverage

Part 1 — Direction computation on node u:

1: **if** $(d(u, I) == 0)$
2: Stop node u motion;
3: remove POI: I;
4: **end if**
5: $B(x(B), y(B))$ center of gravity of $RNG(u)$
6: $x(B) = \frac{1}{|RNG(u)|+1} \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} x(v) + x(I)$
7: $y(B) = \frac{1}{|RNG(u)|+1} \sum_{\substack{v \in RNG(u) \\ d(u,v)<R}} y(v) + y(I)$
8: $\vec{\Delta} = \frac{\overrightarrow{uB}}{||\overrightarrow{uB}||}$;

$1/5$ of the number of nodes are assigned to each POIs to form the barrier. We can see from this figure that the sensors form



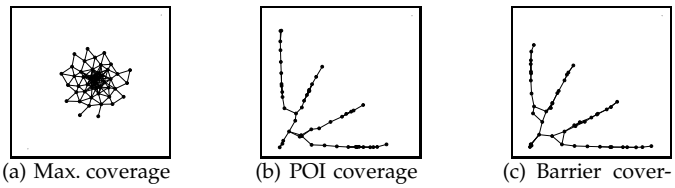
(a) 2 POIs (b) 3 POIs (c) 4 POIs (d) 5 POIs

Fig. 6. Example of deployments for barrier coverage with 5 POIs and a fixed base station at coordinate (0,0). This figure plots the resulting graph after 5000s for different POI location and number with a range of 10 on a square of 100×100 with 90 nodes.

a straight line between the base station and each POI. Unlike for POI coverage in the previous section, all nodes are located between the base station and the POI and thus result in a dense barrier formation when the number of POIs is small (for the same number of nodes).

7.2 Effect of message loss

In the simulation presented above, we do not consider message losses. However, in real deployment, this assumption is not realistic and can strongly affect the deployment results. In this section we present the coverage results regarding the three deployment schemes presented in Section 7.1 when message losses are considered. We consider two different kinds of message losses. 1) Probabilistic. We consider that each message as a probability π to be lost. 2) Log-normal. We consider that message loss is related to the distance between two nodes, following a log-normal shadowing model [33].



(a) Max. coverage (b) POI coverage (c) Barrier coverage

Fig. 7. Resulting graph at the end of the simulation for different coverage schemes with message loss probabilty $\pi = 0.5$. In this simulation there is 50 nodes with a range of 10 on a square of 100×100.

Figure 7 shows the deployment results for the three different schemes considering a simple probabilistic error scheme ($\pi = 0.5$). We can see in these examples that coverage performances

are reduced since for the maximizing coverage scheme, nodes are not fully deployed and POI coverage and barrier coverage are not obtained. These behaviour are mainly due to the fact that when neighbors are not symmetrical or previous RNG nodes disappear, nodes stop moving and wait until the neighborhood table is consistent. Again, it is clear that performance coverage is reduced due to message losses. Moreover, even with the implementation described in Section 7.2, connectivity cannot be guarantee since two nodes may never be aware of each other. Note here that we do not implement the solution provided in [30], [31] for partition detection and reconstruction since we mainly focus on connectivity during the deployment. However, at the end of the deployment, running the algorithms proposed in [30], [31] are convenient solutions. It is worth noting that increasing the simulation time increases the coverage performance. In this section we do not modify the simulation time in order to see the impact of message loss and because we mainly focus on connectivity property.

In order to evaluate the connectivity provided by our algorithm we use a connectivity measure proposed in [34]. This metric called reachability is defined as the fraction of connected node pairs in the network. The reachability of the networks of N nodes is: $\rho = \frac{nb.of connected pairs}{\binom{N}{2}}$. Note here that when $\rho = 1$ the network is connected and when $\rho = 0$ all nodes are isolated. We compute reachability based on the communication range of the nodes derived from an unit disk graph model, thus even if during the simulation no messages can transit over a wireless link, two nodes are considered to be neighbors and should be connected if their distance is lower than the communication range. Figure 8 shows the reachability of each deployment scheme. We can see from this figure that the reachability is close to 1 which means that the network is almost connected.

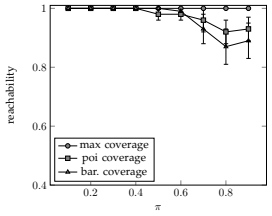


Fig. 8. Probabilistic errors. Reachability for a network of 50 nodes with a range of 10 on a 100×100 square. Reachability is computed at the end of the simulation.

We can note from Figure 8 that for the maximal coverage scheme, the reachability is always $\rho = 1$. This is due to the fact that when the neighborhood of a node is not symmetric of when the node does not have any neighbors, the node does not move which alleviates the effect of message loss. Therefore, the network is always connected. For the POI and the barrier coverage, the reachability is not always 1. This behaviour is due to the fact that for these two deployments when two neighbors are not aware each of the other, they move toward their point of interest and can disconnect the network. Moreover, the deployments presented in Figure 4 and 6 suggest that each node has ~ 2 RNG neighbors. In this case, the probability for a RNG link to be lost and thus for the network to be disconnected is high.

Figure 9 shows the reachability for the same network setting when using the log-normal shadowing model. We can see that the reachability is close to ~ 1 and is close to the performance of probabilistic errors when $\pi = 0.5$. This was expected since

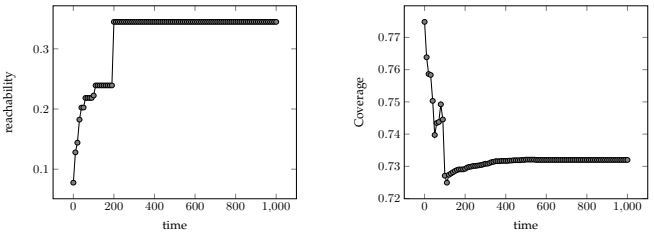
Type	Reach. (conf. interval)
max. coverage	1.00 - [0.00;0.00]
poi coverage	0.99 - [0.02;0.01]
bar. coverage	1.00 - [0.00;0.00]

Fig. 9. Log-Normal errors. Reachability for a network of 50 nodes with a range of 10 on a 100×100 square. Reachability is computed at the end of the simulation.

with the log-normal model the probability to successfully send a message is 0.5 when nodes are at distance R.

8 Special cases

Results related to initial setup of our simulation are presented in this section. We assume that at the beginning of the deployment the graph is not connected. Note that in this section, we only focus on the deployment which tries to maximize the covered area. Figure 10(a) plots an example of the evolution of the reachability when the nodes have random position at the beginning of the deployment.



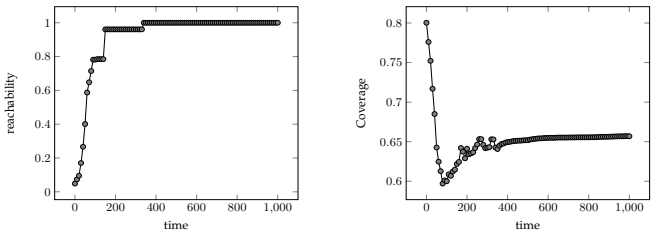
(a) Reachability Evolution (b) Coverage evolution

Fig. 10. Reachability and coverage evolution for a network of 50 nodes with a range of 10 on a 100×100 square. Reachability is computed every $10s$.

Figure 10(a) shows that the reachability is strictly increasing which means that when two nodes u and v are connected, u can reach v (and v can reach u), based on the definition of reachability, until the end of the simulation. We can also see that the reachability is very low, this is due to the fact that when two nodes are connected they are repelled by each other in order to increase the covered area, but they do not try to connect to the other nodes in the network. Note here that the repelling action between two nodes may create other links and therefore increase the reachability. In Figure 10(b) we can see the evolution of coverage. This figure shows that the coverage first decreases from $0s$ to $\sim 70s$. This behaviour is related to the part of Figure 10(a) where the reachability increases. At $\sim 70s$, the coverage increases which is related to the part of Figure 10(a) where the reachability is constant. We can also see this correlation between coverage and reachability at $\sim 110s$ and at after $\sim 200s$. We ran many simulations and the conclusion drawn are the same.

In order to provide a fully connected network, we have modified our algorithm and added a simple process to detect if the network is connected such as in [20]. Contrarily to [20], in our scheme, we do not need a specific node to flood the network, which is not scalable. In our scheme, a specified meeting point is known by each node (in our simulation the center of the field). A virtual node is considered to be located in this meeting point. Each node has a flag called con. When a node is connected to the virtual node (distance lower than R) it sets $con = 1$ and sends this status in its HELLO message. If a

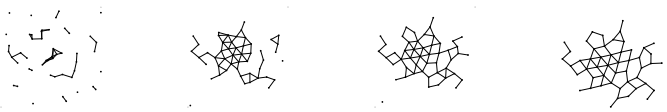
node does not receive any HELLO message with $con = 1$ and is not connected to the virtual node, it moves toward the virtual node position. If it receives an HELLO message with $con = 1$, it sets its own flags, $con = 1$, and runs maximal coverage algorithm as described in Section 7. The variable con can also be used to detect network partition but this improvement is left to future work. Note here that it is also possible to run different deployments.



(a) Reachability Evolution (b) Coverage evolution

Fig. 11. Reachability and coverage evolution with connectivity checking for a network of 50 nodes with a range of 10 on a 100×100 square. Reachability is computed every $10s$.

Figure 11 plots the reachability and the coverage evolution for a sample simulation when connectivity checking is used. We can see from Figure 11(a) that the reachability increases rapidly and the network is fully connected. Figure 11(b) shows that the coverage is first decreasing to provide connectivity and then is increasing. We can see that when the network is connected $\sim 350s$, the coverage is already close to its final value. This means that new connected nodes are already close to their final position. This assumption is confirmed by Figure 12 which plots the graph evolution at different simulation times.



(a) 20s (b) 100s (c) 200s (d) 500s

Fig. 12. Evolution of node's position and associated graph depending on time. In this simulation there is 50 nodes with a range of 10 on a square of 100×100. Nodes are not connected at the beginning of simulation

9 Conclusion

Connectivity is an important property in wireless networks and especially in wireless sensor networks. In this paper we provided some localized algorithms for mobile sensor deployment with connectivity guarantee. Our algorithm is divided into two independent parts. 1) **Direction computation**. In this part, the direction of the mobile sensor is computed depending on the application requirements. In this paper, we provide three examples of requirements for mobile sensor application deployment. The first deployment tries to maximize the area coverage. We showed that our deployment scheme provides a coverage close to the regular pattern coverage with squares. The second deployment is for Point of Interests (POI) coverage. We showed that the number of nodes involved in the connectivity preservation is independent of the number of nodes in the network. Therefore increasing the number of nodes, increases the coverage of the POI. Third, we proposed an example of barrier coverage and show that when an end point is defined, the deployment form a line between the starting point (with a fixed base station) and the end point. We also show that increasing the number of nodes increases the density of the line. 2) **Connectivity preservation.** In this part, we provided a connectivity preservation scheme to avoid nodes to be disconnected during their deployment. To preserve connectivity, nodes only maintain the connections with a subpart of its neighbors during the deployment. We chose the Relative Neighborhood Graph since it can be computed locally and it maintains global connectivity. We show by analysis that with a perfect physical channel the connectivity is guaranteed. Moreover, we show by simulation that our connectivity preservation scheme provide an high degree of reachability when message losses are considered.

We can note that we used unit disk graph model for description and evaluation of the algorithm but that it can also be applied with realistic models. The important point is that each node has to be able to discover its neighborhood and to compute a subset of its neighbors which guarantees connectivity preservation. It can be done by applying a variation of RNG. The Euclidean distance is simply replaced by a symmetrical metric such as received signal strength indicator (RSSI). The RNG algorithm uses the product of $RSSI(a,b) \times RSSI(b,a)$ ($RSSI(u,v)$ is the signal strength of HELLO messages sent by u and received by v) instead of $d(a,b)$.

The independence between the direction computation and the connectivity preservation allows the modification of each part without modifying the other part. The next steps of this work will focus on other connectivity preservation schemes and properties such as k-connectivity or with a degree constraints on each node. We will try to propose mobile deployments where coverage maximization, POI coverage and barrier coverage are needed at the same time or depending on the network evolution. Moreover, more investigation regarding efficient message losses handling, obstacle avoidance scheme and energy consumption will be done.

Acknowledgments

This work is partially funded by the French National Research Agency (ANR) under the VERSO RESCUE project (ANR-10-VERS-003) and the French Institut National de Recherche en Informatique et Automatique (INRIA) under the research action MISSION.

References

[1] J. Cartigny, F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovic, "Localized LMST and RNG based minimum-energy broadcast protocols in ad hoc networks," *Ad Hoc Networks (Elsevier)*, vol. 3, no. 1, pp. 1 – 16, 2005.

[2] S. Kumar, T. H. Lai, and A. Arora, "Barrier coverage with wireless sensors," in *International conference on Mobile computing and networking (ACM Mobicom)*, pp. 284–298, 2005.

[3] W. Cheng, M. Li, K. Liu, Y. Liu, X. Li, and X. Liao, "Sweep coverage with mobile sensors," in *International Parallel & Distributed Processing Symposium(IEEE IPDPS)*, pp. 1–9, 2008.

[4] X. Li, and H. Frey, and N. Santoro, and I. Stojmenovic, "Focused Coverage by Mobile Sensor Networks," in *International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, pp. 466–475, 2009.

[5] X. Li, and N. Mitton, and I. Ryl, and D. Simplot. "Localized Sensor Self-Deployment with Coverage Guarantee in Complex Environment," in *International Conference on AD-HOC Networks and Wireless (ADHOC-NOW)*, pp. 138–151, 2009.

[6] G. T. Toussaint, "The relative neighbourhood graph of a finite planar set," *Pattern Recognition*, vol. 12, no. 4, pp. 261 – 268, 1980.

[7] J. Carle and D. Simplot-Ryl, "Energy-efficient area monitoring for sensor networks," *Computer (IEEE)*, vol. 37, no. 2, pp. 40–46, 2004.
[8] D. Simplot-Ryl, I. Stojmenovic, and J. Wu, "Energy-efficient backbone construction, broadcasting, and area coverage in sensor networks," *Handbook of Sensor Networks (Wiley)*, pp. 43–380 343–380, 2005.
[9] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, "Near-optimal sensor placements: maximizing information while minimizing communication cost," *Information Processing in Sensor Networks (IPSN)*, pp. 2–10, 2006.
[10] D. Pompili, T. Melodia, and I. F. Akyildiz, "Deployment analysis in underwater acoustic wireless sensor networks," in *Proceedings of the International workshop on Underwater networks (ACM WUWNet)*. pp. 48–55, 2006.
[11] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai, "Deploying wireless sensors to achieve both coverage and connectivity," in *International symposium on Mobile ad hoc networking and computing (ACM Mobihoc)*, pp. 131–142, 2006.
[12] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks (Elsevier)*, vol. 6, no. 4, pp. 621 – 655, 2008.
[13] B. Wang, H. B. Lim, and D. Ma, "A survey of movement strategies for improving network coverage in wireless sensor networks," *Computer Communications (Elsevier)*, vol. 32, no. 13-14, pp. 1427 – 1436, 2009.
[14] X. Li, and A. Nayak, and D. Simplot-Ryl, and I. Stojmenovic, "Sensor Placement in Sensor and Actuator Networks", *Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication (Wiley)*, 2010.
[15] M. A. Batalin and G. S. Sukhatme, "Spreading out: A local approach to multi-robot coverage," in *International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2002, pp. 373–382.
[16] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *International Conference on Robotics and Automation (IEEE ICRA)*, vol. 2, pp. 500–505, 1985.
[17] R. Shahidi, M. Shayman, and P. Krishnaprasad, "Mobile robot navigation using potential functions," in *International Conference on Robotics and Automation (IEEE ICRA)*, pp. 2047–2053 vol.3, 1991.
[18] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *International Conference on Robotics and Automation (IEEE ICRA)*, vol. 2, pp. 1217–1222, 2002.
[19] S. Poduri and G. Sukhatme, "Constrained coverage for mobile sensor networks," in *International Conference on Robotics and Automation (IEEE ICRA)*, vol. 1, pp. 165–171 Vol.1, 2004.
[20] 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, pp. 836–848, 2009.
[21] 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*, vol. 8, no. 2, pp. 218–230, 2009.
[22] K. Savla, G. Notarstefano, F. Bullo, "Maintaining limited-range connectivity among second-order agents," *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 187–205 , 2009.
[23] M. Schuresko and J. Cortes, "Safe graph rearrangements for distributed connectivity of robotic networks", *Conference on Decision and Control (IEEE CDC)*, pp. 4602 –4607, 2007.
[24] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S. Srinivasa, and R. Sukthankar, "Brief paper: Decentralized estimation and control of graph connectivity for mobile sensor networks," *Automatica (Elsevier)*, vol. 46, no. 2, pp. 390–396, 2010.
[25] M. Zavlanos and G. Pappas, "Potential fields for maintaining connectivity of mobile networks," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 812–816, Aug. 2007.
[26] C. Shen, W. Cheng, X. Liao, and S. Peng, "Barrier coverage with mobile sensors," in *International Symposium on Parallel Architectures, Algorithms, and Networks (ISPAN)*, pp. 99–104, 2008.
[27] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," *Discrete Mathematics (Elsevier)*, vol. 86, no. 1-3, pp. 165–177, 1990.
[28] J. Jaromczyk and G. Toussaint, "Relative neighborhood graphs and their relatives," *Proceedings of the IEEE*, vol. 80, no. 9, pp. 1502–1517, 1992.
[29] V. J. Lumelsky and A. A. Stepanov, "Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape." *Algorithmica (Elsevier)*, vol. 2, pp. 403–430, 1987.
[30] K. Akkaya and F. Senel, "Detecting and connecting disjoint subnetworks in wireless sensor and actor networks," *Ad Hoc Networks (Elsevier)*, vol. 7, no. 7, pp. 1330 – 1346, 2009.
[31] N. Atay and B. Bayazit.., "Mobile wireless sensor network connectivity repair with k-redundancy', in *Algorithmic Foundation of Robotics (Springer)* vol. 57, pp. 35 – 49, 2009.
[32] A. Chen, S. Kumar, and T. H. Lai, "Designing localized algorithms for barrier coverage," in *International conference on Mobile computing and networking (ACM Mobicom)*. pp. 63–74, 2007.
[33] L. Qin and T. Kunz, "On-demand routing in manets: The impact of a realistic physical layer model," in *International Conference on Ad Hoc Networks and Wireless (ADHOC-NOW)*, pp. 37–48, 2003.
[34] S. Perur and S. Iyer, "Characterisation of a connectivity measure for sparse wireless multi-hop networks," *Ad Hoc & Sensor Wireless Networks*, vol. 3, no. 4, pp. 311–330, 2007.

Tahiry Razafindralambo received his MSc in applied statistics and computer science from the university of Antananarivo in 2001 and his PhD. degree in Computer Science from the INSA de Lyon in 2007. He is currently an INRIA full researcher. His research interests are mainly focused on distributed algorithms and protocols design for autonomous distributed and mobile system and more specifically wireless sensor, actuator and robots networks.

David Simplot-Ryl David Simplot-Ryl, member of the Institut Universitaire de France, is scientific head of the POPS project-team (joint project of INRIA, Universit Lille1 and CNRS) which is focused on small computing devices like smartcard or electronic tags. After a PhD (1997) in theoretical computer science, he joined the Universit Lille1 - Sciences et Technologies where he is full professor since 2004. His research interests include sensor and mobile ad hoc networks, mobile and distributed computing, embedded operating systems, smart objects and RFID technologies.