Points of Interest Coverage with Connectivity Constraints using Wireless Mobile Sensors

Milan Erdelj, Tahiry Razafindralambo, David Simplot-Ryl

Points of Interest Coverage with Connectivity Constraints using Wireless Mobile Sensors

Milan Erdelj[1], Tahiry Razafindralambo[1], and David Simplot-Ryl[2]

[1] INRIA Lille, {milan.erdelj, tahiry.razafindralambo}@inria.fr
[2] Univ. Lille 1, david.simplot@univ-lille1.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 constraints and provides the solution to it using Relative Neighborhood Graphs (RNG). 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 [12, 4]. 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 with 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.

Sensor placement 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 [6]. *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 [8]. *Point of Interest coverage* aims at monitoring specific points in the field of interest [9]. Different examples and results related to the deployment of sensors can be found in [18].

Previous works on Points of Interest (PoI) coverage using mobile sensors, such as [9], 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 our proposed solution, each sensor moves toward a PoI but also maintains the connectivity with a subset of its neighboring sensors. The use of the Relative Neighborhood Graph (RNG) reduction for choosing the subset of neighbors and local connection preservation provides a global connectivity of the network. Once the 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 cover 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*, 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 background which includes state of the art, assumptions, definitions and a problem statement. Section 3 describes our deployment algorithm with its properties. Simulation results are given in Section 4 where we consider static, moving and multiple PoI coverage. Real implementation of our algorithm using Wifibots[2] is presented in Section 5 and conclusions are drawn in Section 6.

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 PoI 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 [6, 13] for static random deployment strategies, to [10, 3] for off-line computation of sensor placement and to [18] or [15] for complete surveys. There are mainly three ways to optimize the deployment or the placement of mobile sensors: the coverage pattern [16], grid quorum [7] and virtual force based movements [5].

This paper belongs to the virtual force based movement category, where sensors are repelled or attracted each other by using virtual forces like electromagnetic particles. The sensors move step by step and the virtual forces are computed based on the set or a subset of neighboring sensors. With this deployment strategy connectivity and PoI coverage can be provided.

In this article, we consider an environmental monitoring application. 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. Surprisingly, very few works consider the actual problem of PoI coverage. To the best of our knowledge, the only work that consider PoI coverage is [9]. In [9], authors propose an algorithm to monitor some specific points periodically. Unlike the work presented in this paper, results from [9] do not consider connectivity issue. In [11], 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.

Our work considers single and multiple PoI coverage where connectivity has to be kept between the sensors that cover the PoI 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 Motivation and Preliminaries

By considering the environmental monitoring, 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. Furthermore, it is possible to have more that one event and these events could be mobile. To dynamically adapt to the changing requirements, the deployment algorithm must provide properties such as connectivity all along the deployment procedure. 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.*

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.*

2.3 Relative Neighborhood Graph

The Relative Neighborhood Graph (RNG) [14] 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.

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 [14]. 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 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. The direction of a sensor is given by the following unit vector : $\overrightarrow{\Delta} = \overrightarrow{d_p}/||\overrightarrow{d_p}||$, where $\overrightarrow{d_p}$ is the vector toward the PoI. 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 between 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, all the RNG neighbors stay connected. After the computation is finished, sensor u moves according to vector $\overrightarrow{m}$.

In order to avoid an infinite small movements of sensors, we add a condition on d. 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 Deployment Algorithm

Our deployment process is formally described in Algorithm 1. The algorithm is divided into three parts which 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 considers connectivity preservation and the sensor's movement is performed in the third part. Since these three parts are independent, it is simple to modify each part independently from the others. It is thus easy to modify the direction computation while using the same path planning algorithm and maintaining connectivity.

Algorithm 1 Deployment process

Part 1 – Direction computation on sensor u:

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

Part 2 – Distance/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 new position using: speed ν, direction $\overrightarrow{\Delta}$ and distance d.

2: Take obstacle into account.

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 [14].

Theorem 2. *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 $h = 0$.

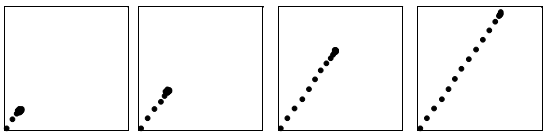
Theorems above show that our algorithm preserves connectivity all along the deployment procedure. Furthermore, we bring proof that, at the end of deployment, sensors used for connectivity are more likely to form a straight line toward the PoI and to be at the distance of $R - \epsilon_1$ from their neighbors.

4 Deployment Simulations

This section shows the performance evaluation results of our algorithm regarding the static, moving and multiple PoI coverage. Simulations were performed using WSNet[1]. In the simulations, δ is set to $5s$ and 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.

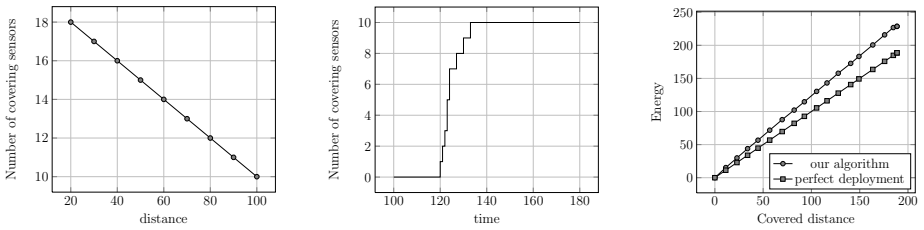
4.1 Static PoI

Deployment example. Figure 1 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.



(a) 30s (b) 60s (c) 120s (d) 180s

Fig. 1. 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 $[70, 100]$.



(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. 2. Coverage quality, deployment speed and energy consumption.

Coverage quality. The Figure 2(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 used for connectivity and 16 sensors cover the PoI.
Deployment speed. The Figure 2(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 $120s$).
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 2(c) shows the energy consumption of each sensor for a deployment of 20 sensors and a PoI at $[100, 100]$. 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 $E(u) = 105\times1+1\times21 = 126$ which is very close to 130.

4.2 Moving PoI

Deployment strategies and examples. We present three different strategies for covering a new PoI when the sensors are already deployed. In the first strat-

egy, STR1, the sensors first move back to the base station before deploying toward the new location of the PoI. This strategy provides a high coverage quality but increases the deployment duration and the amount of energy consumed. In the second strategy, STR2, the sensors try to move directly toward the location of the PoI without going back to the base station. This 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, STR3, is a mix of STR1 and STR2, in which sensors first move toward the segment $[b, p]$ and after, when the distance between the particular sensor and the segment is lower than $R/4$, toward the PoI. This strategy combines the advantages of STR1 and STR2.

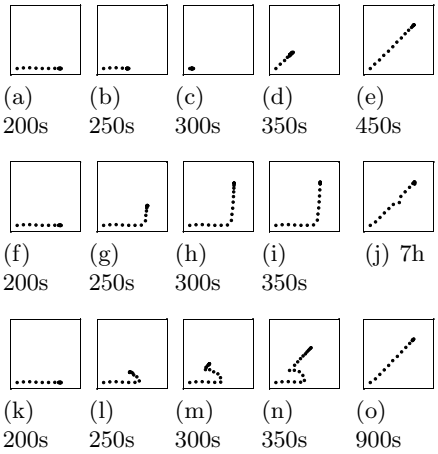


Fig. 3. Sensors' positions depending on time. In this simulation there is 20 sensors with a range of 10 on a square of 100×100. The PoI is first located at $[70, 0]$ and then at $[70, 70]$ after $200s$.

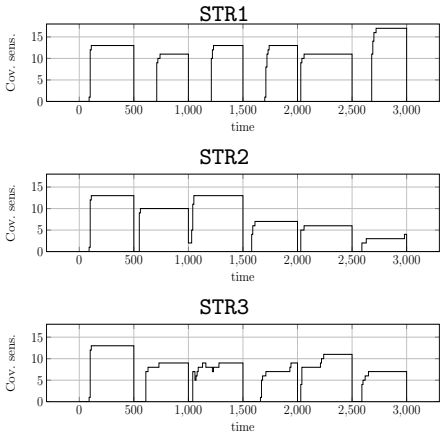


Fig. 4. Number of covering sensors w.r.t time. Simulation parameters: $R = 10$, 20 sensors including the base station. The simulation lasts $3000s$. A new location of the PoI is chosen every $500s$.

The Figure 3 shows the example of deployment for different tracking strategies. Figures 3(a) to 3(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 3(f) to 3(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 3(k) to 3(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.

Deployment performance. We run a simulation of $3000s$ with 20 sensors and move the PoI at a random location every $500s$. The Figure 4 plots the number of covering sensors depending on time, coverage quality and (re)deployment speed for these three strategies. We can see from Figure 4 that each new PoI location is covered by at least one sensor for each strategy and that, from the coverage quality point of view, `STR1` performs very good compared to other strategies. From the redeployment speed point of view, `STR2` shows very good performances. We can see 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 $[75, 1]$.

4.3 Multiple PoIs

Deployment strategies and examples. We have developed two strategies for the coverage of multiple PoIs. In our first strategy, `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 strategy, `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 [17].

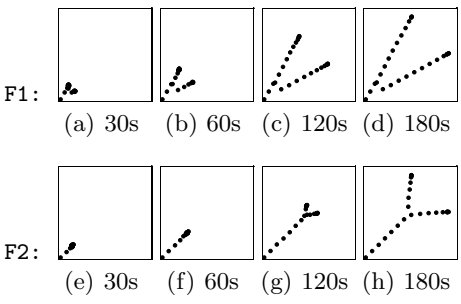


Fig. 5. Sensors' positions with multiple PoIs depending on time.

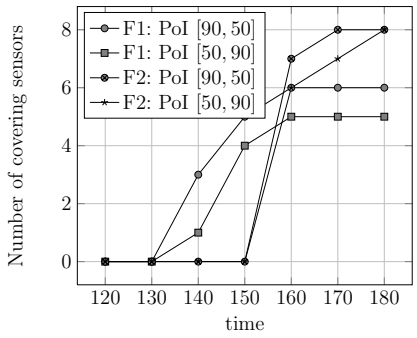


Fig. 6. Number of covering sensors w.r.t time for F1 and F2.

The Figure 5 shows the deployment example for `F1` and `F2` respectively. In these simulations, we use 30 sensors and two PoIs at $[90, 50]$ and $[50, 90]$. For `F1`, we consider that the set of sensors is divided into two subsets and each subset is assigned to one PoI. The Figure 5 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 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 moving toward its PoI, the sensor needs to reach the intermediate point.
Deployment performance. Figure 6 plots the number of covering sensors depending on time for two 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`. Note that for `F1`, the number of covering sensors is not equal for two PoIs since we consider 30 sensors in our simulation, including the base station. Therefore, 14 sensors are dedicated to one PoI and 15 sensors to the other. This is not the case for `F2` since a subset of sensors is used in common for connectivity.

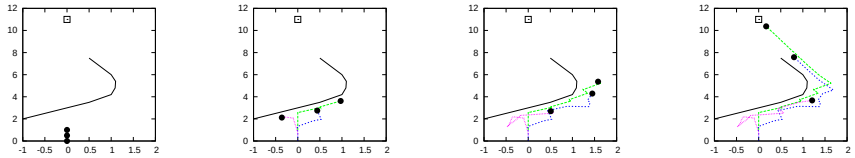
5 Implementation

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 [2] 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 802.11a/b/g 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 it arbitrarily and discarded messages received from robots that are out of communication range.

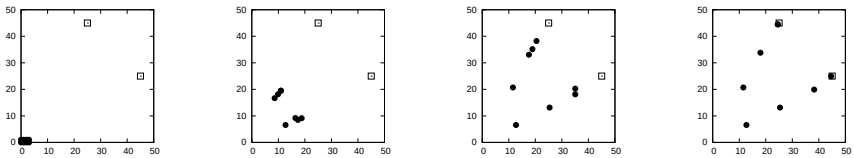
We have also implemented a simple obstacle avoidance since Wifibots are equipped with two IR proximity sensors. When a robot encounters an obstacle it stops moving and considers the obstacle in it's next direction computation. The right (left) hand rule is used to bypass the obstacle depending on the value from each IR sensor. The integration of an efficient obstacle avoidance scheme 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 frequency of Hello messages. Due to space limitations we will not describe the Wifibot implementation of our algorithm in details.

Figure 7 shows the example of deployment for a single PoI as presented in Section 4.1 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. Figure 8 shows results regarding multiple PoI coverage as presented in Section 4.3. In this experimentation we used 8 Wifibots with a range of $15m$ and two PoIs at $[25, 45]$ and $[45, 25]$.



(a) 0s (b) 20s (c) 40s (d) 60s

Fig. 7. Wifibot deployment with an obstacle.



(a) 0s (b) 60s (c) 120s (d) 150s

Fig. 8. Multiple targets at $[20, 45]$ and $[45, 20]$ with comm. range of $15m$ (8 robots).

6 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 Relative Neighborhood Graph (RNG). Indeed, if a graph G is connected, the RNG extracted from G is also connected. Hence, during their movements, the sensors only have to keep the connection with their RNG neighbors to keep the whole graph connected. Moreover, the RNG computation 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. WSNet. http://wsnet.gforge.inria.fr.
2. WifiBot. http://www.wifibot.com.
3. X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *international symposium on Mobile ad hoc networking and computing (ACM Mobihoc)*, pages 131–142, New York, USA, 2006.
4. G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange. Sensorscope: Out-of-the-box environmental monitoring. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 332–343, Los Alamitos, USA, 2008.
5. M. Batalin and G. Sukhatme. Spreading out: A local approach to multi-robot coverage. In *International Symposium on Distributed Autonomous Robotic Systems (DARS)*, pages 373–382, Fukuoka, Japan, 2002.
6. J. Carle and D. Simplot-Ryl. Energy-efficient area monitoring for sensor networks. *Computer*, 37(2):40–46, 2004.
7. S. Chellappan, W. Gu, X. Bai, D. Xuan, B. Ma, and K. Zhang. Deploying wireless sensor networks under limited mobility constraints. *IEEE Transactions on Mobile Computing*, 6(10):1142–1157, Oct. 2007.
8. A. Chen, S. Kumar, and T. Lai. Designing localized algorithms for barrier coverage. In *ACM international conference on Mobile computing and networking (ACM Mobicom)*, pages 63–74, New York, USA, 2007.
9. W. Cheng, M. Li, K. Liu, Y. Liu, X. Li, and X. Liao. Sweep coverage with mobile sensors. In *IEEE International Parallel & Distributed Processing Symposium (IEEE IPDPS)*, pages 1–9, Miami, USA, 2008.
10. A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 2–10, Nashville, USA, 2006.
11. X. Li, H. Frey, N. Santoro, and I. Stojmenovic. Focused-coverage by mobile sensor networks. In *IEEE International Conference on Mobile Adhoc and Sensor Systems (IEEE MASS)*, pages 466 –475, Macau, China, 2009.
12. K. Martinez, R. Ong, and J. Hart. Glacsweb: a sensor network for hostile environments. In *IEEE Conference on Sensor and Ad Hoc Communications and Networks. (IEEE SECON)*, pages 81–87, Santa Clara, USA, 2004.
13. D. Simplot-Ryl, I. Stojmenovic, and J. Wu. Energy-efficient backbone construction, broadcasting, and area coverage in sensor networks. *Handbook of Sensor Networks*, pages 343–380, 2005.
14. G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4):261 – 268, 1980.
15. 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.
16. 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.
17. P. Winter. Steiner problem in networks: a survey. *Networks*, 17(2):129–167, 1987.
18. 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.