Autonomous Deployment of Heterogeneous Mobile Sensors

N. Bartolini, T. Calamoneri, T. La Porta, S. Silvestri

Abstract—In this paper we address the problem of deploying heterogeneous mobile sensors over a target area. Traditional approaches to mobile sensor deployment are specifically designed for homogeneous networks. Nevertheless, network and device homogeneity is an unrealistic assumption in most practical scenarios and previous approaches fail when adopted in heterogeneous operative settings. For this reason we introduce VorLag, a generalization of the Voronoi based approach which exploits the Laguerre geometry. We theoretically prove the appropriateness of our proposal to the management of heterogeneous networks. In addition, we demonstrate that VorLag can be extended to deal with dynamically generated events or uneven energy depletion due to communications. Finally, by means of simulations, we show that VorLag provides a very stable sensor behavior, with fast and guaranteed termination and moderate energy consumption. We also show that VorLag performs better than its traditional counterpart and other methods based on virtual forces.

Index Terms—Device heterogeneity, self-deployment, Voronoi-Laguerre diagrams.

1 Introduction

The deployment of mobile sensors is attractive in many scenarios. Mobile sensors may be used for environmental monitoring or public safety to track the dispersion of pollutants, gas plumes or fires. In such scenarios it is difficult to achieve an exact sensor placement through manual means. Instead, sensors are deployed from a distance, e.g. from a safe location or from an aircraft, and then reposition themselves to provide the required sensing coverage.

The potential of such applications has inspired a great deal of work on algorithms for deploying mobile sensors. Most of this work has addressed the deployment of homogeneous sensors to achieve a uniform coverage of a certain density in a specific Area of Interest (AoI).

In this paper we address two more practical and challenging problems: (i) the deployment of heterogeneous sensors to achieve full coverage, and (ii) the deployment of sensors to achieve coverage of varying density within an AoI. The first application accommodates sensors that may have different sensing ranges due to design or operating conditions, for example depleted battery supplies or damage to a transducer. The second application addresses the need for a higher density of sensing resources at a particular site where perhaps an event has been detected and requires more analysis.

In particular, we realize that the Voronoi based approach presented in [1] does not solve the problems of deployment with heterogeneous sensors or varying density over a field. Namely, in these scenarios, the sensors do not cover the required area completely, because they stop moving when they wrongly perceive that they are exploiting their sensing capabilities at the maximum extent. To solve this problem we introduce the notion of Laguerre distance into the Voronoi based algorithm. By using the Laguerre distance we are able to explicitly take account of device diversity and varying density requirements. The resulting algorithm, which we call VorLag, solves both deployment problems effectively. The primary additions to the original Voronoi approach are the use of the Laguerre distance and the redefinition of some algorithm rules to ensure algorithm termination and improve convergence time.

We compare VorLag with an algorithm based on virtual forces, i.e. belonging to another class of well accepted deployment algorithms. We modify the algorithm introduced in [2] so that it may operate in scenarios requiring the use of heterogeneous sensors or deployment of varying density. We find that VorLag has better performance and characteristics than the virtual force algorithm. Because the virtual force algorithm balances the distance between sensors, and does not specifically target sensor heterogeneity or a specific coverage density, it takes longer to converge, and under some circumstances is not guaranteed to converge at all. Also, we find that unlike VorLag, the virtual force algorithm requires off-line tuning of several parameters that have a large impact on its performance which makes the virtual force algorithm impractical.

In summary, our contributions are:

- We identify the limitations of Voronoi-based algorithms to deal with heterogeneous sensors; we introduce the Laguerre distance and show several important properties that allow it to be used in this setting;

- *N. Bartolini, T. Calamoneri and S. Silvestri are with the Department of Computer Science, Sapienza University of Rome, Italy. E-mail: {bartolini, calamo, silvestris}@di.uniroma1.it*
- *T. La Porta is with the Networking and Security Research Center, Penn State University, PA. E-mail: tlp@cse.psu.edu*

- We propose a new algorithm based on the use of Voronoi diagrams in the Laguerre geometry to solve the problem of deploying heterogeneous mobile sensors; we also extend the new algorithm to make it work in operative settings with time-varying and position-dependent coverage requirements;
- We extend a previously published algorithm based on virtual forces to accommodate heterogeneous sensors and prove important properties about this new algorithm.

We theoretically and experimentally give evidence that the Voronoi-based approach does not operate correctly in a setting with heterogeneous sensors. We also compare VorLag with the virtual force algorithm and determine the fundamental causes behind the limitations of the virtual force approach.

The VorLag algorithm is practical in that it provides very stable sensor behavior, with fast and guaranteed termination and moderate energy consumption. It does not require manual tuning or perfect knowledge of the operating conditions, and works properly even if the sensor positioning is imprecise. The algorithm only requires loose synchronization and local communication. Because it converges quickly and does not require a priori knowledge of the deployment environment, it is also well suited for dynamic environments in which the sensing density requirements change over time.

The paper is organized as follows. Related work is presented in Section 2. In Section 3 we motivate the problem and introduce some preliminary concepts. Section 4 presents the algorithm VorLag. In Section 5 we address the problem of density driven deployment. Section 6 summarizes a virtual force based algorithm that we use for performance comparisons whose results are shown in Section 7. Section 8 concludes the paper with some final remarks.

2 Related Work

Various approaches have been proposed to self-deploy mobile sensors. The virtual force approach (VFA) models the interactions among sensors as a combination of attractive and repulsive forces. As a result of these antagonist forces, the sensors spread throughout the environment. One of the first algorithms based on VFA was presented in [3]. Drawbacks of VFA include complex required tuning of several parameters and an oscillatory behavior of sensors. Possible improvements to decrease the oscillations include the introduction of dissipative forces [3], [4], or the definition of arbitrary thresholds as stopping conditions [5], [6]. The tuning of such thresholds is laborious and relies on an off-line configuration. The virtual force model is also at the basis of several other proposals [2], [7], [8], [9]. Of these proposals we focus on the one presented in [2] under which a dynamically calculated constraint on the length of sensor movements prevents oscillations. This algorithm is described in more detail in Section 6 and is referred in this paper as a benchmark for performance comparisons.

Techniques based on computational geometry model the deployment problem in terms of Voronoi diagrams or Delaunay triangulations. According to [1], each sensor iteratively calculates its Voronoi polygon, determines the existence of coverage holes and moves to a better position if necessary. This approach inspired our proposal and it can be obtained as an instance of our general approach, when sensor capabilities are homogeneous. A dual approach exploits Delaunay triangulation [10]. This approach does not guarantee oscillation avoidance if proper threshold parameters are not set.

In [11], the authors introduce a technique for sensor deployment for specific settings of the sensing radius. Unlike this work, our approach deals with the more typical devices for which the transmission radius is significantly larger than the sensing radius, hence coverage implies connectivity. Two other approaches based on the construction of a regular triangular lattice are proposed in [12] and [13], whereas some discussions on the appropriateness of lattice deployments can be found in [14] and [15]. A grid shaped deployment technique is also proposed in [16] where sensors are static devices dropped in the area of interest by mobile robots. The work [17] illustrates the use of systems theory to analyze emergent behaviors in biological systems and to design autonomous and reliable robotic networks. The proposed techniques rely on assumptions of device homogeneity that we do not need in our proposal.

Recent papers [18], [19] focus on static sensor deployment with variable density in order to mitigate the effects of the uneven energy depletion due to communication with a sink [20], [21]. The work [4] introduces a unified solution for sensor deployment and relocation to adapt the density to the proximity of events of interest. In [22] the problem of sensor heterogeneity is specifically addressed, but under assumptions on the network topology that are very restrictive with respect to those considered in this paper.

We conclude this section by pointing out that the idea of using generalized Voronoi diagrams is not new in the area of sensor networks. The work [23] uses a generalization of the Voronoi approach to deploy devices endowed with elliptic sensing capabilities, whereas the work [24] makes use of Voronoi generalizations to calculate the routing paths of sensors endowed with different initial energy. To the best of our knowledge, none of the previous works uses the approach proposed here to address the problem of heterogeneous mobile sensor deployment or deployment of varying density.

3 Motivation and Preliminaries

In the following subsections we show the limits of existing Voronoi based approaches to guide the movements of heterogeneous sensors. We then give the basics of our new algorithm.

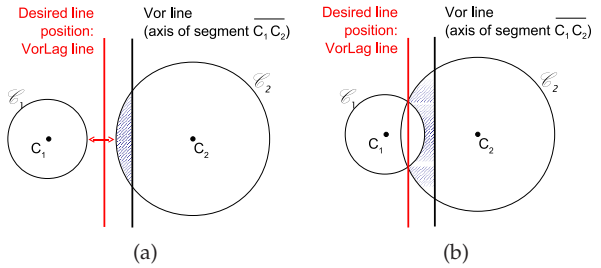


Fig. 1: Different positions of the line which is equidistant to C_1 and C_2 according to the Euclidean (Vor) and to the Laguerre distance (VorLag) in the case of intersecting (a) and non-intersecting circles (b).

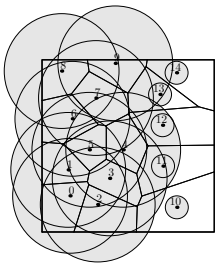


Fig. 2: Critical configuration for the traditional Voronoi algorithm.

3.1 Traditional Voronoi approach

We recall that given N generating points in the plane, $C_i = (x_i, y_i)$ with $i = 1, \ldots, N$, the *Voronoi polygon* $V_{\mathsf{Vor}}(\mathscr{C}_i)$ of the generating point C_i is defined as

$$V_{\mathsf{Vor}}(C_i) = \{P \in \Re^2 | d_E^2(C_i, P) \leq d_E^2(C_j, P), \forall j \neq i\}.$$

The Voronoi algorithm [1] works on a round basis. In each round every sensor communicates with its neighbors to discover their position, then calculates its Voronoi polygon, and moves inside it towards a point where it can contribute a better coverage of the polygon. The choice of the destination point can be based on several criteria.

The algorithm is motivated by the observation that in a homogeneous network, each edge of a Voronoi diagram lies on a line which partitions the AoI in two half planes, each one covered better by either one or the other of the two generating sensors. In the homogeneous setting, coverage holes are always crossed by at least one edge, hence it is correct to use Voronoi diagrams to discover them. Furthermore, a Voronoi polygon is covered better by movements of its generating sensor, rather than of any other one.

These properties no longer hold in the case of heterogeneous sensors thus causing the Voronoi-based approach to fail. Figure 1(a) shows an example in which the line "Vor" is equidistant from the points C_1 and C_2, centers of the circles $\mathscr{C}_1$ and $\mathscr{C}_2$, respectively. It is easy to see that since this line does not cross the intersection between the two circles $\mathscr{C}_1$ and $\mathscr{C}_2$, it does not partition the plane as required and incorrectly assigns the shaded portion of the circle $\mathscr{C}_2$ to the sensor located in the point C_1. Instead, the desired line position is the one that assigns each sensor to the half plane that it can cover best, as in the case of the line labeled "VorLag". In subsection 3.2 we will show that this line is equidistant from the points C_1 and C_2 in the Laguerre geometry.

Figure 1(b) shows another case in which the Voronoi axis does not properly partition the AoI as it is done by the VorLag axis.

Let us consider the Figure 2 in which several heterogeneous sensors are deployed over a rectangular AoI, where the left zone is covered redundantly and the right zone is largely uncovered. The use of the Voronoi algorithm produces no movements in such a configuration. The sensors with a small sensing circle (sensors 10-14) do not move, as their sensing range is completely included by their Voronoi polygon. The sensors with larger sensing circles (sensors 0-9), do not move either, because they already cover their own polygon completely. Figure 2 represents a critical configuration for the Voronoi algorithm for which the wrong placement of the edges leads to a *barrier effect*. This effect is one of the reasons of the inapplicability of the traditional Voronoi approach even in the case of varying density requirements as will be clarified in Section 5.

3.2 Voronoi diagrams in the Laguerre geometry

The *Voronoi diagram* is one of the fundamental concepts in computational geometry. It has been generalized in many ways as summarized in [25]. A generalization can be obtained by replacing the notion of Euclidean distance with a variety of different formulations. Unfortunately, most of them lead to diagrams whose edges are curves, whereas in the ordinary Voronoi diagrams they are portions of straight lines. An exception is the Voronoi-Laguerre diagram [26]. In this case the diagram is formed by straight lines. This property is particularly attractive as it imposes that polygons are convex, which simplifies the calculation of the sensor destinations. Moreover, as shown in Theorem 3.1, the Voronoi polygon in the Laguerre geometry properly identifies the responsibility region of each sensor in a heterogeneous setting, guiding its movement towards the region it can cover best.

Given a circle $\mathscr{C}$ with center $C = (x_C, y_C)$ and radius r_C, and a point of the plane $S = (x_S, y_S) \in \Re^2$, the Laguerre distance $d_L(\mathscr{C}, S)$ between the circle $\mathscr{C}$ and the point S is defined in terms of the Euclidean distance $d_E(C, S)$ between points C and S:

$$d_L^2(\mathscr{C}, S) = d_E^2(C, S) - r_C^2.$$

It should be noted that this metric is not a distance in the mathematical sense. Actually $d_L^2(\mathscr{C}, S)$ can be negative. The sign of $d_L^2(\mathscr{C}, S)$ depends on the position of S with respect to the circle $\mathscr{C}$. It is negative if S lies inside circle $\mathscr{C}$, whereas it is positive if S is located outside. In this sense, rather than a "distance", this metric should only be interpreted as a "degree of farness" [27].

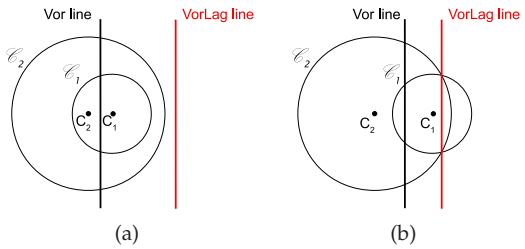


Fig. 3: Radical axes of two circles in the Euclidean and in the Laguerre geometry.

Lemma 1. *Given two circles $\mathscr{C}_1$ and $\mathscr{C}_2$ with distinct centers C_1 and C_2, and radii r_1 and r_2, respectively, the locus of the points equally distant, in the Laguerre geometry, from the two circles is a straight line, called the radical axis of $\mathscr{C}_1$ and $\mathscr{C}_2$.*

The radical axis is perpendicular to the segment connecting the centers C_1 and C_2 of the two circles. This axis is located at distance k from C_1, with $k \triangleq \frac{d_E(C_1,C_2)}{2} + \frac{r_1^2-r_2^2}{2d_E(C_1,C_2)}$.

Proof: Without loss of generality, we can use a Cartesian reference such that $C_1 \equiv (0,0)$ and $C_2 \equiv (d_E(C_1,C_2),0)$. The locus of the points equidistant from the two circles, according to the Laguerre geometry, can be parametrized by the point $P(t) = (x(t),y(t)) \in \Re^2$, by varying the parameter $t \in \Re$. This locus is such that $d_L(P(t),\mathscr{C}_1) = d_L(P(t),\mathscr{C}_2)$. If the centers coincide, the locus is either the whole plane (in the case with $r_1 = r_2$, as all the points of the plane are equidistant to the coincident circles $\mathscr{C}_1$ and $\mathscr{C}_2$) or does not exist (in the case with $r_1 \neq r_2$, as all the points of the plane are closer to the circle having larger radius). Hence we investigate the properties of such a locus in the case of distinct centers $C_1 \neq C_2$, i.e. with $d_E(C_1,C_2) > 0$. In this case, the equation of the aforementioned locus is:

$$x^2(t) + y^2(t) - r_1^2 = (d_E(C_1,C_2) - x(t))^2 + y^2(t) - r_2^2,$$

from which we deduce

$$x(t) = \frac{d_E(C_1,C_2)}{2} + \frac{r_1^2 - r_2^2}{2d_E(C_1,C_2)} = k, \ \forall t \in \Re.$$

As the equation of this locus is independent of the parameter t, it represents a straight line of equation $x = k$, thus proving the lemma. □

We now focus on a case that does not occur with ordinary Voronoi polygons. Depending on the extension of the overlap between two circles, the two centers may fall on the same side of the radical axis as shown in Figure 3. The following lemma characterizes these situations.

Lemma 2. *Given two circles $\mathscr{C}_1$ and $\mathscr{C}_2$ with distinct centers C_1 and C_2, and radii r_1 and r_2, respectively, their centers lie on the same side of the radical axis if and only if $d_E^2(C_1,C_2) < |r_1^2 - r_2^2|$.*

Proof: Without loss of generality we position the Cartesian reference as we did in the proof of Lemma 1. The two centers can lie on the same side of the axis both on the right and on the left side. They both lie on the left side of the axis if and only if their Euclidean distance is such that $d_E(C_1,C_2) \leq k$, that is if and only if $d_E^2(C_1,C_2) \leq r_1^2 - r_2^2$. This is feasible only when $r_1 \geq r_2$.

The two centers both lie on the right side of the radical axis if and only if k is negative, that is $k < 0$, which occurs if and only if $d_E^2(C_1,C_2) \leq r_2^2 - r_1^2$. This is feasible only when $r_1 \leq r_2$.

Merging the two results, the thesis follows. □

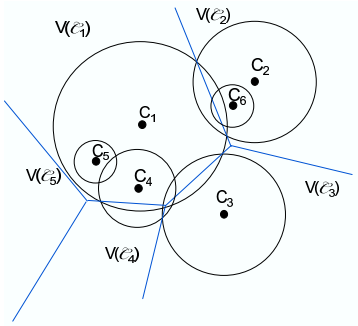


Fig. 4: Voronoi-Laguerre diagram with empty and null polygons

Given N circles in the plane $\mathscr{C}_i$ with centers $C_i = (x_i,y_i)$ and radii r_i, $i = \ldots,N$, the *Voronoi-Laguerre polygon* $V(\mathscr{C}_i)$ for circle $\mathscr{C}_i$ is defined as

$$V(\mathscr{C}_i) = \{P \in \Re^2 | d_L^2(\mathscr{C}_i,P) \leq d_L^2(\mathscr{C}_j,P), \forall j \neq i\}.$$

As a consequence of Lemma 1, $V(\mathscr{C}_i)$ is always convex and its vertices are called *Voronoi-Laguerre vertices*. Obviously, if $r_i = r_j$ for all $i,j = 1,\ldots,N$, the Voronoi-Laguerre diagram reduces to the ordinary Voronoi diagram.

We highlight that the polygon $V(\mathscr{C}_i)$ may not contain any point of the plane when the half-planes generated by the radical axes of a circle $\mathscr{C}_i$ have no overlap. In this case, the Voronoi-Laguerre polygon $V(\mathscr{C}_i)$ is called a *null polygon*. We also note that in view of Lemma 2, the point C_i may not lay inside $V(\mathscr{C}_i)$, even if this polygon is not null. Such a polygon is called an *empty polygon*.

Figure 4 shows an example of Voronoi-Laguerre diagram that contains both null and empty polygons. Namely, the Voronoi-Laguerre polygon of $\mathscr{C}_6$ is null, and the polygons of $\mathscr{C}_4$ and $\mathscr{C}_5$ are empty. On the contrary, the polygons of $\mathscr{C}_1$, $\mathscr{C}_2$ and $\mathscr{C}_3$ are non-empty and consequently non-null.

The following theorem is fundamental for the formulation of the Voronoi-Laguerre algorithm that we will present in the next section. It states that the intersection of $V(\mathscr{C}_k)$ with a circle $\mathscr{C}_j$ is contained in $\mathscr{C}_k$, hence it does not exist any point in $V(\mathscr{C}_k)$ contained in some $\mathscr{C}_j$, that is not also contained in $\mathscr{C}_k$. Vice-versa, it states that if a point of $V(\mathscr{C}_k)$ is not also in $\mathscr{C}_k$, then that point is definitely uncovered. This property is fundamental for sensors to detect coverage holes.

Theorem 3.1. *Let us consider n circles $\mathscr{C}_i$ with centers $C_i = (x_i,y_i)$ and radii r_i, $i = 1,\ldots,n$, and let $V(\mathscr{C}_i)$ be the Voronoi-Laguerre polygon of the circle $\mathscr{C}_i$. For all $k,j = 1,2,\ldots,n$, $V(\mathscr{C}_k) \cap \mathscr{C}_j \subseteq \mathscr{C}_k$.*

Proof: By contradiction, assume there exists a point $P \in V(\mathscr{C}_k)$ contained in $\mathscr{C}_j$ for some $j \neq k$ but not contained in $\mathscr{C}_k$. On the one hand, as $P \in V(\mathscr{C}_k)$ and in view of the definition of Voronoi-Laguerre polygon, it must be $d_L(\mathscr{C}_k, P) < d_L(\mathscr{C}_j, P)$ that is, equivalently,

$$d_E^2(\mathscr{C}_k, P) - r_k^2 < d_E^2(\mathscr{C}_j, P) - r_j^2. \quad (1)$$

On the other hand, since P is contained in $\mathscr{C}_j$ but not in $\mathscr{C}_k$, $d_E(\mathscr{C}_j, P) \leq r_j$ and $d_E(\mathscr{C}_k, P) > r_k$. Substituting these inequalities in Equation 1 we get $0 < 0$, a contradiction. □

In the next section, we show how the concept of a Voronoi-Laguerre diagram can be applied to partition the AoI to support the movements of heterogeneous sensors.

4 The Voronoi-Laguerre approach for heterogeneous sensor deployment

In this section we describe the algorithm VorLag, based on the construction of the Voronoi diagram in the Laguerre geometry by N heterogeneous sensors, in order to self-deploy in the AoI. We first state our assumptions, give a high level view of the algorithm and finally provide details.

4.1 Assumptions

Given N heterogeneous sensors:

1) The sensing and communication radii of the sensor s_i are r_i and r_i^{tx}, respectively, with $i = 1, \ldots, N$.
2) The communication and the sensing radii of any two sensors are such that $r_k + r_l < \min_i r_i^{tx}$, for all $k, l = 1, \ldots, N$, that is any two sensors are able to communicate if their sensing circles are tangential.
3) Each sensor knows the coordinates of the AoI and can determine its own location (e.g. using low cost GPS, notice that the algorithm is not particularly sensitive to inexact location).
4) The sensors are loosely synchronized (in order to provide a round-based execution).
5) The sensors move at possibly different speeds v_i, $i = 1, \ldots, N$.

4.2 The idea

VorLag is based on the calculation of the Voronoi-Laguerre diagram determined by the sensor locations over the AoI and their related sensing radii. Such a diagram partitions the AoI into disjoint polygons, each of them related to one generating sensor. Hence each sensor uses the information related to its Voronoi-Laguerre polygon to determine the presence of coverage holes and to decide future movements.

Our sensor deployment protocol runs iteratively. In each round, the sensors broadcast their locations and construct their local Voronoi-Laguerre polygons on the basis of the information received from neighbors. Each sensor evaluates the existence of coverage holes within its polygon and makes movement decisions accordingly.

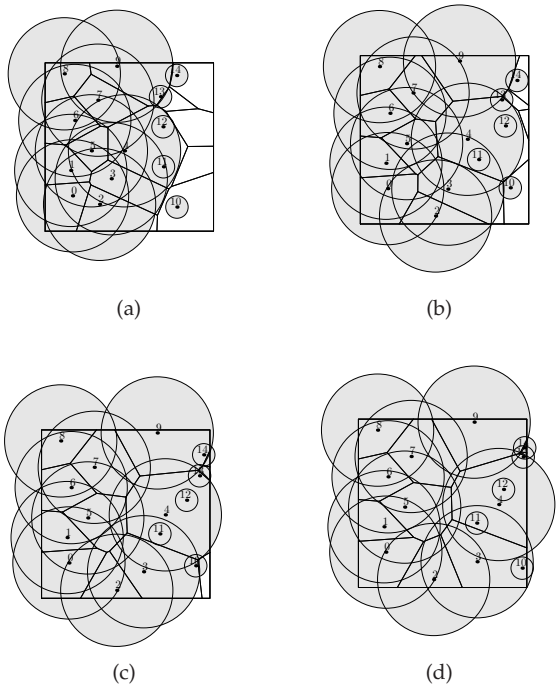


Fig. 5: Execution of the VorLag algorithm under an initial configuration that is critical for the traditional Voronoi approach: initial configuration (a), round 6 (b), round 9 (c), round 12 - final configuration (d).

It should be noted that, as formalized by Lemma 2, in the Voronoi-Laguerre diagrams, there are some interesting situations that do not occur with ordinary Voronoi diagrams. First, the polygon of a sensor may possibly be null (the sensor has no polygon: this can happen when its sensing range is completely included in the union of the sensing circles of other sensors). In this case, the sensor should not move, as there is no direction that it can take to locally improve its coverage, and it should wait for other sensors to place themselves. Second, a sensor may be situated outside its polygon, that is therefore necessarily empty. In this case, the sensor should move towards its polygon if the movement can improve the coverage, as the generating sensor is the best candidate to improve the coverage of its own polygon.

In Figure 5 we show progressive runs of the algorithm execution under the initial configuration of Figure 2. We see that for VorLag, unlike Vor, this setting does not cause a barrier effect.

4.3 The details

In this subsection we describe the algorithm in more detail: we introduce the Voronoi-Laguerre diagram construction and explain the way it can be used to guide the basic movements of the sensors. The pseudo-code of the algorithm executed by each sensor is in Figure 6.

To construct its Voronoi-Laguerre polygon, each sensor calculates the radical axes it generates with respect to all

SKETCH OF THE VORLAG ALGORITHM AT EACH ROUND:

```
Perform neighbor discovery
Compute the Voronoi-Laguerre polygon V(s_i)
if V(s_i) is null
   do nothing
else
   if a coverage hole exists
      compute O_m(s_i)
      compute local coverage
      if |v_{i,O_m(s_i)}| > d_max_i
         shrink |v_{i,O_m(s_i)}| to be d_max_i
      if (a movement according to v_{i,O_m(s_i)}
      increases the local coverage)
         move according to v_{i,O_m(s_i)}
```

Fig. 6: VorLag executed by the sensor s_i.

its neighbors, by using the locally available information. Each radical axis divides the plane into two half-planes, of which only one should be considered; the intersection of all the considered half-planes determined by each radical axis is the Voronoi-Laguerre polygon of the sensor.

If a sensor has a null polygon, it does not move, as a null polygon is generated only when the sensor is in an overcrowded area and there is no movement it can make to locally improve the coverage.

If the sensor has a non-null polygon, the algorithm determines the target location of the sensor s as the point inside its Voronoi-Laguerre polygon whose distance to the farthest Voronoi-Laguerre vertex is minimized. This point is called *miniMax* of the polygon of s, denoted as $O_m(s)$. Notice that, with respect to the calculus of the miniMax point, either the Laguerre or Euclidean distance may be used equivalently, with the same result.

Positioning sensors in their miniMax points can reduce the variance of the distances to the Voronoi-Laguerre vertices, resulting in a more regularly shaped Voronoi-Laguerre polygon, which better utilizes the device sensing capabilities.

Notice that, due to the limited communication range, a node may not be able to perceive the presence of some neighbor sensors. Hence we account for some inaccuracies in the local construction of the polygons.

In order to avoid excessive movements and collisions with the sensing disk of any other approaching sensor, the sensor s_i is only allowed to make movements that are smaller than $d_{\mathtt{max}i} = \frac{r_{tx}}{2} - r_i$, where $r_{tx} = \min_i r_i^{tx}$. It should be noted that we impose that $d_{\mathtt{max}i}$ is the maximum distance traversed in a single round along $\vec{v}_{O_m(s_i)}$, that is the vector directed from the position held by s_i to $O_m(s_i)$. Hence, together with the minimum sensor speed $v_{\mathtt{min}} = \min_i v_i$, this influences the duration of the algorithm round.

In order to reduce useless movements, we introduce a *movement-adjustment* scheme similar to the one used in [1]: a sensor moves towards its target location only if the movement leads to an increase in the local coverage of its own polygon.

We conclude this section by highlighting that we adopt the optimization proposed in [1] that "explodes" clusters of sensors at the first round in order to have a more uniform initial deployment and a faster algorithm execution. Briefly, each sensor determines the number of its communicating neighbors. If this number is above a certain threshold the sensor chooses a random position within an area centered at itself whose extent is the size of the area that would be occupied by the same number of neighbors if all the sensors were evenly deployed.

4.4 Algorithm properties

Before proving some properties of the algorithm VorLag, we make some observations and introduce some notation.

The Voronoi-Laguerre polygon locally computed by each sensor can be inaccurate with respect to the one calculated with complete information. We hereafter refer to *real polygon* as to the polygon constructed under complete information, whereas we define as *local polygon* the polygon that is constructed on the basis of the information that is locally available. In particular, the local polygon of the sensor s includes its real polygon, but can be larger because s does not take account of the sensors that are not in radio proximity.

In Figure 7 we show an example where two sensors s_1 and s_2 are out of the communication range of s_3. As a consequence s_1 and s_2 cannot know the position of s_3, hence they do not keep it into account when building their local polygon. Likewise, s_3 does not have any information regarding the position of the sensors s_1 and s_2. Therefore there is a discrepancy between local and real polygons. The real polygon $V(\mathscr{C}_1)$ of s_1 is AHIE, while its local polygon is AHF. The real polygon $V(\mathscr{C}_2)$ of s_2 is DHIG, while its local polygon is DHFBC. Similarly, s_3 assumes its local polygon is the whole AoI, whereas its real polygon $V(\mathscr{C}_3)$ is EIGCB.

Because of the possible discrepancy between the local and real polygons, the movement of a sensor s towards the miniMax point of its local polygon can target a position that is potentially outside the real polygon of s. Indeed even when a constraint on the maximum movement d_{max} is applied, the heterogeneity among devices is such that the "unknown" axis can be located at a distance from the generating sensor which is smaller than the threshold d_{max}. In fact, there is no way to set the value of d_{max} so as to ensure that a sensor does not exit its real polygon. This is due to the fact that the position of the Voronoi-Laguerre axis depends not only on the positions but also on the radii of potentially unknown neighbor nodes.

The area where a sensor s_i can improve the coverage of its local polygon with a single round movement is the region polygon $P(s_i)$ generated by the intersection of the local polygon of sensor s_i, with the circle centered in the location of s_i and radius $r_{tx}/2$. Indeed the farthest point

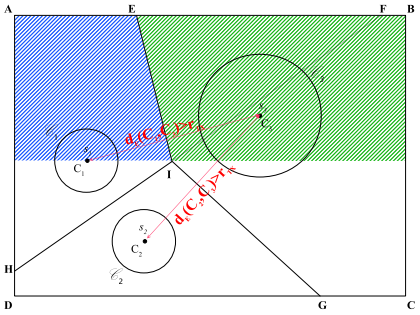


Fig. 7: Example of local construction of the Voronoi-Laguerre polygons

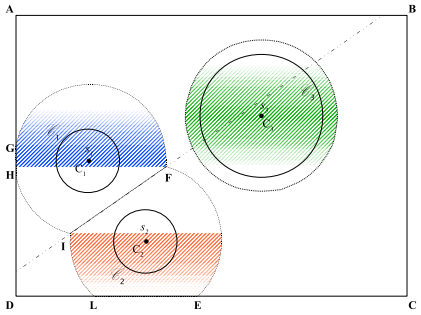


Fig. 8: Partition of the AoI with curve polygons

that can be sensed by s_i with a single round movement is at distance $d_{\mathtt{max}} + r_i = r_{tx}/2$ from the current position of s_i. The region $P(s_i)$ will be hereby referred shortly as the *curve polygon* of the sensor s_i. From the previous considerations we derive the following lemma:

Lemma 3. *$P(s_i) \cap P(s_j) = \emptyset$, for all $i, j = 1, \ldots, N$ with $i \neq j$.*

Figure 8 shows the construction of the sets $P(s_i)$ for three sensors, that correspond to the shaded areas surrounding the sensor positions.

The following lemma states that the covered points of $P(s_i)$ are certainly covered by the generating sensor s_i, and possibly by other sensors.

Lemma 4. *Let us consider N sensors s_i located at C_i with sensing radii r_i, $i = 1, \ldots, N$. Let $\mathscr{C}_i$ be the circle centered in C_i with radius r_i. For all $i, j = 1, \ldots, N$, $(P(s_i) \cap \mathscr{C}_j) \subseteq \mathscr{C}_i$.*

Proof: The curve polygon $P(s_i)$ may contain points that either belong to the real Voronoi polygon $V(\mathscr{C}_i)$ or are located outside $V(\mathscr{C}_i)$. For the points that belong to $V(\mathscr{C}_i)$, Theorem 3.1 holds and the property follows trivially. The other points are necessarily uncovered because they are located at a distance from any other sensor s_j with $j \neq i$, that is larger than $r_{tx}/2$, since $r_j < r_{tx}/2$, $\forall j = 1, \ldots, N$ by assumption (2) of section 4.1. □

Lemma 4 provides the ground for the proof of convergence and termination of VorLag.

Theorem 4.1. *The VorLag algorithm converges.*

Proof: Let $P^{(k)}(s_i)$ be the curve polygon $P(s_i)$ at the k-th round, and let us consider the following set family at round k:

$$\mathscr{P}^k \triangleq \{P^{(k)}(s_1), P^{(k)}(s_2), \ldots, P^{(k)}(s_N), \text{AoI} \backslash \cup_{i=1}^{N} P^{(k)}(s_i)\}.$$

Due to Lemma 3, $\mathscr{P}^k$ is a partition of the AoI. To make it easier to follow the proof, in Figure 8, we provide an example of the construction of the partition $\mathscr{P}^k$. The AoI is partitioned into four sets, the three shaded areas are assigned to the three sensors as their responsibility regions. The fourth element of the partition is constituted by the zones of the AoI that are not shaded in the figure.

Let $A_i^{(k)}$ and $A_i^{(k)}(s_i)$ be the covered area of $P^{(k)}(s_i)$, considering the coverage contribution of all the sensors and of the sole s_i, respectively. Let $\hat{A}_i^{(k)}$ be the covered area of $P_i^{(k)}$, considering the sensor positions at round $(k+1)$. In general $\hat{A}_i^{(k)}(s_i) \neq A_i^{(k+1)}(s_i)$, as they are calculated on different curve polygons. Finally, let $A_{total}^{(k)}$ be the total area covered by all sensors in the network at round k.

The element AoI $\backslash \cup_{i=1}^{N} P^{(k)}(s_i)$ of the partition $\mathscr{P}^k$ is only constituted by uncovered points that cannot be covered by any sensor with only a one-round movement. Hence this element does not contribute to $A_{total}^{(k)}$, that can be calculated as $A_{total}^{(k)} = \sum_{i=1}^{N} A_i^{(k)}$.

Due to Lemma 4, $A_i^{(k)} = A_i^{(k)}(s_i)$. According to the movement adjustment, if a sensor s_i moves then $A_i^{(k)}(s_i) < \hat{A}_i^{(k)}(s_i)$. Because other sensors can contribute to the coverage of $P^{(k)}(s_i)$ at round $(k+1)$, $\hat{A}_i^{(k)}(s_i) \leq \hat{A}_i^{(k)}$.

The previous inequalities imply that, if at least one sensor moves at round k:

$$A_{total}^{(k)} = \sum_{i=1}^{N} A_i^{(k)} < \sum_{i=1}^{N} \hat{A}_i^{(k)}. \tag{2}$$

As the total coverage at round $(k+1)$ only depends on the node positions and not on the partition in use:

$$\sum_{i=1}^{N} \hat{A}_i^{(k)} = A_{total}^{(k+1)}. \tag{3}$$

Therefore, by combining Equations 2 and 3 we can conclude that, if at least one sensor moves $A_{total}^{(k)} < A_{total}^{(k+1)}$.

Since the total coverage is upper bounded by the area of the AoI the algorithm converges. All the sensors stop moving when no coverage increase can happen. □

In theory, although the algorithm will converge, it may take progressively smaller infinitesimal steps towards convergence, and thus never terminate. We therefore introduce a positive *minimum movement threshold* ϵ.

Corollary 1. *The VorLag algorithm terminates if movements are allowed only if their extension exceeds a positive minimum movement threshold.*

The introduction of ϵ ensures fast termination and power saving, at the expense of a small loss in the coverage extension.

Theorem 4.2. *At any fixed round k, the complexity of the computation of a sensor s_i is $O(\Delta_k(s_i))$, where $\Delta_k(s_i)$ is the number of sensors that are in radio proximity to s_i.*

Proof: At each round k, the sensor s_i discovers all $\Delta_k(s_i)$ sensors to which it is within radio proximity and computes its Voronoi polygon in the Laguerre measure. This takes $O(\Delta_k(s_i))$ time. Then, s_i has to compute the miniMax point of its Voronoi polygon in order to move towards it. Megiddo [28] proposed a solution for the equivalent smallest enclosing circle problem which is linear in the number of points that must be included in the circle. It follows that, applying this algorithm, the processing time of each sensor s at round k is $O(\Delta_k(s))$. □

Observe that, if the starting configuration is a cluster of sensors, during the first rounds $\Delta_k(s_i)$ can be a $\Theta(N)$ but when the sensors distribute over the AoI with average density ρ, the number of sensors in radio proximity to s_i are in average $\rho\pi {r_i^{tx}}^2$, where r_i^{tx} is the transmission radius of sensor s_i.

5 The Voronoi Laguerre algorithm for a density driven deployment

The algorithm described in Section 4 finds a natural extension to the case of applications in which a deployment of homogeneous sensors is required at varying density over the AoI. We translate this problem into the problem of deploying heterogeneous sensors endowed with position dependent sensing radii, called *positional radii*, under uniform density requirements.

Given N homogeneous sensors, endowed with equal sensing radius R and transmission radius r_{tx}, we consider the position dependent density requirements expressed in terms of average distance from a sensor located in (x, y) to its neighbor nodes $d_{avg}(x, y)$. Since the optimal deployment consists in a regular hexagonal tiling, we can define the positional radius as $r(x, y) = d_{avg}(x, y)/\sqrt{3}$. VorLag can be used in this new operative context if we consider each of the uniform sensors as endowed with this positional radius.

It should be noticed that the use of positional radii to modulate the deployment density would not be possible under the ordinary Voronoi algorithm because of the barrier effect shown in Figure 2.

Recall that d_{max_i} is the maximum distance that the sensor s_i is allowed to traverse without overlapping the sensing range of other approaching sensors. As the positional radius of the sensor s_i varies with its own position, even the proper value of $d_{\mathtt{max}i}$ should be position dependent and calculated on the basis of the sensing radius that the sensor s_i would have at the movement destination. To simplify the calculation of $d_{\mathtt{max}i}$ we refer to the following formulation which does not depend on the particular sensor, nor on the destination coordinates: $d_{\mathtt{max}} = \frac{r_{tx}}{2} - \max_{(x,y)\in AoI} r(x, y)$.

Inasmuch as under the considered applicative scenario the sensing capability of each sensor varies with the sensor position, we need to modify the movement adjustment scheme accordingly.

We refer to the same notation introduced in Section 4.4, and in particular, we consider the curve polygon $P(s_i)$ generated by the intersection of the local polygon of sensor s_i, with the circle centered in its location and radius $r_{tx}/2$. Due to the new formulation of the maximum traversed distance per round $d_{\mathtt{max}}$, in this varying density scenario, $P(s_i)$ includes the area where the sensor s_i can improve the coverage in a single round.

Indeed, the farthest point that can be sensed by s_i with a single round movement leading it from (x_i, y_i) to (x'_i, y'_i), is at distance $d_{max} + r_i(x'_i, y'_i) \leq r_{tx}/2$ from the current position (x_i, y_i) of s_i. The sets $P(s_i)$, for $i = 1, \ldots, N$ are disjoint by definition.

We now introduce some further notation. Given a sensor s_i, with positional sensing range $\mathscr{C}_i$ centered in $\mathrm{C}_i = (x_i, y_i)$ with radius $r_i(x_i, y_i)$, we define the *weighted coverage* $w(P(s_i), \mathscr{C}_i)$ as follows:

$$w(P(s_i), \mathscr{C}_i) = \int_{P(s_i)\cap\mathscr{C}_i} \frac{R^2}{r^2(x, y)} dxdy.$$

More intuitively, we are introducing a local expansion of the AoI that is proportional to the local contraction in the sensing range due to the definition of the positional radius, where a higher density deployment is required. The new formulation of the movement adjustment condition is the following: at round k, a sensor s_i is allowed to move from the initial position (x_i, y_i) towards the destination (x'_i, y'_i) only if the movement increases the weighted coverage with respect to the current curve polygon $P^k(s_i)$. In other words, at the k-th round, s_i moves towards (x'_i, y'_i) if $w(P^k(s_i), \mathscr{C}_i) < w(P^k(s_i), \mathscr{C}'_i)$, where $\mathscr{C}'_i$ is the circle with center $C'_i = (x'_i, y'_i)$ and radius $r_i(x'_i, y'_i)$, and $w(P^k(s_i), \mathscr{C}'_i)$ is the value of the weighted coverage of the curve polygon $P^k(s_i)$ when s_i is located at C'_i.

Observe that if the sensor s_i in (x_i, y_i) moves towards a point (x'_i, y'_i) of the AoI requiring a smaller positional radius, i.e. $r(x_i, y_i) > r(x'_i, y'_i)$, some possibly useful movements could be impeded by the movement adjustment condition. For instance, if the distance between (x_i, y_i) and (x'_i, y'_i) is smaller than $r(x_i, y_i) - r(x'_i, y'_i)$, the sensing circle at the destination $\mathscr{C}'_i$ would be completely included in the sensing circle that s_i would have in the original position $\mathscr{C}_i$. In order to make the algorithm work in the described situations, we can impose the Lipschitz continuity of the function $r(x, y)$ with Lipschitz constant equal to 1, i.e. $|r(x_i, y_i) - r(x'_i, y'_i)| \leq |d_E((x_i, y_i), (x'_i, y'_i))|$.

Notice that Lipschitz continuity of the density requirement is not a necessary condition for the algorithm to work and terminate. Nevertheless, in order to allow VorLag to achieve the desired density, a non-continuous density requirement should be translated into a Lipschitz continuous function $r(x, y)$. This condition on $r(x, y)$ can be met by artificially smoothing the density requirement over the AoI allowing some over-provisioning of devices over the areas with non-uniform requirements.

In the following theorem we prove the termination of the VorLag algorithm in the context of variable density requirements.

Theorem 5.1. *Given a variable density requirement expressed in terms of positional sensing radius $r(x, y)$ defined over the AoI, the VorLag algorithm is convergent, and thereby terminates provided a positive minimum movement threshold.*

Proof: In this proof we use the following simplified notation: $W_i^{(k)}(s_i) \triangleq w(P^k(s_i), \mathscr{C}_i^k)$ is the weighted coverage at round k of the curve polygon $P^k(s_i)$ due to the sensor s_i, while $W_i^{(k)} \triangleq \sum_{j=1}^{N} w(P^k(s_i), \mathscr{C}_j^k)$ is the weighted coverage of $P^k(s_i)$ due to all the sensors at round k.

Thanks to the fact that the sets $P^k(s_i)$, $i = 1, \ldots, N$, are disjoint, we can create a partition of the AoI as follows:

$$\mathscr{P} \triangleq \{P^k(s_1), P^k(s_2), \ldots, P^k(s_N), \text{AoI} \setminus \cup_{i=1}^{N} P^k(s_i)\}.$$

The partition element $\text{AoI} \setminus \cup_{i=1}^{N} P^k(s_i)$ is made by points that are currently uncovered and so they will remain at the next step of the algorithm. This is due to the limitation to the distance that a sensor can traverse in a single round (d_{max}). Therefore, the total coverage $W_{total}^{(k)}$ can be calculated by summing up the coverage of the first N parts of $\mathscr{P}^k$, hence $W_{total}^{(k)} = \sum_{i=1}^{N} W_i^{(k)}$.

Due to Lemma 4 the coverage of the partition element $P^k(s_i)$ at round k is $W_i^{(k)} = W_i^{(k)}(s_i)$. It follows that $W_{total}^{(k)}$ can be calculated by summing the coverage that each sensor contributes in its curve polygon specifically, thence $W_{total}^{(k)} = \sum_{i=1}^{N} W_i^{(k)}(s_i)$.

Let $\hat{W}_i^{(k)}$ be the weighted coverage of $P_i^{(k)}$ at the $(k+1)$-th round, that is by considering the k-th round curve polygon and the coverage contribution due to all the sensors in their $(k+1)$-round positions. Let also $\hat{W}_i^{(k)}(s_i)$ be the weighted coverage of $P_i^{(k)}$ specifically due to sensor s_i in its $(k+1)$-th round position.

According to the movement adjustment adapted to the case of variable density requirements, if a sensor s_i moves then $W_i^{(k)}(s_i) < \hat{W}_i^{(k)}(s_i)$. Because other sensors can contribute to the coverage of $P^{(k)}(s_i)$ at round $(k+1)$, $\hat{W}_i^{(k)}(s_i) \leq \hat{W}_i^{(k)}$. The previous inequalities imply that, if at least one sensor moves at round k:

$$W_{total}^{(k)} = \sum_{i=1}^{N} W_i^{(k)} < \sum_{i=1}^{N} \hat{W}_i^{(k)}. \tag{4}$$

As the coverage at round $(k+1)$ only depends on the node positions and not on the partition in use, even the weighted coverage does not depend on the partition:

$$\sum_{i=1}^{N} \hat{W}_i^{(k)} = W_{total}^{(k+1)}. \tag{5}$$

Therefore, by combining Equations (4) and (5) we can conclude that if at least one sensor moves then $W_{total}^{(k)} < W_{total}^{(k+1)}$, hence the total coverage increases at each round.

Since the total coverage is upper bounded by $W_{\mathsf{max}} = \int_{AoI} \frac{R^2}{r^2(x,y)} dxdy$ the algorithm converges. All the sensors stop moving when no coverage increase can occur. By introducing a minimum movement threshold we can conclude that the algorithm terminates in a finite number of rounds. □

Notice that, a similar approach can be introduced to address the case of position dependent accuracy requirements in the heterogeneous setting. In this case, it is not possible to formulate the accuracy requirements in terms of density, hence the reduction in the sensing range should be formulated for each individual device according to its specific sensing accuracy which varies with the distance from the monitored area.

6 A VIRTUAL FORCE APPROACH FOR THE DEPLOYMENT OF HETEROGENEOUS SENSORS

In order to evaluate the performance of the VorLag algorithm, we introduce another solution to which we will refer in the Section 7 devoted to experiments.

Among the deployment algorithms proposed for mobile sensors in the literature, very few specifically address the problem of sensor heterogeneity, and none are suitable for our applicative context due to the many assumptions they introduce. This is the reason why we prefer to introduce some modifications to a recently proposed approach to area coverage based on the virtual force technique, called Parallel and Distributed Network Dynamics (PDND), proposed in [2]. In PDND the force exerted by s_i to sensor s_j is modelled as a piecewise linear function. Therefore it is repulsive when the distance between s_i and s_j is lower than an arbitrarily tuned parameter r^*; it is attractive when the distance is larger, until it vanishes at another arbitrarily set distance. The formulation of this force respects the condition of Lipschitz continuity that is necessary to ensure the convergence of PDND. The single sensor movement is limited by a dynamically set upper bound that guarantees that the potential energy is always decreasing, hence avoiding oscillations.

PDND works under the assumption that sensors are homogeneous in terms of sensing and communication capabilities. In order to make the algorithm feasible for heterogeneous sensors, we need to redefine the force that one sensor exerts on the others. According to the algorithm PDND, this implies the definition of the rest distance r^* at which the force exerted by two interacting sensors is null.

We propose a setting of r^* that minimizes the overlaps among neighbor sensors, hence $r^* = r_i + r_j$; that is two sensors try to position themselves so that their sensing circles are tangential.

The modification of PDND that deals with the case of heterogeneous sensors proposed above, can be applied as well to the problem of deploying sensors at varying density by employing the notion of position dependent sensing radius introduced in Section 5.

7 Experimental results

In order to evaluate the performance of VorLag, we compare it with its traditional counterpart (Vor) and with the algorithm PDND described in Section 6. We developed a simulator on the basis of the wireless module of the OPNET environment [29].

We ran three sets of experiments. In the first set we study the deployment of heterogeneous sensors. In the second we consider varying density requirements within the AoI. In the last set, we consider time-varying density requirements due to dynamic missions.

We perform these experiments to quantify the differences in performance and behavior of VorLag, Vor and PDND caused by a fundamental difference in the algorithms: VorLag and Vor specifically target a coverage density and then terminate, while PDND terminates only when its forces are balanced. We expect that Vor will terminate without meeting the coverage requirements and that VorLag will converge faster than PDND, using less energy.

In all of the following experiments, we consider a random deployment of sensors in an AoI of 80m $\times$ 80m. We set the communication radius of all sensors r_{tx} to 11m and the sensor movement speed to 1m/sec. For PDND we use the settings proposed in [2]: the length of a round is 1sec, and the minimum movement threshold $\epsilon_{\mathtt{PDND}}$ is 0.1m. Notice that the setting of the round length of PDND is particularly critical, because the value results in a trade-off of high communication overhead and energy expenditure (shorter rounds) with longer convergence times (longer rounds). For both VorLag and Vor, the duration of a round is determined by the distance $\max_i d_{\mathtt{max}i}$ and by the sensor speed. The plotted results are obtained by averaging the values of 80 simulation runs.

7.1 Heterogeneous sensors

In this set of experiments we consider a network composed of heterogeneous sensors. Hence, we set the radius of each sensor to a random value that is either 2m or 5m.

Figure 9(a) shows an example of initial deployment obtained with 250 sensors. Figures 9(b), 9(c) and 9(d) show the final deployments achieved by VorLag, Vor, and PDND, respectively.

We now compare the algorithms in terms of deployment performance (e.g. coverage and termination time) as well as energy metrics (e.g. average traversed distance and total consumed energy). In the experiments we progressively increase the number of deployed sensors from 50 to 500.

Figure 10(a) shows the percentage of area that is covered at the end of the deployment phase. As expected, the coverage increases with the number of sensors and is complete with a sufficient numbers of sensors with VorLag and PDND, whereas Vor never achieves a complete coverage in the simulated settings. As Figure 10(b) shows, Vor terminates earlier than the other two

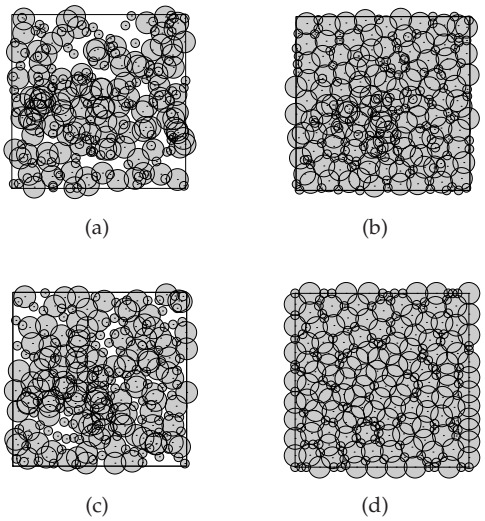


Fig. 9: Random initial deployment of 250 heterogeneous sensors (a). Final deployment obtained with VorLag (b), Vor (c) and PDND (d).

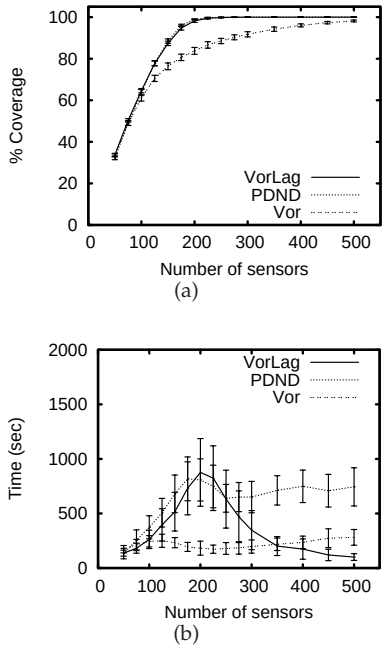


Fig. 10: Percentage of covered AoI (a), completion time (b).

algorithms in the majority of cases. It should be noted that this fast termination denotes that Vor is not moving the sensors sufficiently to achieve a good coverage. By contrast, VorLag and PDND require a similar amount of time to reach the final coverage when few sensors are available. In both cases, when the number of deployed sensors is lower than the minimum required to achieve full coverage, all the available sensors are necessary to expand the network coverage. Hence the deployment time grows as the number of sensors increases. This trend changes to the opposite when the number of sensors exceeds the minimum necessary. In this last case, as the number of sensors increases, VorLag requires significantly less time to achieve the final deployment,

because with more sensors the initial random deployment results in a higher coverage and less movements are needed to complete the coverage. Therefore VorLag terminates earlier than PDND because it stops as soon as the area is completely covered. On the contrary, PDND requires more time because of the nature of the virtual forces: it is more time and energy consuming to find the force balance when there are more sensors available. For example, with 450 sensors, VorLag converges in about 17% of the time required by PDND.

We now compare the performance of the algorithms in terms of energy consumption. A sensor consumes energy as a consequence of movements (start/stop and traversed distance [30]) and communications (sending, listening and receiving messages). Figure 11(a) shows the average distance traversed by the sensors. Vor does not achieve a complete coverage, hence with this algorithm the distance traversed by the sensors is relatively small. Both Vorlag and PDND show an increasing traversed distance when the number of sensors is not sufficient to complete the coverage of the AoI. Since VorLag stops its movement as soon as the area is completely covered, the traversed distance decreases as the number of sensors increases. This is also true for PDND, although the sensors traverse longer distances with respect to VorLag. The average number of starting and stopping actions is shown in Figure 11(b). PDND performs a higher number of starting/stopping actions with respect to VorLag. This is due to the short distance that a sensor moves in each round. According to the definition of PDND, in order to guarantee the convergence of the algorithm, once a sensor has calculated the resulting force, it is allowed to move in the direction of the force for a short distance that guarantees a decrease of the potential energy in that direction. VorLag, on the other hand, makes longer (but still accurate) movements, resulting in a much lower number of starting stopping actions. Also in this case, Vor causes a lower amount of starting/stopping actions with respect to the other algorithms. This apparently good behavior of Vor is instead motivated by its incapability to meet the requirements on coverage. Figure 11(c) shows the unified energy consumption, which includes the energy spent by the sensors for communications and movements, expressed in energy units (eu). The reception of one message corresponds to 1 eu, a single transmission costs 1.2 eu, a 1 meter movement costs 340 eu as a single starting or stopping action [31].

Because the energy spent for communication is generally much smaller than for movements, the energy consumption is dominated by the traversed distance and the number of starting/stopping actions. VorLag consumes less energy than PDND, whose performance is affected by the high number of starting and stopping actions. Moreover VorLag shows once again a good scalability with the number of sensors, as it requires less energy when the number of available sensors grows. Once more, even though Vor consumes a smaller amount of energy than the other two algorithms, it is so because it does not meet the requirements.

Until now we have assumed that sensors have perfect knowledge of their sensing range. If the sensors have a known bounded imprecision, this can be taken into account simply by assuming that sensors work at a virtual radius which is the minimum in their uncertainty interval. VorLag will still guarantee the coverage completeness if the number of sensors is slightly over provisioned.

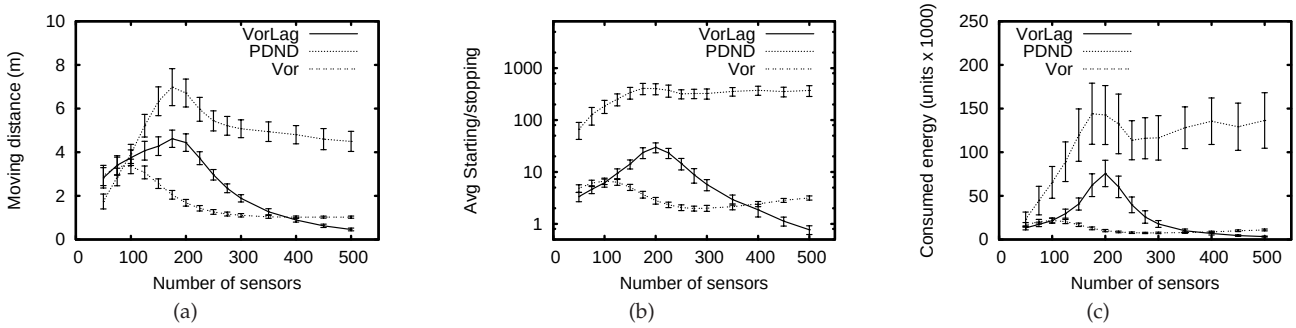
In cases in which the sensors are not aware of their reduced sensing capabilities, both the VorLag and PDND may leave some coverage holes. In Figure 12 we show the amount of coverage achieved by the two algorithms when a certain percentage of the 350 available devices are defective, i.e. have a reduction of 10% of their nominal sensing range. Both the algorithms show a small loss in coverage when the percentage of defective sensors increases. This loss is slightly less with PDND because PDND tends to result in a uniform deployment. Note, however, that PDND requires one order of magnitude more energy than VorLag just to achieve no more than 1% of gain in coverage, as shown in Figure 12.

We conclude this subsection by considering the applicability of these algorithms in the presence of packet losses with probability $p_{\texttt{loss}}$. In particular we consider the cases of no losses $p_{\texttt{loss}} = 0$ and $p_{\texttt{loss}} = 0.4\%$. Both VorLag and PDND are very resilient to packet losses, as shown in Figure 13(a) which shows the coverage achieved over time. Nevertheless, Figure 13(b) evidences that in the case of packet loss the gap between PDND and VorLag in terms of average traversed distance increases with time. This means that VorLag is more attractive than PDND in realistic scenarios where communications are error prone.

7.2 Density driven deployment

In the second set of experiments we consider the problem of covering an AoI with varying density. In these experiments, we assume that all sensors are homogeneous, i.e. have the same sensing radius r_s of 5m.

Figure 14(a) depicts a scenario with 350 sensors randomly placed over an AoI where a circular zone around the epicenter of an event of interest requires more sensors than the rest of the area. The density requirement inside the central high density zone is 0.08 sensors per unit area, corresponding to a position dependent radius of about $r_{high} = 2$m, while the rest of the area only requires coverage, hence a position dependent radius of $r_{low} = 5$m. The smaller circles around the sensors located inside the higher density disk represent the lower position-dependent radii. We recall that the Lipschitz continuity of the position dependent sensing radius is required for VorLag to improve the coverage capabilities of the algorithm, as explained in Section 5. Likewise, this requirement implies the Lipschitz continuity of the force function that is necessary for PDND to terminate, as explained in Section 6. For this reason, we consider a linear



(a) (b) (c)

Fig. 11: Average traversed distance (a), average number of start/stop actions (b), average consumed energy (c).

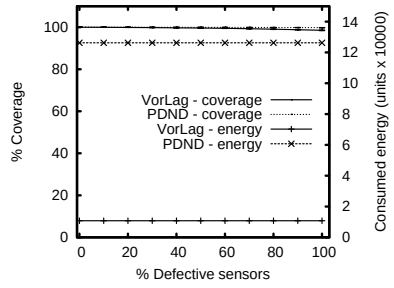
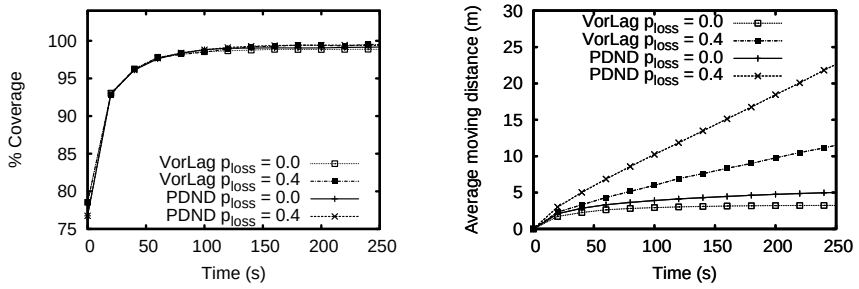
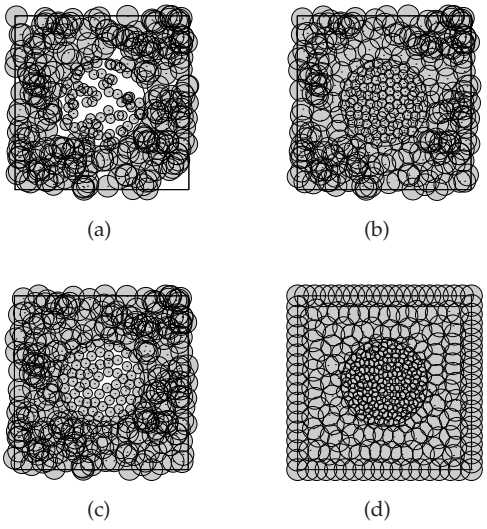


Fig. 12: Coverage and energy consumption varying the percentage of defective sensors.



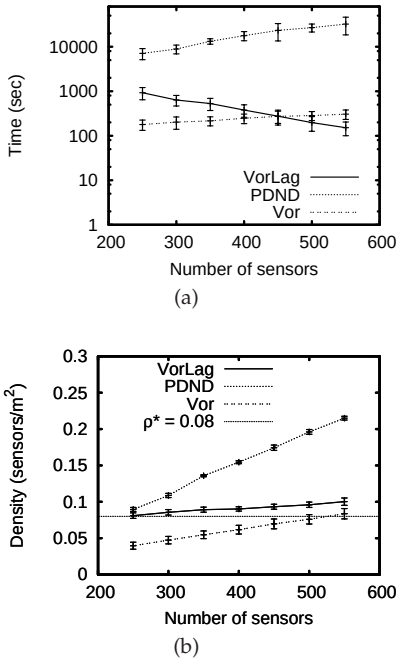
(a) (b)

Fig. 13: Performance of VorLag and PDND in the presence of packet losses: Coverage percentage (a) and average traversed distance (b).



(a) (b) (c) (d)

Fig. 14: An example of random initial deployment of 350 sensors (a), final deployment achieved by VorLag (b), Vor and (d) PDND (c).

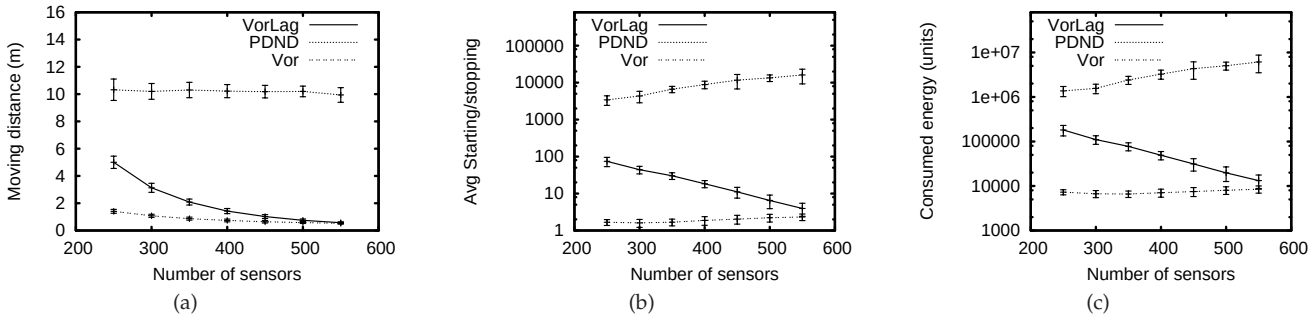


(a) (b)

Fig. 15: Completion time (a), density of the event area (b).

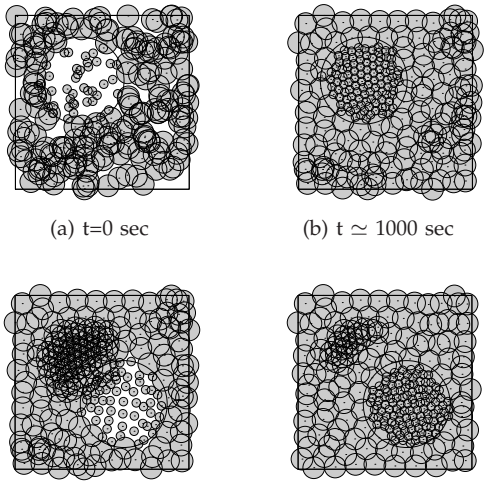
interpolation of the position dependent sensing radius in a circular crown containing the border of the high density zone, so as to ensure a smooth variation. The width of such a crown is set to $r_{high} + r_{low} = 7$m. Figure 14 shows the final deployment achieved by VorLag (b), Vor (c) and PDND (d). Notice the poor performance of Vor, due to the barrier effect detailed in Section 3.

Figure 15(a) shows the termination time of the algorithms. VorLag outperforms PDND. As we already mentioned for the previous set of experiments, although in some scenarios Vor terminates earlier than VorLag, this is only due to its incapability to reach the complete coverage. Figure 15(b) shows the density achieved inside the event area, and the desired density is represented by the horizontal line at $\rho^* = 0.08$. PDND is incapable



(a) (b) (c)

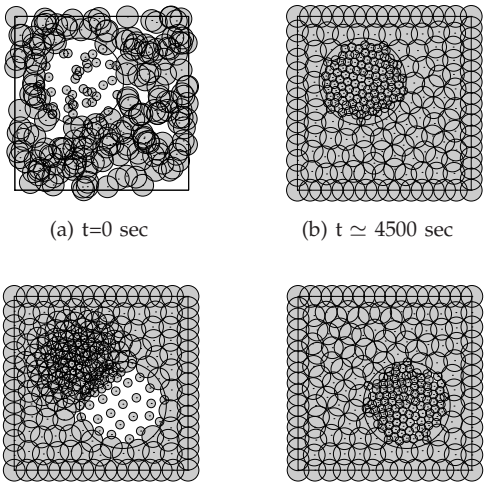
Fig. 16: Average traversed distance (a), average number of starting/stopping (b), average consumed energy (c).



(a) t=0 sec (b) t $\simeq$ 1000 sec

(c) t=7200 sec (d) t $\simeq$ 9000 sec

Fig. 17: VorLag with dynamic missions.



(a) t=0 sec (b) t $\simeq$ 4500 sec

(c) t=7200 sec (d) t $\simeq$ 14000 sec

Fig. 18: PDND with dynamic missions.

of controlling the density around the epicenter and achieves density which is higher than necessary when the number of sensors increases. By contrast, VorLag succeeds in achieving the required density without unnecessary movements. Vor is not capable of meeting the density requirements, except in the case with about 550 sensors, where the requirement is already met by the initial deployment.

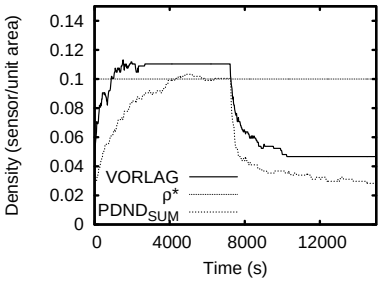
In Figure 16(a) we show the average traversed distance during the deployment phase. VorLag shows the best behavior among the three algorithms, allowing sensors to complete the deployment (which is not done by Vor) by traversing shorter distances with respect to PDND. Figures 16(b) and 16(c) show the average number of starting/stopping actions and the average consumed energy, respectively. These results are explained in a similar way to those presented in Section 7.1.

7.3 Dynamic mission arrival

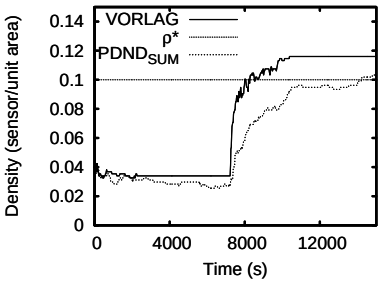
A mission is seen as a change in the coverage requirements over the AoI. Such a change aims at making the sensor density high around the epicenter of the mission, in order to have a better detection of the ongoing event. The density of sensors around a mission is set to 0.1 sensors per unit area. In these experiments we focus on the algorithm self-reconfiguration capabilities, i.e. on the time to relocate sensors in order to fulfill dynamic density requirements. We omit comparisons with Vor as we already pointed out its incapability to face variable density requirements in the previous set of experiments.

Figure 17(a) shows the initial deployment with 300 randomly placed sensors. The disk located at the top left of the AoI represents a first mission, known at the beginning of the deployment. Figures 17(b) and 18(b) show the deployment achieved by VorLag and PDND for the first mission, respectively. Once the first mission has ended a second one appears at time t=7200 seconds, with the same density requirement, but located on the bottom right corner of the AoI, as depicted in Figure 17(c) and 18(c), and requires the network to reconfigure. Figure 17(d) and 18(d) show the final deployments achieved by VorLag and PDND respectively.

In Figures 19(a) and (b) we show the density over time of the first and second mission area respectively. VorLag shows a shorter configuration and reconfiguration time with respect to PDND, although PDND shows a more regular deployment. Note that in this context the goal of



(a)



(b)

Fig. 19: Density of the first mission (a) and second mission (b) area.

the algorithm is to achieve a minimum required density on the mission locations and simple coverage elsewhere. Once this is achieved, there is no reason to move sensors in order to make the deployment more regular. The VorLag algorithm achieves its goal faster than PDND, whereas PDND wastes energy with useless movements creating a regular deployment as side effect.

8 Conclusions

We introduced a novel algorithm, called VorLag, to deploy heterogeneous mobile sensors over a field of interest. The VorLag algorithm works even with time and position dependent density requirements. We formally proved the termination of VorLag under both the applicative scenarios. The proposed algorithm is self-adaptive to runtime changes of the operative conditions. It requires no prior knowledge of the applicative scenario, and does not necessitate manual tuning of its working parameters. We compared VorLag with its traditional counterpart, showing that it resolves stale and critical situations. Furthermore, we introduced modifications to a virtual force based approach in order to make it work in the heterogeneous and variable density scenario. Extensive simulations show that our algorithm outperforms the other approaches in many respects.

References

[1] G. Wang, G. Cao, and T. La Porta, "Movement-assisted sensor deployment," *IEEE Trans. on Mobile Computing*, vol. 6, pp. 640–652, 2006.

[2] K. Ma, Y. Zhang, and W. Trappe, "Managing the mobility of a mobile sensor network using network dynamics," *IEEE Trans. on Parallel and Distributed Systems*, vol. 19, no. 1, pp. 106–120, 2008.

[3] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," *Proc. of DARS*, pp. 299–308, 2002.

[4] M. Garetto, M. Gribaudo, C.-F. Chiasserini, and E. Leonardi, "A distributed sensor relocation scheme for environmental control," *The ACM/IEEE Proc. of MASS*, pp. 1–10, 2007.

[5] J. Chen, S. Li, and Y. Sun, "Novel deployment schemes for mobile sensor networks," *Sensors*, vol. 7, pp. 2907–2919, 2007.

[6] N. Heo and P. Varshney, "Energy-efficient deployment of intelligent mobile sensor networks," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 35, pp. 78–92, 2005.

[7] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *ACM Trans. on Embedded Computing Systems*, vol. 3, no. 1, pp. 61–91, February 2003.

[8] S. Poduri and G. S. Sukhatme, "Constrained coverage for mobile sensor networks," *Proc. of IEEE ICRA*, pp. 165–171, 2004.

[9] M. R. Pac, A. M. Erkmen, and I. Erkmen, "Scalable self-deployment of mobile sensor networks; a fluid dynamics approach," *Proc. of IEEE IROS*, pp. 1446–1451, 2006.

[10] M. Ma and Y. Yang, "Adaptive triangular deployment algorithm for unattended mobile sensor networks," *IEEE Trans. on Computers*, vol. 56, pp. 946–847, 2007.

[11] 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, no. 6, pp. 836–848, 2009.

[12] N. Bartolini, T. Calamoneri, E. Fusco, A. Massini, and S. Silvestri, "Autonomous deployment of self-organizing mobile sensors for a complete coverage," *ACM/Springer Wireless Networks*, vol. 16, no. 3, pp. 607–625, 2010.

[13] X. Li, H. Frey, N. Santoro, and I. Stojmenovic, "Focused-coverage by mobile sensor networks," *The IEEE Proc. of MASS*, 2009.

[14] A. Bar-Noy, T. Brown, M. P. Johnson, and O. Liu, "Cheap or flexible sensor coverage," *Distributed Computing in Sensor Systems, Lecture Notes in Computer Science*, vol. 5516, pp. 245–258, 2009.

[15] M. P. Johnson, D. Sar, A. Bar-Noy, T. Brown, D. Verma, and C. W. Wu, "More is more: the benefits of denser sensor deployment," *The ACM/IEEE Proc. of INFOCOM*, 2009.

[16] C.-Y. Chang, C.-T. Chang, Y.-C. Chen, and H.-R. Chang, "Obstacle-resistant deployment algorithms for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 6, pp. 2925–2941, 2009.

[17] S. Martinez, J. Cortes, and F. Bullo, "Motion coordination with distributed information," *IEEE Control Systems Magazine*, pp. 75–88, August 2007.

[18] X. Wu, G. Chen, and S. K. Das, "On the energy hole problem of nonuniform node distribution in wireless sensor networks," *Proc. of IEEE MASS*, pp. 180–187, 2006.

[19] M. Cardei, Y. Yang, and J. Wu, "Non-uniform sensor deployment in mobile wireless sensor networks," *Proc. of WoWMoM*, pp. 1–8, 2008.

[20] J. Li and P. Mohapatra, "Analytical modeling and mitigation techniques for the energy hole problem in sensor networks," *Pervasive and Mobile Computing*, no. 3, pp. 233–254, 2007.

[21] S. Olariu and I. Stojmenovic, "Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting," *Proceedings of INFOCOM*, pp. 1–12, 2006.

[22] M. Lam and Y. Liu, "Two distributed algorithms for heterogeneous sensor network deployment towards maximum coverage," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3296–3301, 2008.

[23] A. Gusrialdi, S. Hirche, T. Hatanaka, and M. Fujita, "Voronoi based coverage control with anisotropic sensors," *Proc. of the American Control Conference (AAC)*, 2008.

[24] H. Ammari and S. Das, "Promoting heterogeneity, mobility, and energy-aware voronoi diagram in wireless sensor networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 19, no. 7, pp. 995–1008, 2008.

[25] F. Aurenhammer, "Voronoi diagrams - a survey of a fundamental geometric data structure," *ACM Computing Surveys*, vol. 23, no. 3, pp. 345–405, 1991.

[26] H. Imai, M. Iri, and K. Murota., "Voronoi diagram in the Laguerre geometry and its applications," *SIAM J. Comput.*, vol. 14, no. 1, pp. 93–105, 1985.

[27] K. Sugihara, "Laguerre voronoi diagram on the sphere," *Journal for Geometry and Graphics*, vol. 6, no. 1, pp. 69–81, 2002.
[28] N. Megiddo, "Linear-time algorithms for linear programming in $\Re^3$ and related problems," *SIAM J. Comput.*, vol. 12, pp. 759–776, 1983.
[29] "Opnet technologies inc." *http://www.opnet.com*.
[30] G. Sibley, M. Rahimi, and G. Sukhatme, "Mobile robot platform for large-scale sensor networks," *IEEE International Conference on Robotics 158 and Automation (ICRA)*, pp. 1143–1148, 2002.
[31] G. Anastasi, M. Conti, A. Falchi, E. Gregori, and A. Passarella, "Performance measurements of mote sensor networks," *Proc. of ACM MSWiM 2004*, pp. 174–181.

Novella Bartolini graduated with honors in 1997 and received her PhD in computer engineering in 2001 from the University of Rome, Italy. She is now assistant professor at the University of Rome. She was researcher at the Fondazione Ugo Bordoni in 1997, visiting scholar the University of Texas at Dallas in 1999-2000 and research assistant at the University of Rome 'Tor Vergata' in 2000-2002. She was program chair and program committee member of several international conferences. She has served on the editorial board of Elsevier Computer Networks and ACM/Springer Wireless Networks. Her research interests lie in the area of wireless mobile networks and web based systems.

Tiziana Calamoneri graduated in Mathematics in 1992 and got her Ph.D. in Computer Science in 1997, at University of Rome "La Sapienza," Italy. Since 2006, she is associate professor at the Department of Computer Science, University of Rome "La Sapienza" where she also was assistant professor from 2000 to 2006. Her research interests include sensor networks, parallel and sequential graph algorithms, layout of networks topologies and optimal routing schemes, channel assigment in wireless networks.

Thomas F. La Porta is a Distinguished Professor in the department of computer science and engineering at Penn State University. He joined Penn State in 2002. At Penn State, Dr. La Porta is the Director of the Networking and Security Research Center. Prior to joining Penn State, he was with Bell Laboratories since 1986. There he was the Director of the Mobile Networking Research Department where he led various projects in wireless and mobile networking. He is an IEEE Fellow, Bell Labs Fellow, received the Bell Labs Distinguished Technical Staff Award in 1996, and an Eta Kappa Nu Outstanding Young Electrical Engineer Award in 1996. He also won Thomas Alva Edison Patent Awards in 2005 and 2009. Dr. La Porta received his B.S.E.E. and M.S.E.E. degrees from The Cooper Union, New York, NY, and his Ph.D. degree in Electrical Engineering from Columbia University, New York, NY. He was the founding Editor-in-Chief of the IEEE Transactions on Mobile Computing, and served as Editor-in-Chief of IEEE Personal Communications Magazine for three years. He has published over 150 technical papers and holds over 30 patents. His research interests include mobility management, signaling and control for wireless networks, mobile data and sensor systems, and network security.

Simone Silvestri graduated with honors in 2005 and is currently a PostDoc researcher in Computer Science at Sapienza, University of Rome, Italy. He was a visiting scholar at the Electrical and Electronic Engineering Department, Imperial College, London. He served as program committee member of several international conferences. His research interests lie in the area of sensor networks and web based systems.