2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

A Nonlinear Optimization Approach to Coverage Problem in Mobile Sensor Networks

Jalal Habibi, Hamid Mahboubi and Amir. G. Aghdam

Abstract— **A distributed Voronoi-based sensor deployment approach is proposed to optimize sensor network coverage. It is assumed that each sensor can construct its Voronoi polygon using the information it receives from the neighboring sensors. To increase the local coverage of the sensors, it is desired to find a point in each polygon, such that if the corresponding sensor is placed there, then its covered area within the polygon is maximized. A nonlinear optimization algorithm is proposed based on the gradient projection to find the optimal location for each sensor. The algorithm can be implemented in a distributed fashion with minimum communication among the sensors. Examples are provided to demonstrate the effectiveness of the proposed approach in terms of convergence rate, coverage performance, and energy consumption.**

I. Introduction

The past decade has seen significant advances in micro-electromechanical systems (MEMS) technology. This in turn has enabled the development of highly reliable low-cost sensor networks for a broad range of applications. Some examples of recent applications of this type of network include traffic surveillance, intrusion detection, environmental monitoring and object tracking, to name only a few [1], [2], [3], [4].

Coverage optimization is an important problem in mobile sensor networks, which has been addressed by researchers in various fields of science and engineering [5], [6]. This problem is concerned with maximizing the area covered by the network, while some important constraints such as energy efficiency and limited communication among the sensors [7], [8]. In other words, it is desired to develop distributed deployment algorithms to move the sensors in such a way that the total coverage of the network is maximized, subject to some constraints on the energy consumption. Additional constraints such as collision and obstacle avoidance and may be introduced, depending on the specifics of a particular application. Furthermore, no *a priori* knowledge of the initial positions of the sensors may be available [9], [10].

In [11] an approach called *basic protocol* is introduced to find the position of each sensor iteratively such that the total coverage is maximized. An energy efficient strategy called *virtual movement protocol* is proposed in [12] which takes the communication cost into account before the sensors are moved. In both techniques Voronoi diagram [13], [14] is used to find the coverage holes. Three distributed self-deployment strategies, namely VEC (vector-based), VOR (Voronoi-based), and minimax are presented in [12] to find the proper position for each sensor. In the VOR strategy, the desired location for each sensor is calculated based on its distance from the vertices of the corresponding Voronoi polygon. The algorithm performs poorly when a sensor is located close to a narrow edge in its Voronoi polygon. The VEC algorithm, on the other hand, attempts to move the sensor in such a way that they are evenly distributed in the sensing field. This algorithm may not perform satisfactorily for a network with a large number of sensors. In the Minimax strategy the location of each sensor in its Voronoi polygon is obtained such that its maximum distance from the vertices of the polygon is minimized. The shortcoming of this algorithm is similar to that of the VOR strategy.

A distributed Voronoi-based coverage maximization strategy is proposed in the present article which is entitled as *Max-Area*. The main contribution of the proposed algorithm is that in order to maximize the total coverage, each sensor at each round maximizes its own coverage in its Voronoi cell. Finding the optimum point inside each polygon is a complicated task and is solved here using some approaches form nonlinear optimization theory.

The paper is organized as follows. Some preliminaries and problem statement is outlined in the next section. As a main element of proposed coverage optimization algorithm, so-called sensor location problem is introduced in section III. Detailed nonlinear optimization approach to sensor location problem is presented in section IV. The overall multi-sensor coverage optimization algorithm is described in section V along with presentation of simulation results. Section VI concludes the paper.

II. Preliminaries

The sensing field is assumed to be a flat two-dimensional space with known boundaries and no obstacle. Furthermore, it is assumed that initially the sensors are randomly located in the field. Let the sensing field be denoted by F_S, and the sensors by S_i, $i \in \mathbf{n} := \{1,\dots,n\}$, with equal sensing range of R_s and communication range of R_c. A general point in $\mathbb{R}^2$ is represented by $q = [q_1, q_2]^T$. The position of the sensor S_i is denoted by x_{S_i} with the corresponding sensing disk of $D(x_{s_i})$.

The Voronoi polygon (or cell) of the sensor S_i is defined as $\Pi_i = \{q \in \mathbb{R}^2 : d(q,x_{s_i}) \leq d(q,x_{s_j}), \forall j \in \mathbf{n}, j \neq i\}$ where $d(.,.)$ denotes the euclidean distance between two points.

This work has been supported by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN-262127-07.

J. Habibi, H. Mahboubi, A. G. Aghdam are with the Department of Electrical and Computer Engineering, Concordia University, 1455 de Maisonneuve Blvd. W., EV005.139, Montréal, Québec, H3G 1M8, CANADA. {jalal,h_mahbo,aghdam}@ece.concordia.ca

The union of n Voronoi polygons $\Pi_1,\ldots,\Pi_n$ constitutes a diagram called the Voronoi diagram.

The Voronoi decomposition in a sensor network, is used to assign to each sensor a region in the field to monitor. It is important to note that any sensor in the network can construct its own Voronoi polygon using the information about the location of itself and its neighbors in the field. This means that any sensor deployment strategy based on the Voronoi diagram can be implemented in a distributed manner with limited information available for each agent about the whole network.

Distributed Voronoi-based coverage improvement optimization approaches in mobile sensor networks can, in general, be described in an abstract simplified form in the sequel.

General Procedure for Voronoi-based Coverage Improvement

1) Based on the information received from neighboring sensors, each sensor S_i constructs its own Voronoi polygon Π_i.
2) Each sensor calculates the area it covers in its own Voronoi polygon.
3) Using a proper destination point selection strategy, each sensor finds a candidate point inside its own Voronoi polygon as its next position.
4) Each sensor calculates the area it would cover if it moved to the new position. If the covered area from the new position would be more than the current covered area by a prespecified amount (e.g. 1 %), the sensor moves to the new location. If not, it remains in its current position.
5) If any of the sensors in the previous step moves to a new point, the algorithm repeats from step 1. If not, then every sensor has already reached a stable point, and the algorithm is terminated. ■

It is shown in [12] that the total coverage of mobile sensors is convergent, given the locally increasing coverage of each sensor. The coverage optimization methods which follow the above-mentioned framework differ mainly in the destination point selection strategy for each sensor (step 3). In fact, network coverage performance highly depends on the deployment strategy for relocating individual sensors.

The problem statement will be defined in the next section.

III. Optimal sensor location (OSL) problem

Consider a sensor S located at x_s, and let it have a disk-shaped sensing domain $D(x_s)$ with radius R_s. Assume that a density function $\varphi(q)$ is given over the corresponding convex polygon Π, to determine coverage priority for different points in the polygon (and eventually in the field). In other words, the density function specifies the relative importance of coverage for different points inside Π, and is considered as a weight function over the polygon.

The OSL problem looks for an optimal point inside Π, such that if the sensor moves to that point, its weighted covered area inside Π would be maximized. This can be formulated as the following optimization problem:

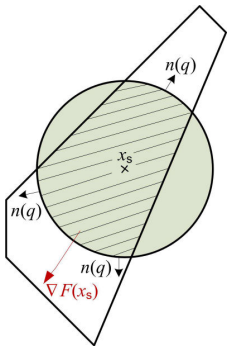

Fig. 1: Geometrical interpretation of the optimal sensor location problem.

$$x_s^* = \arg\max_{x_s\in\Pi} F(x_s) = \arg\max_{x_s\in\Pi} \int_{\Pi\cap D(x_s)} \varphi(q)dq \qquad (1)$$

A geometrical representation of the optimal sensor location problem is demonstrated for a simple setup in Fig. 1. It is desired in this figure to maximize the hatched region by properly locating the center of the disk (x_s) inside the polygon. The vectors $n(q)$ and $\nabla F(x_s)$ are important elements of the proposed optimization algorithm in this paper and will be referred later.

The problem introduced above is, in fact, a nonlinear optimization problem with linear constraints (imposed by the edges of the polygon). Note that in general it is very difficult to express the objective function as an explicit function of the elements of x_s (i.e., decision variables). Even the computation of the objective function for a specific value of x_s requires double integration over a convex region, which is simple but time-consuming. The limited computational capability of the sensors should be taken into account in the design and implementation of any deployment strategy for distributed mobile sensing agents.

It can be observed that the solution to the sensor location problem is obvious for some extreme cases. Let the center and radius of the smallest enclosing disk of a polygon be denoted by x_{enc} and R_{enc}, respectively. The center of the largest inscribed disk inside a polygon is known as the Chebyshev center of the polygon and can be computed efficiently, e.g., by solving a simple linear programming problem [15]. Let the center and radius of this circle (also shown in Fig. 2), be denoted by x_{cheb} and R_{cheb}, respectively. Obviously, x_{cheb} is the solution to the OSL problem if the density function is uniform over the polygon and the sensing range is less than or equal to R_{cheb}. However, for the cases where $R_{cheb} < R_s < R_{enc}$ the solution is not straightforward, and a method is proposed in the next section to tackle this problem.

IV. Nonlinear Optimization Approach to OSL Problem

The approach presented in this section is a gradient-based optimization algorithm which finds a suitable movement

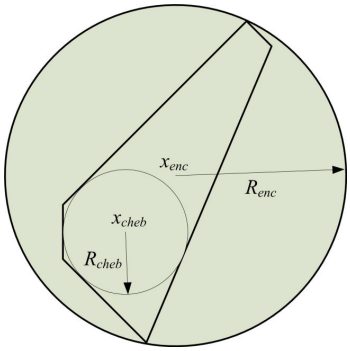

Fig. 2: Enclosing and Chebyshev circles.

direction p_k in an iterative fashion to increase the objective function in the OSL problem, based on its gradient with respect to the sensor position x_s. Given the direction p_k, a procedure called *line search* determines the optimal step size for maximizing the objective function in that direction. The optimization variable is updated in each iteration as follows:

$$x_{k+1} = x_k + \alpha_k p_k \tag{2}$$

The value of α_k is obtained from the line search procedure as noted above. It is discussed later that for the most efficient solution, the movement of the sensor should not be exactly in the direction of the gradient of the objective function; instead, it should be a scaled version of it. Different elements of this optimization strategy are described in detail in the following subsections.

A. Computation of the Gradient Vector

The gradient of the objective function is computed in the sequel using the results of [16].

Consider a region defined by the intersection of k inequalities $h_i(x_s,q) \leq 0$ for $i = 1,\ldots,k$. By concatenation of the boundary functions $h_i(x_s,q)$ as $h(x_s,q) = [h_1(x_s,q),\ldots,h_k(x_s,q)]^T$, this region can be represented by $h(x_s,q) \leq 0$. Define also $\mu(x_s) = \{q \in \mathbb{R}^n : h(x_s,q) \leq 0\}$, and denote the boundary of this set by $\partial\mu(x_s)$. The boundary of the region corresponds to the equalities in the above formulation. Note that this boundary has k segments, where each segment can be expressed as:

$$\partial_i\mu(x_s) = \mu(x_s) \cap \{q \in \mathbb{R}^n : h_i(x_s,q) = 0\} \tag{3}$$

A generic integral function over the region is considered as follows:

$$F(x_s) = \int\limits_{h(x_s,q)\leq 0} p(x_s,q)dq \tag{4}$$

The gradient of $F(x_s)$ with respect to x_s can be computed as:

$$\nabla_{x_s}F(x_s) = \int\limits_{\mu(x_s)} \nabla_{x_s}p(x_s,q)dq - \sum_{i=1}^{k} \int\limits_{\partial_i\mu(x_s)} \frac{p(x_s,q)}{\left\|\nabla_q h_i(x_s,q)\right\|} \nabla_{x_s}h_i(x_s,q)dL \tag{5}$$

For the sensor location problem, it is desired to find the gradient with the integrand $p(x_s,q) = \varphi(q)$. For a polygon with m facets, the corresponding region is described by a set of m linear inequalities as $Hq - K \leq 0$, where $H_{m\times 2}$ and $K_{m\times 1}$ are matrices with real entries. The sensing disk centered at x_s can also be expressed as $\|q-x_s\|^2 - R_s^2 \leq 0$, or equivalently as $(q-x_s)^T(q-x_s) - R_s^2 \leq 0$. With this formulation, one can set $h_i(x_s,q) = H_i q - K_i$ for $i \in \mathbf{m} := \{1,\ldots,m\}$ (where the H_i and K_i denote the i-th row of the matrices H and K, respectively) and $h_{m+1}(x_s,q) = (q-x_s)^T(q-x_s) - R_s^2$.

Now, one can find the gradient of the objective function from equation (5). Since the density function is independent of x_s, thus $\nabla_{x_s}\varphi(q) = 0$, and hence the first term in the right side of (5) vanishes. Also, the first m functions defining the region ($h_i(x_s,q)$ for $i = 1,\ldots,m$) do not depend on x_s, which means that $\nabla_{x_s}h_i(x_s,q) = 0$, for $i = 1,\ldots,m$. The gradients of $h_{m+1}(x_s,q)$ is computed as follows:

$$\nabla_{x_s}h_{m+1}(x_s,q) = -2(q-x_s) \tag{6}$$

$$\nabla_q h_{m+1}(x_s,q) = 2(q-x_s) \tag{7}$$

Therefore, the gradient of $F(x_s)$ with respect to the position of the sensor is given by:

$$\nabla_{x_s}F(x_s) = \int\limits_{\partial_{m+1}\mu(x_s)} \frac{q-x_s}{\|q-x_s\|}\varphi(q)dL \tag{8}$$

where $\partial_{m+1}\mu(x_s)$ is the portion of the perimeter of the sensing disk which is inside the polygon P. Note that the (8) is, in fact, the integral of a vector normal to the perimeter, pointing out of the sensing disk, and that the result of the integration is also a vector (the normal vector is denoted by $n(q)$ in Fig. 1).

The gradient function (8) provides an ascent direction for the area of the disk inside the polygon, which implies that moving the disk in this direction will increase the covered area within the polygon.

The points on the perimeter of the sensing disk can be characterized as:

$$q = x_s + R_s \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \tag{9}$$

where $\theta \in [0, 2\pi)$. As a result, $\|q-x_s\| = R_s, \forall q \in \partial_{m+1}\mu(x_s)$. On the other hand, the path $\partial_{m+1}\mu(x_s)$ can also be described by a number of arcs defined by a set of intervals $\Theta = \{[\theta_1,\theta_2],[\theta_3,\theta_4],\ldots\}$. Thus, the integral in the right side of (8) can be equivalently calculated as follows:

$$\nabla_{x_s}F(x_s) = R_s \int\limits_{\theta\in\Theta} \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \varphi(q)d\theta \tag{10}$$

where q in given by (9). Note that dL in (8) is replaced by $R_s d\theta$ in (10). This relation is important in that it presents a simple technique for finding the gradient by each individual mobile computing agent. More precisely, by choosing a sufficiently large number of points on the perimeter and using a proper numerical integration method, one can find a sufficiently close approximation of (10).

In order to reformulate the underlying optimization as a more conventional minimization problem, define $\bar{F}(x_s) = -F(x_s)$. As a result, the OSL problem can be rewritten as:

$$x_s^* = \arg\min_{x_s \in \Pi} \bar{F}(x_s) = \arg\min_{x_s \in \Pi} [- \int_{\Pi \cap D(x_s)} \varphi(q) dq] \quad (11)$$

It is important to note that the sensor is displaced after the iterations in (2) converge to a sufficiently small neighborhood of the optimal location. In order to distinguish the iteration variable in different steps of (2) from the sensor location, the vector of decision variables is denoted by x in the following formulation:

$$\nabla \bar{F}(x) = -R_s \int_{\theta \in \Theta} n(q) \varphi(q) d\theta \quad (12)$$

(note that the vector x will eventually converge to the optimum point x_s^*).

In order to present the proposed nonlinear optimization approach, some relevant analytical results are first derived in the next subsection.

B. Optimality Conditions

The gradient vector given in the previous subsection is used here to develop some important results. To this end, the *Karush-Kuhn-Tucker (KKT)* optimality conditions are obtained next for the solution of previously-mentioned constrained nonlinear optimization problem ([17], p. 342).

Theorem 1. *Given the optimization problem (11), at least one of the following conditions holds:*

(i) $\nabla \bar{F}(x_s^*) = 0$

(ii) $x_s^* \in \partial\Pi$

In addition for any positive-valued function of $\varphi(q)$*, the first holds at the optimum, and* $x_s^* \notin \partial\Pi$*.*

The proof is omitted here for the sake of brevity. ■

The analytical results given above can be used to define a criterion for termination of the underlying iterative nonlinear optimization algorithm. Moving in a descent direction, the optimum is reached at the boundary of the Voronoi polygon unless the gradient becomes zero elsewhere. In particular, for positive-valued density functions the stationary point of the generated sequence can be distinguished by the gradient measure only.

C. Line Search

Given a descent direction, it is of great importance to find a proper step size to move. This is usually performed using one of several existing line search methods. In particular, consider the following optimization problem:

$$\alpha_k = \arg\min_{\alpha} \bar{F}(x_k + \alpha p_k) \quad (13)$$

For a given function $\bar{F}(x)$, this is an optimization problem in scaler α and can be solved using simple optimization methods. However for the OSL problem the objective function is known implicitly, and hence conventional optimization techniques are ineffective. Alternatively, one can use a *direct line search* to find the optimal value of α by iteratively evaluating the objective function on the selected line. This requires the computation of the weighted intersected area of the polygon and the sensing disk while the center of the disk moves on the selected line. Such a computation is time-consuming in general , and to avoid it a line search based on the information obtained from the gradient vector is proposed here. To this end, it is important to note that the differentiation of the function $\bar{F}(x_k + \alpha p_k)$ with respect to the parameter α yields $p_k^T \nabla \bar{F}(x_k + \alpha p_k) = 0$. This means that at the optimum point, the gradient is perpendicular to the search direction p_k. Define the function:

$$M(\alpha) = p_k^T \nabla \bar{F}(x_k + \alpha p_k) \quad (14)$$

One can find the zero of $M(\alpha)$ numerically (e.g., using the Newton-Raphson search). Obviously this is much less computationally demanding than the direct line search approach.

Using the optimal value of α at some iterations might lead to a point x_{k+1} outside the Voronoi polygon. A projection procedure can be used to handle this problem as spelled out in the next subsection.

D. Projection onto the Voronoi Polygon

It is desired to project any point that falls outside the convex Voronoi polygon at some iteration of the proposed nonlinear optimization approach, onto the polygon. For a point $x_0 \notin \Pi$, its projection onto Π is denoted by $[x_0]_\Pi$ and is defined as follows:

$$[x_0]_\Pi = \arg\min_{x \in \Pi} \|x - x_0\|^2 \quad (15)$$

The minimization problem given above seeks the nearest point to x_0 inside Π and constitutes a quadratic programming problem. However, the convexity of the Voronoi polygons as well as the simplicity of the corresponding geometry makes the projection a rather straightforward problem. The following lemma states that the projected point is either a vertex of the Voronoi polygon or a perpendicular foot of x_0 on a facet.

Lemma 1. Consider the Voronoi polygon Π with m vertices denoted by v_i for $i \in \mathbf{m}$ and m facets, and denote the perpendicular foot of x_0 onto the i-th facet by $(x_0)_\perp^i$, for $i \in \mathbf{m}$ (note that $(x_0)_\perp^i$ may be on the extension of a facet, and hence may not belong to $\partial\Pi$). Define the set I as:

$$I = \{i \in \mathbf{m} : (x_0)_\perp^i \in \partial\Pi\} \quad (16)$$

and the corresponding points as:

$$(x_0)_\perp^I = \{(x_0)_\perp^i : i \in I\} \quad (17)$$

Then:

$$[x_0]_\Pi = \arg\min_{x \in A} \|x - x_0\|^2 \quad (18)$$

where:

$$A = \begin{cases} (x_0)_\perp^I & I \neq \emptyset \\ \{v_1, \ldots, v_m\} & I = \emptyset \end{cases} \quad (19)$$

The proof is omitted here for the sake of brevity. ■

The lemma simply states that the search for finding the projected point can be limited to the set of perpendicular

foots on the border of the polygon (if any) and the vertices of the polygon. Using the above lemma, one can limit the search domain in order to find the projected point more efficiently.

E. Scaled Gradient Projection Algorithm

A gradient projection method with non-diagonal scaling of the descent direction will now be provided. The descent-based optimization methods might demonstrate zigzag behavior while approaching the optimum point. It is known that using the Hessian matrix of the objective function ($\nabla^2\bar{F}(x)$) as the scaling matrix in the optimization algorithm results in better convergence. In particular, one can use the following:

$$\nabla^2\bar{F}(x_k)p_k = -\nabla\bar{F}(x_k) \tag{20}$$

or equivalently:

$$p_k = -[\nabla^2\bar{F}(x_k)]^{-1}\nabla\bar{F}(x_k) \tag{21}$$

Typically, in the OSL problem the computation of the Hessian matrix and its inverse is cumbersome. To remedy this shortcoming, one can use a positive-definite matrix as an approximation to the Hessian matrix at each iteration. One of the efficient approaches for this purpose is called *Broyden class*, and will be used here. The approximate matrix for the inverse of the Hessian matrix is obtained iteratively, initiated by an arbitrary positive definite matrix (e.g., the unity matrix) and updated at each iteration in such a way that the positive-definiteness of the matrix is preserved ([18] p. 470). Define s_k and y_k as follows:

$$s_k = x_{k+1} - x_k \tag{22}$$

$$y_k = \nabla\bar{F}(x_{k+1}) - \nabla\bar{F}(x_k) \tag{23}$$

and denote the approximation of the inverse of the Hessian matrix with H_k. The Sherman-Morrison update is given by:

$$H_{k+1} = \left[I - \frac{s_k y_k^T}{y_k^T s_k}\right] H_k \left[I - \frac{y_k s_k^T}{y_k^T s_k}\right] + \frac{s_k s_k^T}{y_k^T s_k} \tag{24}$$

Now, using the formula (21) with the above approximate matrix, one can write:

$$p_k = -H_k\nabla\bar{F}(x_k) \tag{25}$$

The scaling given above for the descent direction improves the convergence of the gradient projection algorithm ([19] p. 233).

To find a proper termination condition for the proposed iterative algorithm, two different cases are considered for the optimum point: (i) it is located inside the polygon; (ii) it is located on the boundary. For the first case, $\|\nabla\bar{F}(x)\|$ is small enough to deduce that the current position is in a sufficiently small neighborhood of the optimum point. Since the updating formula for H_k uses the difference of the gradient vectors in the last two last steps of the algorithm, the value of $\|y_k\|$ can also be important to check as part of the termination condition, for the numerical stability of the algorithm. On the other hand, if $x_s^* \in \partial\Pi$, it can be shown that for the optimum point $x_s^* = [x_s^*]_\Pi$. This means that in this case, sufficiently small values of $\|s_k\|$ represents "small" neighborhood of the optimum point.

Following the above discussion, one can choose a small positive number ε and compare it with $\|\nabla\bar{F}(x)\|$, $\|y_k\|$ and $\|s_k\|$ in each round of the algorithm. As soon as any of these values is smaller than ε, the algorithm is terminated. This will be hereafter be referred to as the *termination condition*.

Using the results obtained thus far, the nonlinear optimization approach for solving the OSL problem will be provided next. To avoid trivial cases described before, it is assumed that the sensing range is neither too small nor too large, such that the covered area within the Voronoi polygon is not fixed w.r.t. the location of the sensor (within a sufficiently small neighborhood). This issue will be further clarified in section V.

Scaled Gradient Projection Algorithm

1) Choose the initial values of x_k and H_k as $x_0 = x_s$ and $H_0 = I$.
2) Compute the value of $\nabla\bar{F}(x_k)$ from (12), and the descent direction p_k from (25).
3) Perform the line search procedure described earlier to find the value of α_k.
4) Update the desired position as $x_{k+1} = x_k + \alpha_k p_k$.
5) If x_{k+1} falls outside the corresponding Voronoi polygon Π, use the projection procedure and set $x_{k+1} = [x_{k+1}]_\Pi$.
6) Compute the value of $\nabla\bar{F}(x_{k+1})$.
7) Check the termination condition; if it is satisfied, then set $x_s^* = x_{k+1}$ and terminate the algorithm. Otherwise, go to the next step.
8) Find s_k and y_k from (22) and (23), respectively, as well as the matrix H_{k+1} using the updating formula (24).
9) set $k = k+1$ and go to step 2. ■

To illustrate the above algorithm, consider the OSL problem for a Voronoi polygon depicted in Fig. 3. Let the sensing radius be $R_s = 1.5$m. Let also the coverage priority of the field be characterized by a non-uniform density function given below:

$$\varphi(q) = e^{-2(q_1-2.8)^2-2(q_2-1)^2}$$

To clarify the procedure, the level sets of the objective function are also depicted in Fig. 3. Note that the objective function in the OSL problem (the term in the right side of 11) implicitly takes the required constraints into account. As a result, the level sets are not centered around the maximum of $\varphi(q)$ which corresponds to the point $\bar{q} = [2.8, 1]^T$. In fact, placing the sensor at $\bar{q}$ would lead to poor weighted coverage inside the polygon (note that $\bar{q}$ is outside the polygon, and hence some part of the covered area of the corresponding sensor falls outside the polygon).

Given the initial point $x_0 = [2.5, 3.7]^T$, the points generated in the first two steps of the algorithm fall outside the polygon, and hence the projection procedure is to be used. The path toward the optimum point under the proposed algorithm is also shown in Fig. 3. To better illustrate the projection results, a zoomed view of the points near the boundary of the polygon is also depicted in Fig. 4. It can be observed

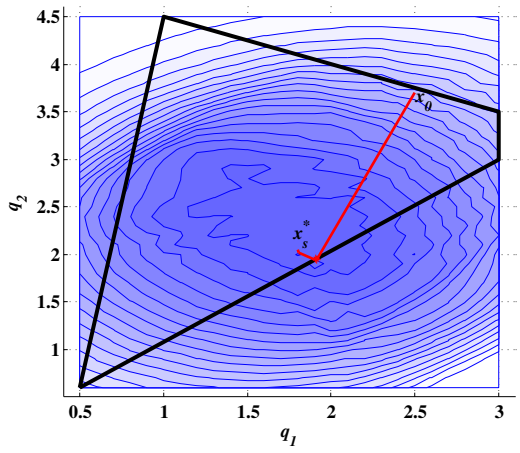

Fig. 3: Sequence of the points obtained by using the proposed scaled gradient projection algorithm.

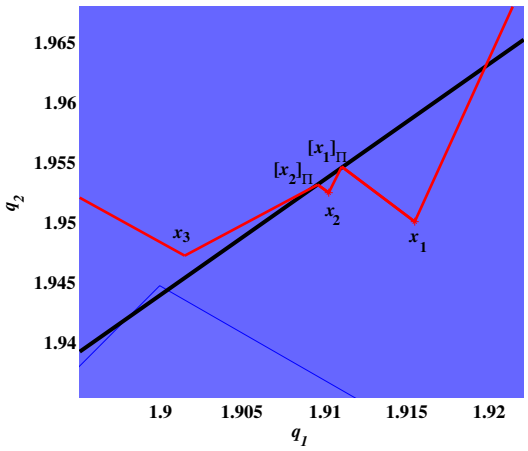

Fig. 4: Projection of the points near the boundary of the polygon.

from this figure that the algorithm converges to the point $x_s^* = [1.80, 2.05]^T$ (which is sufficiently close to the optimum point) in only 5 steps.

Remark 1. Note that new measurements are not required in the proposed optimization procedure. In other words, a sensor does not need to move before the termination of the iterations in the optimization algorithm. The sensor physically moves to its new location once the optimum point inside the polygon is obtained.

Remark 2. The sensor placement is discussed in [7], and the problem is introduced as the *area problem*. The gradient is derived in an alternative way, and an algorithm is presented accordingly, which uses line search. The approach presented here, however, is more general as it considers the relevant constraints, and also properly scales the steepest descent direction.

V. Sensor Networks Coverage Optimization

In a multi-sensor setting, a strategy similar to the single-sensor case discussed in the previous section is adopted by every sensor for coverage optimization. This technique is referred to as the *Max-Area* strategy.

A steady-state configuration of the network w.r.t. a given sensor deployment strategy is characterized as the case where the sensors are properly located in the plane such that none of them is required to move under this deployment strategy. In a Voronoi-based algorithm, this means that the sensing coverage of none of the sensors would increase by the prescribed threshold in the corresponding region Voronoi cell if the sensors move to their new locations under that deployment strategy.

In general, coverage maximization problem in a sensor network using a distributed decision making is a challenging problem. Hence, different coverage algorithms are usually compared to each other by Monte Carlo simulations, where the performance criteria are assessed by simulating the algorithms for several random initial values. The same method of comparison is used here to show the superiority of the proposed algorithm over the Minimax as a well-known coverage strategy [12]. The rate of convergence, final coverage pattern and traveled distance (as a measure of energy consumption) are common assessment criteria for a multi-sensor coverage algorithm, and will be used in the simulations to compare the efficiency of the strategies.

Example 1. To examine the Max-Area strategy, it is applied to a network of 24 mobile sensors in a 50m by 50m square field. Let $R_s = 6$m and $R_c = 20$m, and consider the initial sensor configuration (and the corresponding Voronoi diagram) depicted in Fig. 5. The movement paths of the sensors under the Max-Area strategy are also given in this figure and the blank circles show the position of the sensors after 16 rounds. The initial coverage factor (defined as the ratio of the covered area in the field to the total area of the field) in this example is 63%, and the proposed algorithm achieves 88% coverage in 16 rounds. The configuration of the network (location of sensors and their local coverage along with the corresponding Voronoi polygons) after 16 rounds is depicted in Fig. 6. The relatively short movement paths and 25% improvement in total coverage confirm the effectiveness of the Max-Area strategy in this example.

Example 2. To compare the performance of the Max-Area and Minimax strategies, 50 different initial settings of 30 identical sensors with the same sensing and communication capabilities as the previous example are considered, and both algorithms are applied to each setting. Taking the average coverage factor for each algorithm over the 50 settings, the graphs depicted in Fig. 7 are obtained. It can be seen from this figure that the Max-Area strategy outperforms Minimax in terms of both coverage and the rate of convergence.

This figure shows that in the worst case scenarios, the Minimax algorithm takes a longer time compared to the Max-Area algorithm to converge (in particular, in this example it took four more rounds for the Minimax algorithm to meet the termination condition). The average travel distance of the sensors (which is important as far as energy consumption is concerned) under both algorithms is about 119 meters. Note

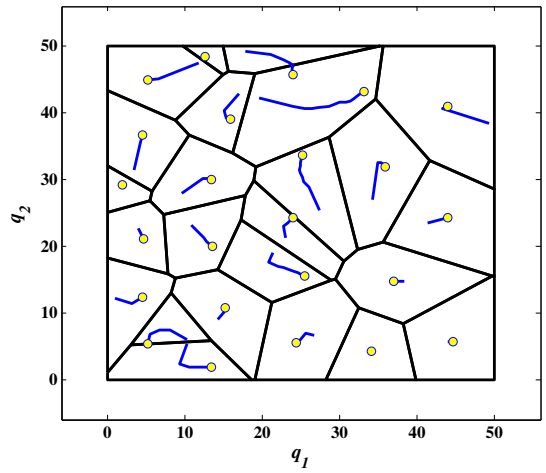

Fig. 5: The initial configuration of sensor network considered in Example 1, and the corresponding Voronoi diagram.

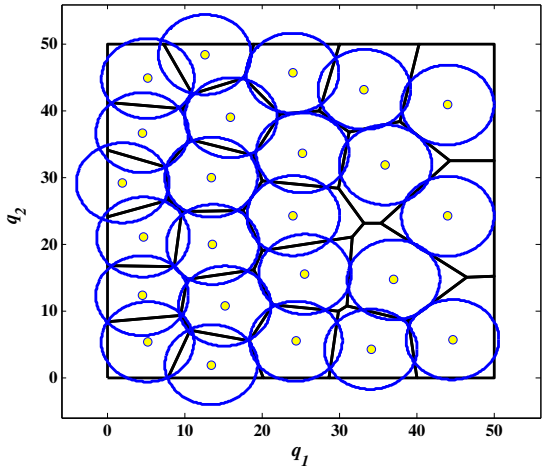

Fig. 6: The configuration of the sensor network after 16 rounds of the Max-Area strategy in Example 1.

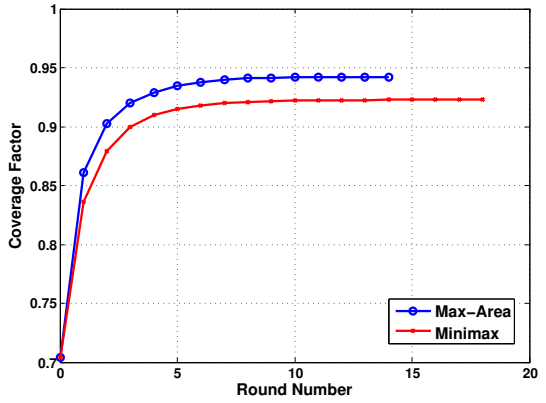

Fig. 7: Coverage factor in different rounds of the Max-Area and Minimax strategies in Example 2.

that the better performance of the Max-Area algorithm comes at the price of a higher computational complexity.

VI. Conclusion

In this paper, a nonlinear optimization algorithm called *Max-Area* is introduced to maximize coverage in a mobile sensor network by properly relocating the sensors. Under the proposed strategy, each sensor uses available information about other sensors to find its optimal location in the corresponding Voronoi polygon. The algorithm is based on the gradient projection technique which is properly scaled for better convergence. A step-by-step procedure is provided to find the desired location of each sensor systematically. Monte-Carlo simulations demonstrate the efficacy of the proposed strategy in maximizing the network coverage.

References

[1] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proceedings of 6th Annual International Conference on Mobile Computing and Networking*, 2000, pp. 56–67.

[2] G. J. Pottie and W. J. Kaiser, *Wireless Integrated Network Sensors*. ACM New York, NY, USA, 2000.

[3] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie, "Protocols for self-organization of a wireless sensor network," *IEEE Personal Communications*, vol. 7, pp. 16–27, 2000.

[4] H. Mahboubi, A. Momeni, A. G. Aghdam, K. Sayrafian-Pour, and V. Marbukh, "An efficient target monitoring scheme with controlled node mobility for sensor networks," *IEEE Transactions on Control Systems Technology*, to appear.

[5] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. K. Saluja, "Sensor deployment strategy for target detection," in *Proceedings of 1st ACM International Workshop on Wireless Sensor Networks and Applications*, 2002, pp. 42–48.

[6] A. Howard, M. J. Matarić, and G. S. Sukhatme, "An incremental self-deployment algorithm for mobile sensor networks," *Autonomous Robots*, vol. 13, pp. 113–126, 2002.

[7] J. Cortés, S. Martínez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 11, no. 4, pp. 691–719, 2005.

[8] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage problems in wireless ad hoc sensor networks," in *Proceedings of 20th IEEE INFOCOM*, 2001, pp. 1380–1387.

[9] D. Wang, J. Liu, and Q. Zhang, "Mobility-assisted sensor networking for field coverage," in *Proceedings of IEEE Global Communications Conference*, 2007, pp. 1190–1194.

[10] H. Mahboubi, K. Moezzi, A. G. Aghdam, K. Sayrafian-Pour, and V. Marbukh, "Distributed deployment algorithms for improved coverage in mobile sensor networks," in *Proceedings of IEEE Multiconference on Systems and Control*, 2011, to appear.

[11] G. Wang, G. Cao, P. Berman, and T. F. L. Porta, "A bidding protocol for deploying mobile sensors," *IEEE Transactions on Mobile Computing*, vol. 6, pp. 563–576, 2007.

[12] G. Wang, G. Cao, and T. F. L. Porta, "Movement-assisted sensor deployment," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 640–652, 2006.

[13] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi Tessellations: Applications and algorithms," *SIAM review*, vol. 41, pp. 637–676, 1999.

[14] R. Klein, *Concrete and Abstract Voronoi Diagrams*. Springer, 1989.

[15] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[16] S. Uryasev, "Derivatives of probability functions and some applications," *Annals of Operations Research*, vol. 56, pp. 287–311, 1995.

[17] D. Luenberger and Y. Ye, *Linear and Nonlinear Programming*. Springer, USA, 2008.

[18] S. N. I. Griva and A. Sofer, *Linear and Nonlinear Optimization*. SIAM, USA, 2009.

[19] D. Bertsekas, *Nonlinear Programming*. Athena Scientific, USA, 1999.