Event-driven sensor deployment using self-organizing maps

Cris Koutsougeras*

Department of Computer Science and Industrial Technology,
Southeastern Louisiana University,
Hammond, LA 70402, USA
E-mail: ck@selu.edu
*Corresponding author

Yi Liu and Rong Zheng

Department of Computer Science,
University of Houston,
Houston, TX 77204, USA
E-mail: yi.liu@uh.edu
E-mail: rzheng@uh.edu

Abstract: Coverage is an important optimization objective in pre and post-deployment stage of a Wireless Sensor Network (WSN). In this paper, we address the issue of placing a finite set of sensors to cover an area of arbitrary geometry. Unlike many existing works concerned with uniform coverage of a target area, we take into account the realistic consideration of the probability density for events to be sensed, termed as *event-driven coverage*. The objective is to distribute sensors so that the distribution density of the sensors matches that of the probability density of events to be sensed. The expected event distribution is assumed to be stationary and known a priori, directly or indirectly, in the form of sample maps. In this context we explore and evaluate the concept of Self-Organizing Maps (SOMs) and its derivative variants to address the coverage problem. Various forms of SOMs methods as well as the known methods of Virtual Fields are also compared via experimentation.

Keywords: sensor deployment; event-driven coverage; Self-Organizing Map; SOM.

Reference to this paper should be made as follows: Koutsougeras, C., Liu, Y. and Zheng, R. (2008) 'Event-driven sensor deployment using self-organizing maps', *Int. J. Sensor Networks*, Vol. 3, No. 3, pp.142–151.

Biographical notes: Cris Koutsougeras received a PhD from the Department of Computer Science at Case Western Reserve University in 1988, his MS from the University of Cincinnati and a BS in Electrical Engineering from the National Technical University of Athens, Greece. He was a Faculty at Tulane University in New Orleans for 18 years and he is currently the Head of the Department of Computer Science and Industrial Technology of Southeastern Louisiana University. He was the Leader of Tulane's 2005 Darpa Grand challenge team which developed the Gray Team's Kat-5.

Yi Liu is presently a Master's student at the University of Houston. His research interests include wireless sensor networks.

Rong Zheng received her PhD from the Department of Computer Science, UIUC in 2004 and received an ME and a BE in Electrical Engineering in 1998 and 1996 from Tsinghua University, PR China. She is now an Assistant Professor in the Department of Computer Science at the University of Houston. She received the National Science Foundation CAREER Award in 2006. Her current research interests include resource management of large-scale distributed systems, modeling and prototyping of wireless systems.

1 Introduction

Systems with a large number of sensors have become common in many different domains of applications. The development of wireless sensors which are able to communicate their data to stations without a wire link have accelerated the use of sensors in real world applications since their deployment becomes easier and at lower cost. As the main utility of Wireless Sensor Networks (WSNs) is to detect and collect information in some designated region, spatial distribution and coverage of wireless sensors became a key optimisation goal in sensor deployment and post-deployment phases. For example, in a security system, sensors need to be placed to completely

cover certain areas of complex geometry without leaving any blind spots and we wish to do that with a minimal number of sensors. In air-quality monitoring applications, one wishes to deploy sensors in locations where contaminants are likely to occur. Interestingly, we observe that the sensor coverage problem bares much similarity with capacity (cell) planning in cellular networks, where the sensors are base-stations communicating with consumer equipment like cell phones. In this case, maximum coverage of large areas of varying population (and thus varying traffic demand) is desired with a restricted number of base-stations. This problem becomes even more interesting in the case where the sensors are mobile and they can be rearranged even after deployment, especially when distribution of signal sources is not stationary. For example, in the self-healing minefield project (DARPA/ATO, 2002) developed by DARPA, a network of mines is established and could recover from breaches in the network. After part of the minefield is breached, the rest of the network detects and determines how to heal the network by moving individual mines.

The Self-Organizing Map (SOM) is a method for unsupervised learning, based on a grid of artificial neurons whose weights are adapted to topologically match the distribution of vectors in a sample set that is used as exemplar or training set. It has been successfully applied in vector quantization, a minimal distortion encoding technique based on compression principles, because it provides an approximate method for computing the Voronoi vectors in an unsupervised manner.

In this paper, we model the sensor deployment problem as a geographical area with a random (but not necessarily uniform) distribution of events to be sensed, and we address the question of what is an efficient way to distribute a given finite set of sensors in this area so as to reflect the expected sparseness or concentration of events. We adopt the SOM approach since it specifically targets topologically correct mappings of probability density distributions. A couple of SOM variants are examined and experimental evaluations are performed. In addition, we provide a comparison with the Virtual Fields method for which spurious non-proper stable distributions exist – a result for which proof is provided.

The rest of this paper is organized as follows. In Section 2, we present a review of related work in sensor deployment. A formal definition of the event-driven sensor deployment problem statement is given in Section 3 along with basic concepts on SOM. In Section 4, the basic algorithm and its variants for event-driven sensor deployment are presented; their performance is evaluated in more detail in Section 5. Finally, we summarize the results and conclude this paper in Section 6.

2 Related work

Various approaches have been proposed in the literature to address the sensor deployment problem. These algorithms differ in whether they are applicable for offline deployment or online relocation of sensors in postdeployment stage, whether accurate location information is required, and whether irregular areas and heterogeneous coverage requirements can be met.

Among the existing methods, the potential field-based approaches are closest to our proposed solution (Howard et al., 2002; Zou and Chakrabarty, 2003). In Howard et al. (2002), each node is subjected to virtual forces, which repel nodes from one another and from obstacles. At the same time, nodes are subjected to viscous friction forces. As a result, an initial compact configuration of nodes is expected to spread out and finally reach an equilibrium state. The approach does not require centralized control and exact localization information. However, the final placement is dependent on initial sensor location. Furthermore, it cannot handle the non-uniform coverage requirement when the probability distribution of events varies over the target area.

A similar idea has been explored in Zou and Chakrabarty (2003). The main difference is that in addition to repelling forces between close-by sensor pairs, attractive forces among distant sensors are also considered. This solution is known as the Virtual Force Algorithm (VFA). Using a combination of attractive and repellant forces, VFA attempts to improve the coverage of an initial random placement of the sensor network. However, as we demonstrate in Section 5 and Appendix, VFA can suffer a problem of under-coverage and poor placement for certain configurations of initial sensor placements which are not proper and yet are stable under VFA. Furthermore, it should be noted that VFA addresses uniform coverage requirements only, whereas our proposed solution can facilitate coverage requirements of arbitrary spatial distribution.

Recently, three mobility-assisted sensor deployment protocols, VEC (Vector-based), VOR (Voronoi-based) and Minimax are proposed in Wang et al. (2004) based on Voronoi diagrams. VEC pushes sensors away from a densely covered area; VOR pulls sensors to the sparsely covered area, and Minimax moves sensors to their local centre area. The three protocols share the same principle of moving sensors from densely deployed areas to sparsely deployed areas iteratively. Simulation results show that Minimax performs better than the other two with respect to achieving larger coverage. In a follow up work (Wang et al., 2005), the same authors consider the problem of sensor relocation, which deals with sensor failure or response to new events. The solution consists of two phases. Firstly, redundant sensors are identified. Secondly, the redundant sensors are moved to the under-covered area. Specifically, a Grid-Quorum solution is proposed to quickly locate the closest redundant sensors, and then a cascaded movement solution adjusts relevant sensors' positions at the same time.

Inspired by the minimum-cost maximum-flow problem, Chellappan et al. (2005) constructed a virtual graph based on the initial deployment, and determined routes for the sensors maximising their coverage and minimising the number of flips. The associated algorithm requires a Base station to collect sensors' information and determine the route plan. The coverage improvement by that algorithm is constrained by initial distribution of sensors.

In addition to the aforementioned works directly addressing the sensor deployment problem, there exist other research efforts concerning the detection and recovery of coverage holes in sensor networks (Ghrist and Muhammad, 2005; Watfa and Commuri, 2006). A coverage hole is defined as a region where the coverage requirement is not met. A coverage hole could arise due to the failure of sensors, change of sensors' location, etc. In Watfa and Commuri (2006), an algorithm is devised to find out coverage holes in the region. By examining whether a sensor's sensing disk is covered by another sensor, one can determine whether the sensor is on the boundary of a hole, implicitly indicating hole's existence. It only suggests how to detect holes but it does not provide methods for recovery of holes. The work in Ghrist and Muhammad (2005) presented a novel technique for detecting holes in coverage by means of homology, an algebraic topological invariant. Their technique is coordinate-free and centralized.

3 Preliminaries

In this section, we first present the formulation of our version of the sensor deployment problem, and then provide a brief overview of the method of SOMs.

3.1 Problem statement

We consider a given number of N sensors available for dispersion in a target 2D area of arbitrary shape. The sensors will be sensing events from within this area, and the events can originate anywhere in the area with a stationary and a priori probability density distribution (which however may not necessarily be uniform). We assume that all sensor nodes are homogenous and have similar sensing capability. To determine the coverage of each sensor, a simple disk model is assumed, though it is *not* essential to the execution of our proposed algorithm. Under the disk model, a point p in the target area is covered if and only if there exists at least one sensor within its s radius, where s is the sensing range of a sensor. In the rest of this paper, bold face characters are used to represent vectors.

Definition 1: *(k-coverage): Given an area A and the set of sensors N, we say that A is k-covered if* $\forall p \in \mathbf{A}$, $\exists v_1, v_2, ..., v_k \in \mathbf{N}$, *such that p is covered by* $v_1, v_2, ..., v_k$. *Let then* $c(p)$ *denote the number of sensors covering p.*

In the sensor deployment problem, our goal is to deploy the N sensors in an efficient way to cover the target area subject to expected event distribution. Depending on the number and coverage area of sensors, two situations may arise:

1 There are not enough sensors to provide 1-coverage to the entire area. We wish to cover with high priority the event regions with larger probability density.

2 If the sensors are sufficient for k-coverage, $k \geq 1$, we wish to allocate more sensors in the areas of higher probability density than those with a lower density. This is often more desirable compared to a uniform distribution of sensor nodes for two reasons. Firstly, with higher event probability, more energy in sensing and communicating the sensed data is required. It is desirable to distribute the sensing task among multiple nodes. Secondly, a higher event probability can be an indication of 'importance' of a particular region, for example, secured zones. Thus, it is desirable to have more sensors deployed to ensure high reliability and effectively allocate more bandwidth to areas of more activity.

Therefore, to characterise the 'fitness' of resulting coverage, traditional metrics such as k-coverage is not expressive enough to capture a priori knowledge of event distribution. To this end, we introduce the following definition of *coverage divergence* motivated by the concept of *Kullback-Leibler* divergence in information theory.

Definition 2: (Coverage Divergence): *Let* $f_A(p), p \in \mathbf{A}$ *be the probability density function of events to be sensed in Area A. Given a placement* $\mathbf{P}(\mathbf{N})$ *of sensors N, define* $C_A(p) \triangleq \frac{c(p)}{\sum_{q \in A} c(q)}, p \in A$ *as the coverage density of* $\mathbf{P}(\mathbf{N})$.

The coverage divergence between $f_A(p)$ *and* $C_A(p)$ *is then defined as follows:*

$$D_A(f \| C) = \int_{P \in A, C_A(p) \neq 0} f_A(p) \log \frac{f_A(p)}{C_A(p)} \quad (1)$$

Definition 3: *(Event-driven Sensor Deployment Problem): Given* $f_A(p)$, *the probability density function of events to be sensed in Area A and a set of N sensors,*

$$\min \; D_A(f \| C)$$

A few comments are in order. Firstly, in the definition of coverage divergence in Equation (1), the sum is computed only at points which are covered. This prevents $D_A(f \| C)$ from going to infinity if a point is not covered. An alternative would be to lower bound $C_A(p)$ with a small value. Secondly, by Cover and Thomas (1991), Theorem 2.6.3, $D_A(f \| C)$ is non-negative with equality if and only if $f_A(p) = C_A(p)$, for all p. Since $C_A(p)$ is at best a discretised approximation of $f_A(p)$, $D_A(f \| C)$ is always a positive number. However, the smaller $D_A(f \| C)$, the closer is the approximation. Ideally, if there are infinite number of sensors with infinitely small sensing radium, $D_A(f \| C) \to 0$. Thirdly, in Equation (1), we can say that the 'utility' of covering an undesired location is zero, as $f_A(p) = 0$. Fourthly, since $f_A(p)$ can be an arbitrary probability density function, the above optimisation typically does not have closed form solutions. Therefore, heuristics need to be developed to approximate the optimal solution. Finally, non-preferential sensing coverage can be modelled by setting $f_A(p)$ to be uniform in Area A.

3.2 Self-organizing map

The SOM is a method for unsupervised learning, based on a grid of artificial neurons whose weight vectors are adapted to topologically match the distribution of vectors in a sample set that is used as exemplar or training set. It was first developed by Teuvo Kohonen and is thus also known as a Kohonen map. A brief summary of SOMs is given below. Detailed descriptions of SOM can be found in Kohonen (1982, 1990) and Haykin (1999).

The basic structure of the Kohonen layer consists of N processing elements, each receiving n input signals $x_1, x_2, ..., x_n$ from a layer of fan-out units. The x_j input to Kohonen processing element i has a real weight w_{ij} associated with it. Each processing element calculates its input intensity as $I_i = D(\boldsymbol{w}_i, \boldsymbol{x})$, where $\boldsymbol{w}_i = (w_{i1}, w_{i2}, ..., w_{in})^T$ and $\boldsymbol{x} = (x_1, x_2, ..., x_n)^T$ and $D(\boldsymbol{u}, \boldsymbol{v})$ is a distance measurement function. Once each processing element has calculated its input intensity I_i, a competition takes place to see which unit has the largest input density and its output z_i is set to 1 (in the following we refer to this element as the 'winner'); for all others it is set to 0.

Training data for the Kohonen layer is assumed to consist of a sequence of input vectors $\boldsymbol{x}$. Weight modification takes place in accordance with the adaptive rule, called the Kohonen learning law.

For the winning processing element:

$$\boldsymbol{w_i}^{\text{new}} = (1-\alpha)\, w_i^{\text{old}} + \alpha x \tag{2}$$

where α is a constant, $0 < \alpha < 1$, (thereby the winner moves closer to the current sample $\boldsymbol{x}$) and for all other processing elements:

$$\boldsymbol{w_i}^{\text{new}} = w_i^{\text{old}} \tag{3}$$

As training progresses, the topological distribution (map) of the Kohonen weight vectors becomes denser where the population of $\boldsymbol{x}$ vectors are highest, and similarly they become least dense (or absent) where $\boldsymbol{x}$ vectors hardly ever (or never) appear. Clearly, there exists a direct connection between our event-driven sensor deployment problem and SOM methods. We can model a sensor's location as weight vectors in a Kohonen layer whereas the sequence of training data (called *samples*) reflects the expected event distribution in the targeted area. Adjustment of sensors (weights) is dependent upon their distance to the samples.

4 Sensor deployment using SOM

In this section, we discuss a centralized algorithm to place a set of N sensors in a targeted area so that the distribution of the sensors approximates the distribution of events known a priori. We first describe a base algorithm using SOM and examine its performance. Then, we consider a few variations to the base algorithm that achieve better coverage.

4.1 Base algorithm

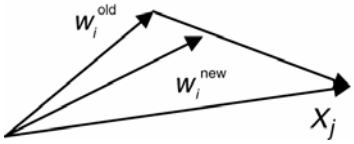
The base algorithm starts with an arbitrary initial placement of sensors, followed by an iterative adaptive process of rearrangement through small movement of sensors. Following the notations in Section 3.2, in the initialisation stage, a set of sample points $\boldsymbol{x}_j = (x_{j1}, x_{j2})^T,\ j = 1, 2, ..., m$ is generated to represent the distribution of events. In essence, the distribution of the samples corresponds to a discretisation of the distribution at hand. Each sensor i is assigned an arbitrary initial location $\boldsymbol{w_i} = (x_{j1}, w_{i2})^T, i = 1, 2, ..., N$.

In the iteration phase, sensor positions are moved based on the following criterion until no more significant movement is attainable. Each iteration focuses on one particular sample and the sensor that is closest to this sample is identified (based on Euclidean distance). This sensor is called the 'winner' for the current sample and this winner is allowed to move closer to the sample. The movement is proportional to the distance of the winner sensor and the current sample. Let the location of the winning sensor for sample $\boldsymbol{x}_j = (x_{j1}, x_{j2})^T$ be $\boldsymbol{w}_i = (w_{i1}, w_{i2})^T$. Updating of sensor i's location is determined by

$$w_i^{\text{new}} = (1-\alpha) w_i^{\text{old}} + \alpha x_j = w^{\text{old}}{}_i + \alpha (x_j - w^{\text{old}}{}_i) \tag{4}$$

with $0 < \alpha < 1$ as shown in Figure 1.

Figure 1 Adjustment of winner sensor location



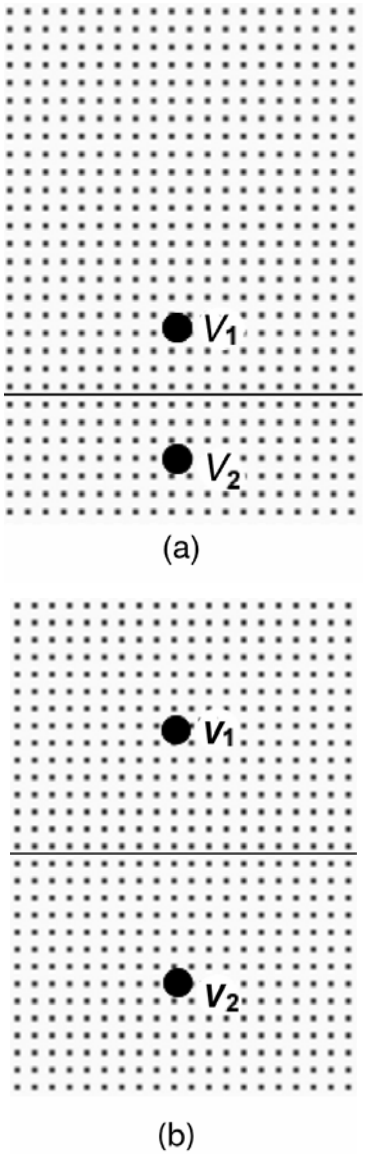
This process iterates over all the samples in multiple runs until there is no more significant movement of any sensor. For each sample, a winner is picked as the closest sensor. The winner then moves closer to the sample to better establish its 'claim'. Accordingly, the rest of the sensors will have to turn away from an area that is dominated by the current winner, and that happens in subsequent iterations with respect to other samples which are not dominated by other sensors.

Next, we provide an intuitive explanation of how SOM works. A Voronoi tessellation is the partitioning of a plane into convex polygons such that each polygon contains exactly one generating point and every point in a given polygon is closer to its generating point than to any other. At the beginning of each run, we can tessellate the target area into Voronoi cells $\{V_i\}$ with the current set of sensor locations as generating points. Therefore, the collection of samples, which a sensor i wins (called *receptive field* of sensor i) is in fact the Voronoi cell V_i. Since each sample within V_i pulls sensor i towards its own position, the collective effect of samples in V_i is to move the sensor i to their collective centre of gravity (mean). After one run, sensors move to new locations resulting in a different tessellation. The process then repeats. An illustrative

example of the behaviour of this simple algorithm is shown in Figure 2. Assume a uniform distribution of samples (to model a uniform event field), shown in Figure 2 as dots on a grid, and only two sensors with initial positions shown in Figure 2(a). It is obvious that sensor v_1 will be winning on all samples above the line shown midway between the two sensors. Similarly sensor v_2 will be winning in the smaller area below the line.

In Figure 2(a), there are a lot more samples above v_1 than there are below it within its receptive field (above the line). Thus, with continued iterations, v_1 is expected to move upwards. While v_1 is moving upwards, there are samples which effectively cross into v_2's receptive field. Thus v_2's receptive field expands upwards and so v_2 is also moving upwards pulled at all times towards the median of its receptive field. The final positions are shown in Figure 2(b) with the two sensors uniformly distributed and the area split evenly in their receptive fields.

Figure 2 (a) Original sensor positions and (b) After iterations



4.2 Improvement of the base algorithm

This simple algorithm works well in general with uniform initial distributions of the sensors. However, our earlier experiments show that it has some drawbacks.

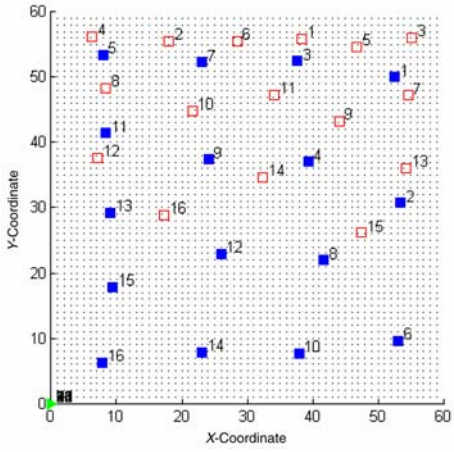
- The final placement is dependent upon the sweeping sequence of the samples. An example is shown in Figure 3. In Figure 3(a), sensors are initially at the lower left corner of the field or follow random placement as indicated by the green triangles. Using sequential sweeping of samples bottom-up, results in a final placement shown by the red squares. Clearly, sensors tend to concentrate on the upper half of the target area, which is not a desired formation. Similar observations can be made with a random initial placement in Figure 3(b).

- If the area to be covered consists of a set of disjoint regions, there may exist stable distributions of sensors as a result of the base algorithm in which some of the sensors' location remain unchanged while certain regions in the targeted area are under-covered. This is possible in that version of the algorithm because due to the topology of the initial placement, some sensors may become dominant over an area and constantly win the competitions over that area thus becoming greedy and effectively mask off the area from other sensors. As a result, other sensors cannot move in for the desirable distribution. An example is given in Figure 4. In Figure 4, events are three times more likely to occur in the upper right regions. Contrary to the targeted distribution (as indicated by the density of dots), there are more sensors in the lower left region. In addition, sensors 12 and 15 are trapped and do not contribute much to the final coverage of the intended rectangle areas because they hardly have won any of the competitions for samples throughout the iterations. To deal with these two problems, we examined two variations of the base algorithm.

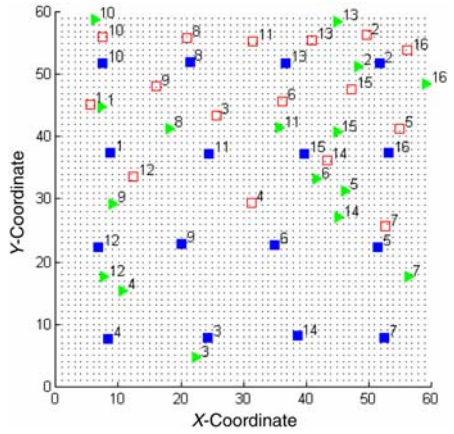
4.2.1 Random permutation of sweeping sequence

In each run, the sweeping of samples follows an arbitrary sequence as opposed to sweeping them in exactly the same sequence every time. This removes the artifact of sequential sweeping. The results are shown as blue solid squares in Figure 3. The difference observed in Figure 3 indicates that varying the order of sweeping, the samples can be advantageous. The reason is that if the order of sweeping is kept fixed, then dynamic equilibrium can arise that locks sensors in arrangements that are not optimal but from which they cannot break away. Varying the order of sweeps breaks away from such arrangements (in a somewhat similar way that small white noise perturbations effect in such cases).

Figure 3 Effect of sweeping sequence. (a) All 16 sensors are initially placed in initial lower left corner of an area. (b) All sensors are initially placed uniformly in targeted area (see online version for colours)



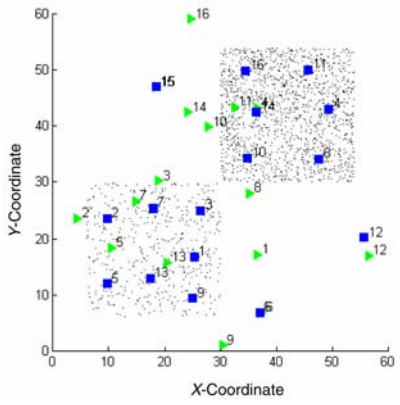
(a)



(b)

Note: Number of samples = 3481, Number of sensors = 16, α = 0.01, run = 500. Initial placement of sensor is indicated by green triangles. Red squares are the results of sequential sweeping, and blue solid squares are the results of the permuted sweeps

Figure 4 Effect of disjoint regions (see online version for colours)



Note: There are altogether 16 sensors to cover two disjoint regions with different event distribution requirement. Triangles give the initial placement of sensors and rectangles are outputs from the base algorithm. Sensors 12 and 15 do not move throughout the iterations of the algorithm.

4.2.2 SOM with block party

In the base algorithm, a single winner is determined for each sample. If a sensor cannot win on any sample (i.e. there is no sample in its receptive field), the position of the sensor remains unchanged. This is undesirable for two reasons,

1 the contribution of the sensor can be zero since it does not cover any region in the targeted area

2 a subset of sensors can be too greedy (i.e. they are the only winners always) and so the coverage will be attempted with only that limited set of sensors, thus the area can be under-covered (as shown in Figure 4).

To avoid this problem, we examined a variant of the base algorithm which is known as the *Block Party* version. The basic idea is to allow the winning sensors to drag along some of its neighbour sensors, thereby essentially sharing its gains upon an area with its neighbours and thus the greediness problem is handled. This is accomplished by letting the immediate neighbours of the winning sensor to move towards the current sample but with a proportionally *lesser* step than that of the winner. In the terminology of Kohonen maps, every sample determines multiple winners, each modifying its weight based on its 'distance' to the top one winner; the intuitive idea being to make the step some sort of Gaussian function of the distance from the top winner.

The question remains about how *neighbourhood* and *distance* are defined. There are two alternatives.

1 *Euclidean space*: A natural definition of neighbouring relationship is Euclidean distance. Let the winner sensor to sample $\boldsymbol{x}_j$ be $\boldsymbol{w}_i = (w_{i1}, w_{i2})$ and its movement is given by $\alpha(\boldsymbol{x}_j - \boldsymbol{w}_i)$. Then, the movement of a sensor $\boldsymbol{w}_k = (w_{k1}, w_{k2})$ is computed as $f(\| \boldsymbol{w}_i - \boldsymbol{w}_k \|)\alpha(\boldsymbol{x}_j - \boldsymbol{w}_i)$, where $\|\bullet\|$ is the L^2 norm. The weight function f is defined as follows:

$$f = \begin{cases} 1 \quad \text{if} \quad d = 0 \\ \left(1 - \left(\dfrac{d}{d_{\text{th}}}\right)^2\right) e^{-d^2/2d_{\text{th}}^2} \quad \text{if} \quad d < d_{\text{th}} \\ 0 \quad \text{if} \quad d > d_{\text{th}} \end{cases} \tag{5}$$

where $d = \| \boldsymbol{w}_i - \boldsymbol{w}_k \|$ is the Euclidean distance between a sensor k to the winner sensor i and d_{th} is the neighbourhood size which is set to be twice the sensing range s in subsequent experiments.

2 *Label space*: Instead of using the actual locations of the sensors, we can label them using coordinates in a label space and define neighbourhood accordingly. Given a sample $\boldsymbol{x}_j$, let the winner sensor $\boldsymbol{w}_i = (w_{i1}, w_{i2})$'s coordinate in label space be $(l_{i1}, l_{i2}), l_{i1}, l_{i2} \in Z$. Its movement is again computed as $\alpha(\boldsymbol{x}_j - \boldsymbol{w}_i)$ (in Euclidean space). Then, the movement of a sensor (l_{k1}, l_{k2}) is computed as

$f(|l_{i1}-l_{k1}|,|l_{i2}-l_{k2}|)\,\alpha\,(\mathbf{x}_j-\mathbf{w}_i)$. The weight function f is defined as follows:

$$f(x,y)=\begin{cases} e\dfrac{x^2+y^2}{M^2} & \text{if} \quad \max(x,y)<\dfrac{M+1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Clearly, $f(x,y)$ is symmetric function defined on a square neighbourhood with side length M. M determines the size of neighbour. An example of the weight (of movement) of neighbour sensors is given in Table 1.

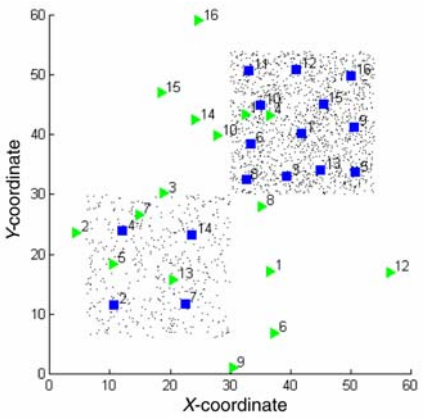
Table 1 Weights of movement of neighbouring sensors, $M = 5$

0.73	0.81	0.85	0.81	0.73
0.81	0.92	0.96	0.92	0.81
0.85	0.96	1	0.96	0.85
0.81	0.92	0.96	0.92	0.81
0.73	0.81	0.85	0.81	0.73

Comparing the above two alternatives, we see that with neighbourhood defined in Euclidean space, the sensors moved in each iteration tend to be more clustered but in principle two sensors that are initially neighbours are not restricted to remain neighbours throughout the algorithm. On the other hand, in label space, the sensors that are initially neighbours will remain labelled as neighbours throughout the algorithm. If we picture the initial placement as a perfect uniform grid, then throughout the algorithm this grid is stretched and skewed to span (cover) just the geometry of the sample space and at the same time being denser (finer) in areas of denser sample populations. Also, as the algorithm progresses, sensors which are neighbours in label space tend to come closer and become neighbours in Euclidean space. Thus, it is likely the two definitions have similar effects.

As an evaluation of the concept, we revisit the example in Figure 4 and apply the block party algorithm in Euclidean space. As shown in Figure 5, the block party algorithm in Euclidean space can indeed improve the efficiency of sensors and achieve the desired coverage requirement.

Figure 5 Block party algorithm in Euclidean space (see online version for colours)



Note: Number of samples = 2000, Number of sensors = 16, α = 0.01, Iteration = 50, size of initial neighbour = 80.

5 Performance evaluation

In this section, we evaluate the performance of the proposed sensor deployment algorithms. We have implemented the proposed algorithms as well as the VFA (Zou and Chakrabarty, 2003) using Matlab. As discussed in Section 2, VFA is only applicable for scenarios where event distribution is uniform in the targeted area. Furthermore, it is difficult to characterise the boundaries of the targeted area in VFA. Both problems can be addressed easily using our proposed framework.

The performance measures of interest when comparing different algorithms are

1 distribution of sensors,

2 coverage ratio, defined as the percentage of targeted area covered

3 speed of convergence measured in number of iterations.

5.1 *Uniform event distribution in square area*

This is the canonical scenario used in many works on sensor deployment. In this experiment, there are 16 sensors with sensing radius 7.5 in a 60 by 60 field.

Figure 6 shows the final placement of the sensors as the result of VFA and base algorithm with random permutation. The coverage of the sensor network is 70.00% in VFA, while it is 76.58% with the proposed algorithm (which is optimal in this case). Our implementation of VFA assumes the binary disk sensing model. The force vector on sensor $\mathbf{w}_j=(w_{j1},w_{j2})$ from $\mathbf{w}_i=(w_{i1},w_{i2})$ is computed as follows. When the distance between the two sensors is beyond d_{th}, the force is attractive. Otherwise, the force is repellent.

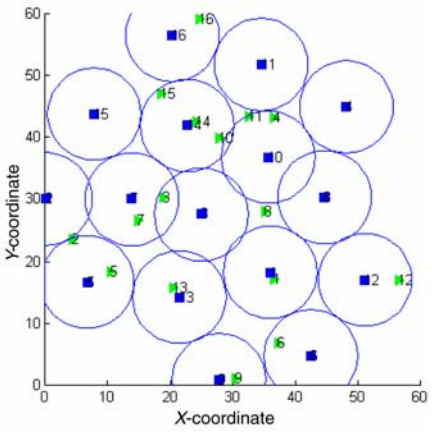
$$\mathbf{F}_{ij}=\begin{cases} c_A\left(\|\mathbf{w}_i-\mathbf{w}_j\|-d_{th}\right)\times\dfrac{\mathbf{w}_i-\mathbf{w}_j}{\|\mathbf{w}_i-\mathbf{w}_j\|} & \text{if} \quad \|\mathbf{w}_i-\mathbf{w}_j\|\geq d_{th} \\ c_R\dfrac{\mathbf{w}_j-\mathbf{w}_i}{\|\mathbf{w}_i-\mathbf{w}_j\|} & \text{otherwise} \end{cases} \tag{7}$$

where c_A and c_R is a constant, d_{th} is the neighbourhood size set to be the sensing range as recommended in Zou and Chakrabarty (2003). Thus, the movement of sensor j is given by $\alpha\sum_{i\neq j}\mathbf{F}_{ij}$.

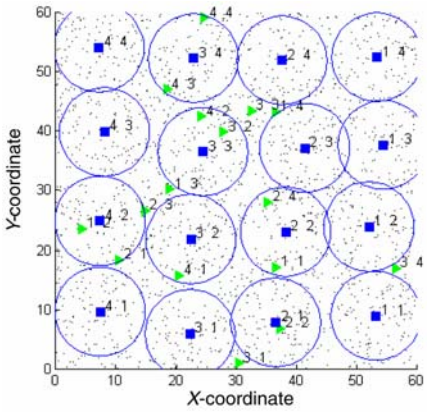
The result shown in Figure 6(a) is not an isolated instance of bad parameter tuning. In fact, we are able to prove (see Appendix) that VFA may stabilise in pathological cases of poor coverage.

Figure 7 shows the evolution of sensor coverage with number of iterations. At round 38, SOM is close to optimal. VFA appears to improve the total coverage fast initially, but slowly converges to steady state with more iterations.

Figure 6 Uniform event distribution in square region (a) VFA (b) SOM using block-party in label space (see online version for colours)

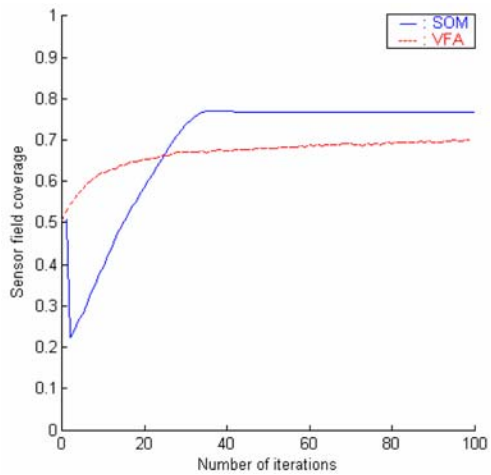


(a)



(b)

Figure 7 Sensing coverage of VFA and SOM (see online version for colours)



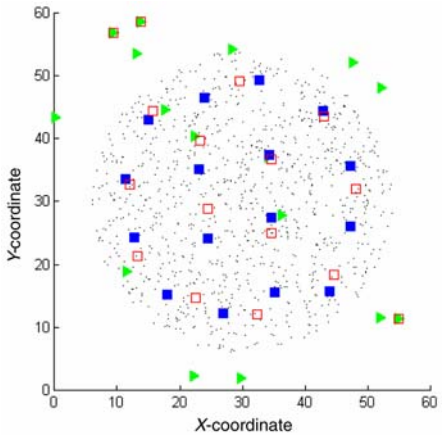
5.2 *Uniform event distribution in 'irregular' area*

Next, we conducted experiments for event distribution in non-rectangular areas. The term *irregularity* is slightly abused in this context. In the first scenario, the target placement is to deploy nodes uniformly inside a circle and in the second scenario in an L shape corridor. In both cases, a total of 16 sensors are deployed. We do not compare our results with VFA as handling of boundary condition is unspecified in VFA. From Figure 8, we see that SOM can naturally handle an arbitrary coverage area by simply generating samples accordingly. Due to the use of block party variation, all sensors contribute to the final coverage.

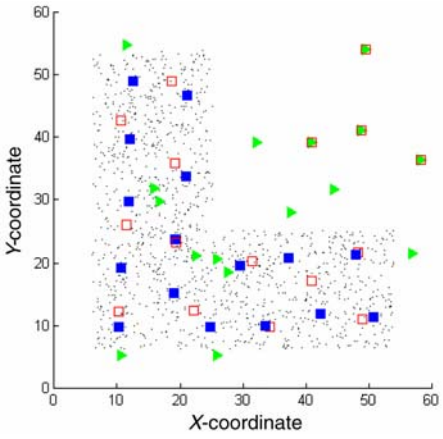
5.3 *Non-uniform event distribution*

In this set of experiment, we consider non-uniform event distributions in targeted areas. In Figure 9(a), the target area is equally divided into four regions with expected *k*-coverage ratio being 1:2:4:8 from bottom-up. A total of 16 sensors are deployed. After applying the SOM algorithm, the number of sensors in each region is 2, 2, 4 and 8, respectively. One interesting observation is that the bottom two sensors are placed closer to the boundary of the two lower regions and thus provide some coverage for the second bottom region.

Figure 8 Uniform distribution of event in irregular areas (a) Circular area and (b) L-shape area (see online version for colours)



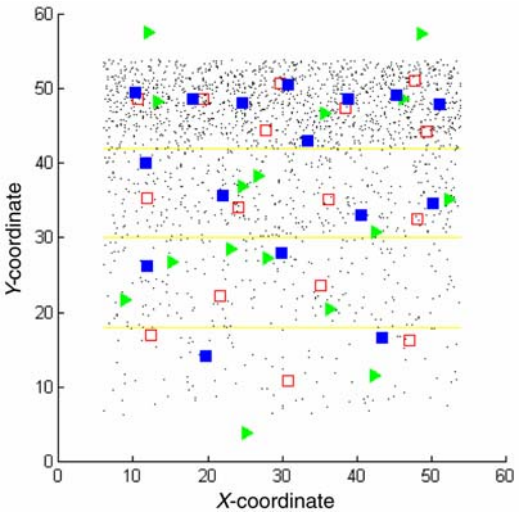
(a)



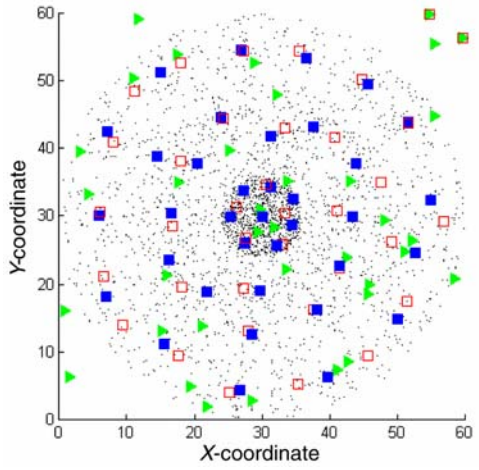
(b)

In the second scenario (Figure 9(b)), radius 1, 3 and 5, respectively. The expected *k*-coverage ratio is 16:4:1 inside out. A total of 36 sensors are deployed. Ideally, the number of sensors should be 12 in each region. After applying the SOM algorithm, the resulting sensors are 8, 14 and 14 in each region. This is somewhat suboptimal. However, one should note that even though we specify event region to be discrete, coverage of sensors is continuous.

Figure 9 Non-uniform event distribution with different coverage requirements (a) Square area and (b) Circular area (see online version for colours)



(a)



(b)

6 Conclusion

In this paper, we formulate the event-driven sensor coverage problem and propose several heuristic methods based on SOMs . Three variants of the SOM method are examined and experimentally evaluated. The results are also compared with the VFA method. Simulation results indicate that the Block-Party-SOM method yields a superior performance in dispersing sensors in a terrain of arbitrary shape so that the density of the sensors approximates the probability density of occurring events, which the sensors must detect. If the probability density of events is assumed uniform then the algorithm will revert to addressing the classic problem of area coverage, also with excellent results. However, the more significant contribution of this work is that it shows an effective way of addressing the coverage problem under the condition that the events to be sensed do not occur with a uniform distribution but an arbitrary one. It is assumed that this event distribution is stationary, but the method is also applicable as is in the case of mobile sensors and non-stationary event distributions that vary slowly relative to the speed of mobility of the sensors.

References

Chellappan, S., Bai, X., Ma, B. and Xuan, D. (2005) 'Sensor network deployment using flip-based sensors', *The 2nd IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, Washington, DC, November.

Cover, T.M. and Thomas, J.A. (1991) *Elements of Information Theory*, 1st edition, New York: Wiley-Interscience.

DARPA/ATO (2002) 'Self-healing minefields', Available at: http://www.darpa.mil/sto/smallunitops/pdf/shm/index.htm

Ghrist, R. and Muhammad, A. (2005) 'Coverage and hole-detection in sensor networks via homology', *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*, Los Angeles, CA.

Haykin, S. (1999) *Neural Networks, a Comprehensive Foundation*, Prentice Hall publisher.

Howard, A., Mataric, M.J. and Sukhatme, G.S. (2002) 'Mobile sensor, network deployment using potential fields: a distributed, scalable solution to the area coverage problem', *Proceedings of the 6th International Symposium Fukuoka*, Japan, 25–27 June 2002.

Kohonen, T. (1982) 'Self-organized formation of topologically correct feature maps', *Biological Cybernetics*, Vol. 43, pp.59–69.

Kohonen, T. (1990) 'The self-organizing map', *Proceedings of IEEE 78*, pp.1464–1480.

Wang, G., Cao, G., and La Porta, T. (2004) 'Movement-assisted sensor deployment', *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Hong Kong, March.

Wang, G., Cao, G., La Porta, T. and Zhang, W. (2005) 'Sensor relocation in mobile networks', *Proceedings of IEEE Infocom*.

Watfa, M. and Commuri, S. (2006) 'Energy efficient approaches to coverage hole detection in wireless sensor networks', *IEEE International Symposium on Intelligent Control*, ISIC, Munich, Germany, October, pp.137–142.

Zou, Y. and Chakrabarty, K. (2003) 'Sensor deployment and target localization based on virtual forces', INFOCOM 2003; 22nd Annual Joint Conf. IEEE Computer and Communications Societies, Vol. 2, pg: 1293- 1303

Appendix analysis of VFA performance

In this section, we provide a rigorous proof that VFA may result in pathologically poor coverage. The key idea is to determine one possible steady state arrangement using VFA.

Lemma 1: *Given a set of n sensors, let* d_{th} *be the threshold distance corresponding to the transition from repellant force to attractive force. There exists a radius r such that when sensors distribute evenly on a circle of radius r, the aggregated force on each sensor is zero.*

Proof: In a circular placement, it is sufficient to examine one sensor and the forces acted on it. Denote this sensor by *v* (see Figure A1). Let *k* be the number of pairs of sensors giving attractive force, excluding the one on the opposite side. Let α_i be the angle between object sensor and *i*th sensor. Clearly, $\alpha_i = (\pi - i \times (2\pi/n))$, $(i = 1,2,...,k)$ and the following equality holds:

$$\sin\frac{\alpha_i}{2} = \sin\left(\frac{\pi - i \times (2\pi/n)}{2}\right) = \cos\left(\frac{\pi}{n} i\right)$$

Nodes on the left side of vertical dash line exert repellant force. And those on the right side give attractive force. Therefore, the distance between K + 1th and sensor *v*, *k*th and object sensor must satisfy the following constraints.

$$\begin{cases} d_a = 2r\sin\dfrac{\alpha_k}{2} = 2r\cos\dfrac{k\pi}{n} > d_{\text{th}} \\ d_{a+1} = 2r\sin\dfrac{\alpha_{k+1}}{2} = 2r\cos\dfrac{(k+1)\pi}{n} < d_{\text{th}} \end{cases} \quad \text{(A1)}$$

Furthermore, in equilibrium state, the total force on sensor *v* should be zero, that is,

$$\sum_{i=1,...,k} |F_{iv}| = \sum_{i=k,...,n} |F_{vi}| \quad \text{(A2)}$$

Using simple algebra, we can obtain a feasible solution for *r* as in Equation (A3):

$$\begin{cases} \dfrac{d_{\text{th}}}{2\cos(k\pi/n)} < \text{r} < \dfrac{d_{\text{th}}}{2\cos(k+1)\pi/n} \\ \text{r} = \dfrac{\left[\left(2\text{t} + 4d_{\text{th}}\sum_{i=1}^{k}\cos(i\pi/n) + \right)\right.}{2(4 + 8\sum_{i=1}^{a}\cos^2(i\pi/n)} \\ \left.\sqrt{\left(2\text{t} + 4d_{\text{th}}\sum_{i=1}^{k}\cos(i\pi/n)^2 + 4(n - 2(k+1))\left(4 + 8\sum_{i=1}^{k}\cos^2(i\pi/n)\right)\right)}\right] \end{cases} \quad \text{(A3)}$$

If we set the threshold to be the same as sensing range as suggested by Zou and Chakrabarty (2003) , that is, $d_{th} = s$, the total coverage of sensors in equilibrium state is upper bounded by

$$n\pi\left[\frac{\left(\left(2t + 4s\sum_{i=1}^{k}\cos(i\pi/n)\right) + \right.}{2\left(4 + 8\sum_{i=1}^{a}\cos^2(i\pi/n)\right)}\right.$$
$$\left.\sqrt{\left(\left(2s + 4s\sum_{i=1}^{k}\cos^2(i\pi/n)\right) + 4(n - 2(k+1))\left(4 + 8\sum_{i=1}^{k}\cos^2(i\pi/n)\right)\right)}\right]^2 \quad \text{(A4)}$$