Fast Sensor Placement Algorithms for Fusion-based Target Detection

Zhaohui Yuan[1,4], Rui Tan[1], Guoliang Xing[2], Chenyang Lu[3], Yixin Chen[3], and Jianping Wang[1]
[1]City University of Hong Kong, HKSAR [2]Michigan State University, USA
[3]Washington University in St. Louis, USA [4]Wuhan University, P.R. China

Abstract

Mission-critical target detection imposes stringent performance requirements for wireless sensor networks, such as high detection probabilities and low false alarm rates. Data fusion has been shown as an effective technique for improving system detection performance by enabling efficient collaboration among sensors with limited sensing capability. Due to the high cost of network deployment, it is desirable to place sensors at optimal locations to achieve maximum detection performance. However, for sensor networks employing data fusion, optimal sensor placement is a non-linear optimization problem with prohibitive computational complexity. In this paper, we present fast sensor placement algorithms based on a probabilistic data fusion model. Simulation results show that our algorithms can meet the desired detection performance with a small number of sensors while achieving up to 7-fold speedup over the optimal algorithm.

1 Introduction

Wireless sensor networks (WSNs) for mission-critical applications (such as target detection [13], object tracking [24], and security surveillance [9]) often face the fundamental challenge of meeting stringent performance requirements imposed by users. For instance, a surveillance application may require any intruder to be detected with a high probability (*e.g.*, $> 90\%$) and a low false alarm rate (*e.g.*, $< 1\%$). Sensor placement plays an important role in the achievable sensing performance of a sensor network. However, finding the optimal sensor placement is challenging because the actual sensing quality of sensors is difficult to predict due to the uncertainty in physical environments. For instance, the measurements of sensors are often contaminated by noise, which renders the detection performance of a network probabilistic.

Most existing works on sensor placement and coverage maintenance are based on *simplistic* sensing models, such as the disc model [1, 17, 21, 22]. In particular, the sensing region of a sensor is modeled as a disc with a certain radius centered at the position of the sensor. A sensor *deterministically* detects the targets/events within its sensing region. Although such a model allows a geometric treatment to the coverage provided by sensors, it fails to capture the stochastic nature of sensing. Moreover, most works based on the disc model do not take advantage of collaboration among sensors.

Data fusion [18] has been proposed as an effective signal processing technique to improve the performance of detection systems. The key advantage of data fusion is to improve the sensing quality by jointly considering the noisy measurements of multiple sensors. For example, real-world experiments using MICA2 motes showed that the false alarm rate of a network is as high as 60% when sensors make their detection decisions independently while the false alarm rate can be reduced to near zero by adopting a data fusion scheme [9]. In practice, many sensor network systems designed for target detection, tracking and classification have employed some kind of data fusion schemes [13, 9, 8].

A key challenge to exploit data fusion in sensor placement is the increased computational cost. When data fusion is employed, the probability of detecting a target is dependent on the measurements of multiple sensors near the target. Therefore, the system detection performance of a fusion-based sensor network has a complex correlation with the spatial distribution of sensors as well as the characteristics of target and environmental noise. As a result, the computational complexity of determining the optimal sensor placement is prohibitively high in moderate to large-scale fusion-based sensor networks.

This paper is focused on developing fast sensor placement algorithms for target detection sensor networks that are designed based on data fusion. In particular, we aim to minimize the number of sensors that for achieving the specified level of sensing performance. The main contributions of this paper are as follows:

- We formulate the sensor placement problem for fusion-based target detection as a constrained optimization problem. Our formulation is based on a probabilistic data fusion model and captures several charac-

teristics of real-world target detection including target signal decay, noisy sensor measurements, and sensors' spatial distribution.

- We develop a global optimal algorithm and two divide-and-conquer algorithms for our sensor placement problem. The optimal algorithm minimizes the number of sensors while meeting specified requirements on system detection probability and false alarm rate. By exploiting the unique structure of the problem, the divide-and-conquer algorithms can find near-optimal solutions at significantly lower computational cost.
- We validate our approach through extensive numerical results as well as simulations based on the real data traces collected in a vehicle detection experiment [8]. Our best algorithm runs up to 7-fold faster than the global optimal algorithm while using a comparable number of sensors in the placement.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the background of data fusion. In Section 4, we formally formulate our sensor placement problem for fusion-based target detection. In Section 5, we present our sensor placement algorithms. We evaluate our algorithms via numerical experiments and trace-driven simulations in Section 6 and Section 7, respectively. Section 8 concludes this paper.

2 Related Work

A number of prior works on sensor placement are focused on minimizing the number of sensors or maximizing the sensing quality provided by a network [5, 1, 17, 21, 22, 3]. However, most of these works adopted the disc sensing model [1, 17, 21, 22]. In contrast, we study the sensor placement problem based on a data fusion model that captures stochastic characteristics of target detection.

Clouqueur *et al.* [3] formulate the sensor placement problem for moving target surveillance based on path exposure, which is computed based on a data fusion model. Different from their work, this paper is focused on detecting stationary targets that may appear at a set of locations. Moreover, we develop both optimal and efficient heuristic sensor placement algorithms. More recently, optimal or approximate algorithms have been proposed to place sensors for monitoring spatially correlated phenomena (such as the temperature in a building) [11, 14, 15, 2]. The sensing models adopted in these works quantify the mutual information [11, 14, 15] and entropy [2] of a continuous phenomena that is observed by sensors. Different from these works, our problem is formulated based on the target detection model that aggregates the noisy measurements of sensors.

There is vast literature on stochastic signal detection based on multi-sensor data fusion. Early work [18] focuses on analyzing optimal fusion strategies for small-scale wired sensor networks (*e.g.*, a handful of radars). Recent work on data fusion [13, 8, 7] have considered the properties of wireless sensor networks such as sensor' spatial distribution and limited sensing capability. In practice, many sensor network systems designed for target detection, tracking and classification [13, 9, 8] have incorporated some kind of data fusion schemes to improve the system performance.

3 Preliminaries

In this section, we describe the background of this work, which includes a single-sensor sensing model and a multi-sensor data fusion model.

3.1 Target and Sensing Model

For many physical signals (*e.g.*, acoustic, seismic, and electromagnetic signals), the energy attenuates with the distance from the signal source. Sensors detect targets by measuring the energy of signals emitted by targets. Denote decreasing function $W(d)$ as the signal energy measured by a sensor which is d meters away from the target. We adopt a signal decay model as follows[1]:

$$W(d) = \begin{cases} \frac{W_0}{(d/d_0)^k} & \text{if} \quad d > d_0 \\ W_0 & \text{if} \quad d \leq d_0 \end{cases} \tag{1}$$

where W_0 is the original energy emitted by the target, k is a decaying factor which is typically from 2 to 5, d_0 is a constant determined by the size of the target and the sensor. This signal attenuation model is widely adopted in the literature [12, 8, 18]. Figure 1 plots the signal energy measurements of an acoustic sensor in the DARPA SensIT vehicle detection experiments [8, 16]. From the figure, we can see that the energy measurement increases linearly with $1/d^2$, which matches the decay model in (1) with $k = 2$.

The measurements of a sensor are corrupted by noise. Denote the noise strength measured by sensor i is N_i, which follows the zero-mean normal distribution with a variance of σ^2, *i.e.*, $N_i \sim \mathcal{N}(0, \sigma^2)$. Suppose sensor i is d_i meters from the target, the signal *energy* it measures is given by

$$U_i = W(d_i) + N_i^2 \tag{2}$$

In practice, the parameters of target and noise models are often estimated using a training dataset before deployment.

[1] Our approach is not dependent on the specific form of $W(d)$.

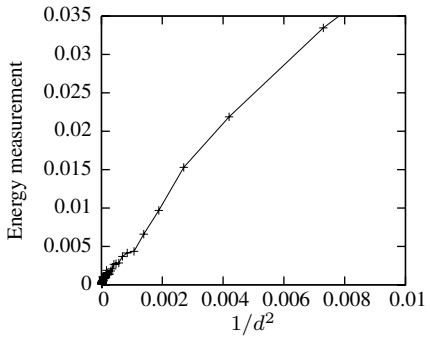


Figure 1. Energy measurement vs. distance. The x-axis is $1/d^2$ (d is the distance from the target, in the unit of meters).

3.2 Multi-sensor Fusion Model

Data fusion [18, 3, 4] is a widely adopted technique for improving the performance of detection systems. A sensor network that employs data fusion is often organized into clusters. Each *cluster head* is responsible for making a final decision regarding the presence of target by fusing the information gathered by member sensors in the cluster.

We adopt a data fusion scheme as follows. Sensors send their energy measurements to the cluster head, which in turn compares the average of all measurements against a threshold η. If the average is greater than η, the cluster head decides that a target is present. Otherwise, it decides there is no target. The threshold η is referred to as the *detection threshold*.

The performance of a detection system is usually characterized by false alarm rate and detection probability. False alarm rate (denoted by P_F) is the probability of making a positive detection decision when no target is present. Detection probability (denoted by P_D) is the probability that a target is correctly detected. Suppose n sensors take part in the data fusion. Under the aforementioned value fusion scheme, the false alarm rate is given by:

$$P_F = \mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n} N_i^2 > \eta\right) = 1 - \mathbb{P}\left(\sum_{i=1}^{n}\left(\frac{N_i}{\sigma}\right)^2 \leq \frac{n\eta}{\sigma^2}\right)$$

As $N_i/\sigma \sim \mathcal{N}(0,1)$, $\sum_{i=1}^{n}(N_i/\sigma)^2$ follows the Chi-square distribution with n degrees of freedom whose Cumulative Distribution Function (CDF) is denoted as $\mathcal{X}_n(\cdot)$. Hence, P_F can be calculated by:

$$P_F = 1 - \mathcal{X}_n\left(\frac{n\eta}{\sigma^2}\right) \tag{3}$$

Similarly, the detection probability can be calculated by:

$$\begin{aligned} P_D &= \mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}\left(W(d_i) + N_i^2\right) > \eta\right) \\ &= 1 - \mathbb{P}\left(\sum_{i=1}^{n}\left(\frac{N_i}{\sigma}\right)^2 \leq \frac{n\eta - \sum_{i=1}^{n} W(d_i)}{\sigma^2}\right) \\ &= 1 - \mathcal{X}_n\left(\frac{n\eta - \sum_{i=1}^{n} W(d_i)}{\sigma^2}\right) \end{aligned} \tag{4}$$

4 Sensor Placement Problem for Fusion-based Target Detection

In this section, we formulate the sensor placement problem for fusion-based target detection. In Section 4.1, we introduce the network model and assumptions. In Section 4.2, we formally formulate the problem. The complexity of problem is reduced in Section 4.3.

4.1 Network Model and Assumptions

We assume that targets appear at a set of known physical locations referred to as *surveillance spot*s, or *spot*s for brief. We are only concerned with the sensor placement for surveillance spots. Surveillance spots are often chosen before network deployment according to application requirements. For instance, in fire detection applications, the surveillance spots can be chosen at the venues with inflammables. In intruder detection applications which require the surveillance over a geographic region, the spots can be chosen densely and uniformly in the region.

Due to the spatial decay of signal energy, the sensors far away from the target experience low Signal-to-Noise Ratio (SNR) and hence make little contribution to the detection. Therefore, we assume that only the sensors close to a surveillance spot participate in the data fusion. For any surveillance spot, we define the *fusion region* as the disc of radius R centered at the spot. The radius R is referred to as *fusion radius* hereafter. Fusion radius plays an important role in the detection performance of a network. On one hand, a conservative fusion radius confines sensors' detection capability despite they may contribute to the surveillance spots outside the fusion radius. On the other hand, a large fusion radius may result in poor detection performance by fusing the irrelevant measurements from distant sensors. The optimal fusion radius is dependent on network density and characteristics of targets and noise. The detailed analysis of the optimal fusion radius is omitted here due to the space limitation and can be found in [23].

Sensors within the fusion region of each surveillance spot forms a cluster to detect whether a target is present at the surveillance spot by comparing the average of all energy

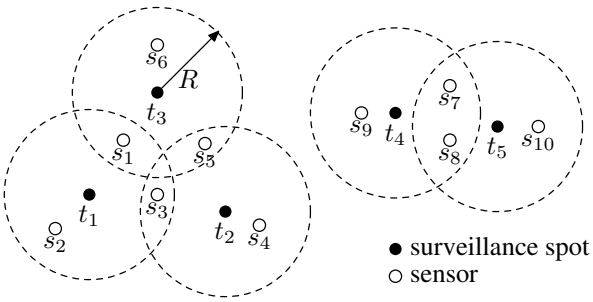


Figure 2. Network model. Dotted circles represent the fusion regions of different surveillance spots.

measurements of the sensors in the cluster with a threshold, as described in Section 3.2. A cluster head is selected to perform data fusion for each detection cluster. For instance, the sensor closest to the surveillance spot may serve as the cluster head. We introduce the following definition.

Definition 1 *A sensor is a* dedicated sensor *if it is only within the fusion region of a surveillance spot; a sensor is a* shared sensor *if it is within the fusion regions of at least two surveillance spots.*

Figure 2 illustrates the network model. In the figure, the dotted circles represent the fusion regions of different surveillance spots. For example, the measurements of s_1, s_2 and s_3 are fused to detect whether a target is present at spot t_1. Sensors s_1, s_3, s_5, s_7, s_8 are shared sensors while s_2, s_4, s_6, s_9, s_{10} are dedicated sensors. Note that a shared sensor reports its measurements to multiple cluster heads.

4.2 Problem Formulation

We define the following notation before we formally formulate the problem.

1. A represents the surveillance field where total K surveillance spots are located. $\mathbf{T} = \{t_j | 1 \leq j \leq K\}$ represents the set of surveillance spots, where $t_j = (x_j, y_j) \in A$ is the coordinates of the j^{th} surveillance spot.
2. C_j, n_j and η_j are the fusion region of t_j, the number of sensors within C_j, and the detection threshold for t_j, respectively.
3. $\mathbf{S} = \{s_i | 1 \leq i \leq N\}$ represents the *sensor placement*, where $s_i = (x_i, y_i) \in A$ is the coordinates of the i^{th} sensor[2] and N is the total number of sensors. $|\mathbf{S}|$ is the cardinality of $\mathbf{S}$, *i.e.*, $|\mathbf{S}| = N$.
4. P_{F_j} and P_{D_j} are the false alarm rate and detection probability of t_j, which can be calculated by (3) and (4), respectively.

[2]As a slight abuse of the notation, s_i (t_i) refers to both the ID and the coordinate pair of the i^{th} sensor (surveillance spot).

We quantify the performance of target detection by a new metric called *(α,β)-coverage*, which is defined as follows.

Definition 2 ((α,β)-coverage) *Given two real numbers, $\alpha \in (0,1)$ and $\beta \in (0,1)$, the surveillance spot t_j is (α,β)-*covered *if $P_{F_j} \leq \alpha$ and $P_{D_j} \geq \beta$.*

The (α,β)-coverage defines the sensing quality provided by the network at a surveillance spot. Our problem is to place the fewest sensors to provide (α,β)-coverage at all surveillance spots. Our problem is formulated as follows.

Problem 1 *Given a surveillance field A and a set of surveillance spots $\mathbf{T}$, find a list of detection thresholds $\{\eta_j | 1 \leq j \leq K\}$ and a sensor placement $\mathbf{S}$ such that the number of sensors $|\mathbf{S}|$ is minimized subject to the constraint that each surveillance spot in $\mathbf{T}$ is (α,β)-covered.*

A straightforward solution to the problem is to check whether N (starting with $N = 1$) sensors are enough to cover all surveillance spots. Besides N, the variables to be determined include the sensor positions $\mathbf{S} = \{(x_i, y_i) | 1 \leq i \leq N\}$ and the detection thresholds $\{\eta_j | 1 \leq j \leq K\}$. Hence, the total number of variables is $1 + 2N + K$. Moreover, there exists a complex nonlinear relationship between the detection performance (*i.e.*, P_{F_j} and P_{D_j}, $1 \leq j \leq K$) and these variables. The exhaustive search to all the variables incurs prohibitively high computational cost even for small-scale networks. As the first step to dealing with the computational complexity, we seek to reduce the number of variables in the problem formulation in the next section.

4.3 Problem Reduction

The number of variables in our problem formulation can be reduced to $1 + 2N$ by exploiting the optimal detection thresholds. For a certain sensor placement, $P_{F_j} \leq \alpha$ is a necessary condition of the (α,β)-coverage of surveillance spot t_j. From (3), this necessary condition becomes $\eta_j \geq \frac{\sigma^2 \mathcal{X}_n^{-1}(1-\alpha)}{n_j}$, where $\mathcal{X}_n^{-1}(\cdot)$ is the inverse function of $\mathcal{X}_n(\cdot)$. Furthermore, according to (4), P_{D_j} decreases with η_j. In other words, the detection threshold should be minimized in order to maximize the detection probability of a sensor placement. Therefore, the detection threshold of a sensor placement can always be set to its lower bound as follows.

$$\eta_j = \frac{\sigma^2 \mathcal{X}_n^{-1}(1-\alpha)}{n_j} \tag{5}$$

We compute the optimal detection threshold of each surveillance spot by (5). Using the optimal detection thresholds, each surveillance spot is (α,β)-covered if and only if the detection probability for each spot is greater than β, or equivalently, $\min_{1 \leq j \leq K}\{P_{D_j}\} \geq \beta$. Hence, only $1 + 2N$ variables need to be determined, *i.e.*, N and $\mathbf{S} = \{(x_i, y_i) | 1 \leq i \leq N\}$. Accordingly, Problem 1 is simplified as follows.

Algorithm 1 The procedure of finding global optimal solution

Input: α, β, surveillance field A, a set of surveillance spots $\mathbf{T}$
Output: Sensor placement $\mathbf{S}$ where $|\mathbf{S}|$ is minimized

1: $N = 1$
2: **repeat**
3: use the CSA solver to find N sensor locations in A that maximize $\min_{1\leq j\leq K}\{P_{D_j}\}$
4: compute η_j for each $t_j \in \mathbf{T}$ by (5)
5: compute P_{D_j} for each $t_j \in \mathbf{T}$ by (4)
6: $N = N + 1$
7: **until** $\min_{1\leq j\leq K}\{P_{D_j}\} \geq \beta$
8: **return** $\mathbf{S}$

Problem 2 *Given a surveillance field A and a set of surveillance spots $\mathbf{T}$, find a sensor placement $\mathbf{S}$ such that the number of sensors $|\mathbf{S}|$ is minimized subject to the following constraint:*

$$\min_{1\leq j\leq K}\{P_{D_j}\} \geq \beta \tag{6}$$

5 Sensor Placement Algorithms

In Section 5.1, we present an optimal sensor placement algorithm for Problem 2. However, this algorithm cannot handle large-scale problems due to high time complexity. In Section 5.2 and 5.3, we propose an efficient divide-and-conquer approach and a corresponding heuristic placement algorithm, respectively. In Section 5.4, we discuss an extension to the divide-and-conquer algorithm by integrating a clustering algorithm.

5.1 Global Optimal Placement

A straightforward optimal solution for Problem 2 is to incrementally search for the optimal sensor placements with different number of sensors under constraint (6). The details are shown in Algorithm 1. It begins with $N = 1$ and iterates for incremental N. In each iteration, the minimum detection probability among all surveillance spots, *i.e.*, $\min_{1\leq j\leq K}\{P_{D_j}\}$, is maximized. Once constraint (6) is satisfied, the global optimal solution is found.

The optimization step that maximizes the minimum detection probability (Line 3 in Algorithm 1) is implemented by a nonlinear programming solver based on the Constrained Simulated Annealing (CSA) algorithm [20]. CSA extends conventional Simulated Annealing to look for the global optimal solution of a constrained optimization problem with discrete variables. CSA allows the objective function and constraint functions to be specified in a procedure instead of in a closed-form. Theoretically, CSA is a global optimal algorithm that converges asymptotically to a constrained global optimum with probability one (Theorem 1 of [20]).

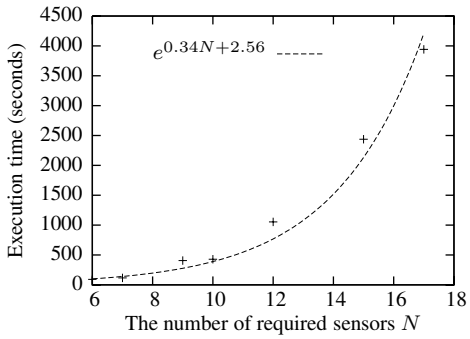


Figure 3. The execution time vs. the number of sensors in the optimal placement.

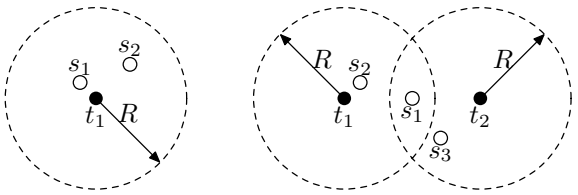
In theory, the complexity of CSA, like other stochastic search algorithms, increases exponentially with respect to the number of variables [20]. Therefore, the nonlinear programming solver has an exponential time complexity with respect to the number of sensors. Hence, for a large-scale placement problem, Algorithm 1 becomes prohibitively expensive.

Figure 3 shows the execution time of Algorithm 1 versus the number of sensors in the optimal solution. The dotted curve is the linear regression of the execution times with different numbers of sensors. We can see that the execution time increases drastically with the number of sensors. For instance, if 100 sensors are to be placed, the projected execution time of the global optimal algorithm is about e^{36} seconds, *i.e.*, 5×10^{10} days.

5.2 Divide-and-Conquer Approach

As described in Section 5.1, all the surveillance spots are processed at one optimization step (Line 3) of Algorithm 1. That is, the positions of sensors are optimized by the CSA solver to cover all the surveillance spots. This strategy significantly increases the computational cost of Algorithm 1 because the execution time of the CSA solver increases exponentially with respect to the total number of sensors that are placed.

In this section, we propose a divide-and-conquer approach to reduce the computational complexity of the global optimal algorithm. A straightforward divide-and-conquer approach is to cover surveillance spots one by one using the CSA solver and then combine all local solutions into a global solution. As the cost for finding each local solution is small, the overall time complexity will be polynomial with respect to the number of surveillance spots. However, a key challenge for implementing this approach is that the local problems (*i.e.*, sensor placements for individual surveillance spots) are *dependent*. This is because the shared sensors contribute to the detection performance of multiple fusion regions. As a result, solving the local problems separately without considering the interdependence between lo-



(a) A spot is covered by two sensors

(b) Two spots are covered by only three sensors due to the fusion region overlap

Figure 4. An example of interdependence between local sensor placements ($W_0 = 0.65$, $d_0 = 1$, $k = 2$, $\sigma^2 = 0.1$, $\alpha = 0.01$, $\beta = 0.9$, $R = 1.6$ m).

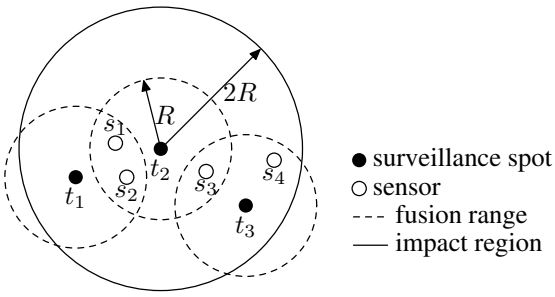


Figure 5. An example of the divide-and-conquer approach.

cal solutions may result in an inefficient global solution. We now illustrate this issue using the following example.

In this example, we use the following parameters for the target and sensing model (defined in Section 3.1): $W_0 = 0.65$, $d_0 = 1$, $k = 2$, $\sigma^2 = 0.1$. We aim to achieve $(0.01, 0.9)$-coverage for each surveillance spot, *i.e.*, $\alpha = 0.01$ and $\beta = 0.9$. If there exists only one spot t_1, two sensors are required, as shown in Figure 4(a). Similarly, if two spots t_1 and t_2 far away from each other are to be covered, we need to place two sensors to cover each of them. When t_1 and t_2 are 1.2 meters apart, their fusion regions overlap. In such a case, three sensors are found by the global optimal solution to cover t_1 and t_2, as shown in Figure 4(b). In the optimal placement, there are two dedicated sensors (s_2, s_3) and one shared sensor (s_1). This example shows that the number of sensors can be reduced by exploiting the overlaps between the fusion regions of nearby surveillance spots. However, if the coverage of each surveillance spot is treated separately, four sensors will be placed. Such inefficiency is the result of ignoring the interdependence between local solutions.

We now describe the basic idea of our divide-and-conquer approach. We define the *impact region* of a surveillance spot as the disc of radius $2R$ centered at the spot, as illustrated in Figure 5. We denote the impact region of t_j as A_j. Any surveillance spot that falls in the impact region of t_j shares part of the fusion region with it. In our approach, the surveillance spots are covered one by one in iterations. When t_j is processed, we first check if the sensors that are placed within A_j in previous iterations can cover t_j and all the surveillance spots within A_j, and additional sensors are then placed if necessary. The key idea of this approach is to reduce the total number of sensors in the global solution by taking advantage of the shared sensors that appear in multiple local solutions. We now illustrate this approach using the following example. Three surveillance spots need to be covered in Figure 5. We first compute a local solution for the impact region of t_1 so that both t_1 and t_2 are covered. In the second iteration, we compute a local solution for the impact region of t_2 to cover t_2 and t_3. As t_2 has been covered by the previous local solution, we only need to place additional sensors in the fusion region of t_3.

5.3 Divide-and-Conquer Sensor Placement

In this section, we present our divide-and-conquer sensor placement algorithm in detail. The pseudo code is given in Algorithm 2. In the divide step, for each surveillance spot t_j, we find the set of spots within the impact region of t_j, which is denoted as $\mathbf{T}_j$. In the conquer step, we compute an optimal local solution for each surveillance spot t_j. In the j^{th} iteration (from Line 3 to 9), the algorithm ensures that spot t_j and all neighboring spots in $\mathbf{T}_j$ are covered using the fewest additional sensors that are placed only in the fusion region of t_j. The optimization step in Line 5 is implemented by the CSA solver discussed in Section 5.1.

Note that shared sensors are favored over dedicated sensors by the optimization process as they can significantly reduce the number of sensors required to cover multiple surveillance spots (including t_j and all spots within its impact region). However, as these sensors are only placed in the fusion region of t_j (Line 5), the detection performance of other surveillance spots outside of A_j will not be affected. After K iterations, all surveillance spots are covered and the algorithm terminates.

A key advantage of Algorithm 2 is that the sensors placed in previous iterations can be reused by the current local solution. Specifically, at Line 5 of Algorithm 2, a sensor placement is computed to cover not only the current surveillance spot but also those within its impact region. As a result, the sensors that are already placed in the shared fusion regions can be utilized for covering the current spot. However, a shortcoming of this strategy is that the local sensor placement of a surveillance spot may become less efficient as more shared sensors are placed to cover neighboring spots in later iterations. In the extreme case, a dedicated sensor may become redundant if the shared sensors placed in later iterations are enough to cover the spot. The cause of this issue is the interdependence between local solutions.

We now describe an improved algorithm to deal with this issue. The pseudo code is given in Algorithm 3. In each

Algorithm 2 The divide-and-conquer sensor placement algorithm

Input: α, β, impact region set $\{A_j|1 \leq j \leq K\}$, and the set of surveillance spots in each impact region $\{\mathbf{T}_j|1 \leq j \leq K\}$
Output: Local optimal sensor placement $\mathbf{S}$

1: $\mathbf{S} = \emptyset$
2: **for** $j = 1$ to K **do**
3: $n = 0$
4: **repeat**
5: place additional n sensors in C_j (denoted by set Δ) to maximize $\min_{t_h \in \mathbf{T}_j}\{P_{D_h}\}$ under placement $\mathbf{S} \cup \Delta$
6: compute η_h for each $t_h \in \mathbf{T}_j$ under placement $\mathbf{S} \cup \Delta$
7: compute P_{D_h} for each $t_h \in \mathbf{T}_j$ under placement $\mathbf{S} \cup \Delta$
8: $n = n + 1$
9: **until** $\min_{t_h \in \mathbf{T}_j}\{P_{D_h}\} \geq \beta$
10: $\mathbf{S} = \mathbf{S} \cup \Delta$
11: **end for**
12: **return** $\mathbf{S}$

round of Algorithm 3 (from Line 2 to 20), all surveillance spots are processed one by one. The algorithm terminates if the current round cannot further reduce the number of sensors in the placement. When the j^{th} spot t_j is processed (from Line 4 to 19), the algorithm first removes all dedicated sensors of t_j, and computes a new local placement $\mathbf{S}'$ using the CSA solver (from Line 8 to 15). If $\mathbf{S}'$ uses fewer sensors than the original placement $\mathbf{S}$, we replace the original placement with $\mathbf{S}'$ (from Line 17 to 19). Note that we do not remove any shared sensors in the placement computed by Algorithm 2 as otherwise the coverage of neighboring surveillance spots may be affected.

We now discuss the convergence of Algorithm 3. As only the new placement with fewer sensors (from Line 17 to 19) in each iteration is acceptable, obviously, the size of the placement computed in each round keeps decreasing. Denote N^* and N_0 as the sizes of the global optimal solution (*i.e.*, the output of Algorithm 1) and the solution of Algorithm 2, respectively. The upper bound of the number of rounds of Algorithm 3 is thus $N_0 - N^*$.

5.4 Cluster-based Divide-and-Conquer

In this section, we discuss a cluster-based divide-and-conquer approach that improves the performance of Algorithm 3 in large-scale dense sensor networks. In Algorithm 3, the CSA solver may be invoked more than once to optimize the sensor placement of each surveillance spot. Specifically, suppose M represents the number of rounds before the termination of Algorithm 3, the CSA solver is invoked for total $(1 + M) \cdot K$ times, which incurs high computational cost when the number of surveillance spots (K) is large. Moreover, once a sensor is placed within the shared fusion region between two spots, its position remains unchanged. Although this property is key to ensure the convergence of Algorithm 3, it may result in inefficient sensor placement. This is because the sensor placement that is initially optimal for a spot may become suboptimal as more shared sensors are placed within the fusion region of the spot (Line 11).

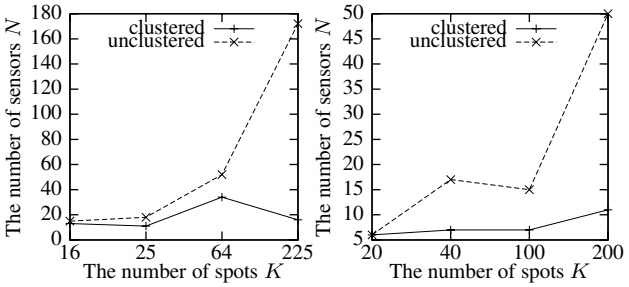
Algorithm 3 The improved divide-and-conquer sensor placement algorithm

Input: α, β, impact region set $\{A_j|1 \leq j \leq K\}$, the set of surveillance spots in each impact region $\{\mathbf{T}_j|1 \leq j \leq K\}$, sensor placement $\mathbf{S}$ computed by Algorithm 2
Output: new sensor placement $\mathbf{S}$

1: **repeat**
2: total sensor number $N = |\mathbf{S}|$
3: **for** $j = 1$ to K **do**
4: find dedicated sensor set D_j of t_j in $\mathbf{S}$
5: **if** $D_j = \emptyset$ **then**
6: skip this iteration for t_j and continue
7: **else**
8: $\mathbf{S}' = \mathbf{S} \setminus D_j$ /* *remove dedicated sensors of* t_j */
9: $n = 1$
10: **repeat**
11: place additional n sensors in C_j (denoted by Δ) to maximize $\min_{t_h \in \mathbf{T}_j}\{P_{D_h}\}$ under placement $\mathbf{S}' \cup \Delta$
12: compute η_h and P_{D_h} for each $t_h \in \mathbf{T}_j$ under placement $\mathbf{S}' \cup \Delta$
13: $n = n + 1$
14: **until** $\min_{t_h \in \mathbf{T}_j}\{P_{D_h}\} \geq \beta$
15: $\mathbf{S}' = \mathbf{S}' \cup \Delta$
16: **end if**
17: **if** $|\mathbf{S}'| < |\mathbf{S}|$ **then**
18: $\mathbf{S} = \mathbf{S}'$
19: **end if**
20: **end for**
21: **until** $|\mathbf{S}| = N$
22: **return** $\mathbf{S}$

We now describe a clustering scheme that not only reduces Algorithm 3's overhead but also improves the quality of sensor placement. The surveillance spots are grouped into clusters according to their proximity. The CSA solver is then executed for each cluster of spots. We employ a greedy clustering algorithm called the Quality Threshold (QT) algorithm proposed by Heyer *et al.* [10]. The objective of QT is to organize the surveillance spots into clusters whose members are geographically close to each other while minimizing the total number of clusters. Specifically, in each iteration, a candidate cluster is created to center at each unclustered spot t with the spots within the impact region of t as members. The candidate with the most members is kept as a real cluster. The clustering procedure continues until there is no unclustered spot left. More details of QT can be found in [10].

We define the impact region of each cluster as the disc of radius $2R$ centered at the cluster head (which is identi-



(a) Regular spots (b) Random spots

Figure 6. The number of sensors vs. the number of surveillance spots

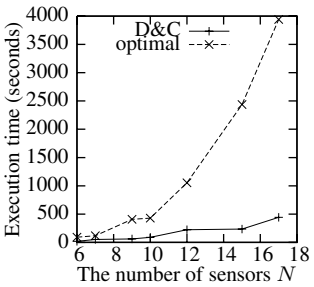


Figure 7. Execution time vs. the number of sensors placed

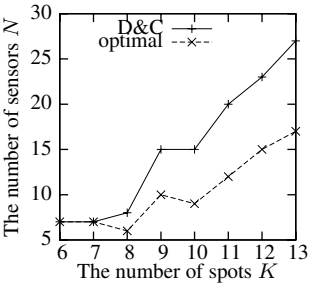


Figure 8. The number of sensors placed vs. the number of spots

fied by QT). Suppose QT yields total L clusters. Denote A_l as the impact region of the cluster whose cluster head is t_l, and $\mathbf{T}_l$ as the set of surveillance spots in A_l. Accordingly, $\{A_i|1 \leq i \leq K\}$ and $\{\mathbf{T}_i|1 \leq i \leq K\}$ in Algorithm 2 and Algorithm 3 should be replaced with $\{A_l|1 \leq i \leq L\}$ and $\{\mathbf{T}_l|1 \leq i \leq L\}$, respectively. When the surveillance spots are densely distributed, L is much smaller than K. Hence, the number of invocations of the CSA solver is reduced from $(1+M)\cdot K$ to $(1+M)\cdot L$. Moreover, the surveillance spots close to each other are always clustered together and their sensor placement is jointly optimized, which can significantly reduce the total number of sensors.

6 Numerical Results

In this section, we conduct numerical experiments to evaluate the performance of the sensor placement algorithms proposed in Section 5. The impact of surveillance spot clustering is evaluated in Section 6.1. We compare the divide-and-conquer placement algorithm with the global optimal algorithm and a greedy algorithm in Section 6.2.

The parameters of the signal decay and noise models are set as follows: $W_0 = 400$, $d_0 = 1$, $k = 2$, $\sigma^2 = 1$. The surveillance field A is a 30×30 m^2 square area. The surveillance spots are chosen regularly (*i.e.*, on regular grid points) or randomly. The bounds of false alarm rate and detection probability (*i.e.*, α and β) are set to be 1% and 90%, respectively. The data fusion range is set to 7.76 m, which is the optimal value derived in our analysis [23].

6.1 Impact of Clustering

We first evaluate the impact of the spot clustering. We run the divide-and-conquer algorithm with and without QT clustering for total 4 regular and 4 random layouts, respectively. The results are shown in Figure 6(a) and Figure 6(b). Figure 6(a) plots the number of placed sensors versus the number of surveillance spots regularly distributed.

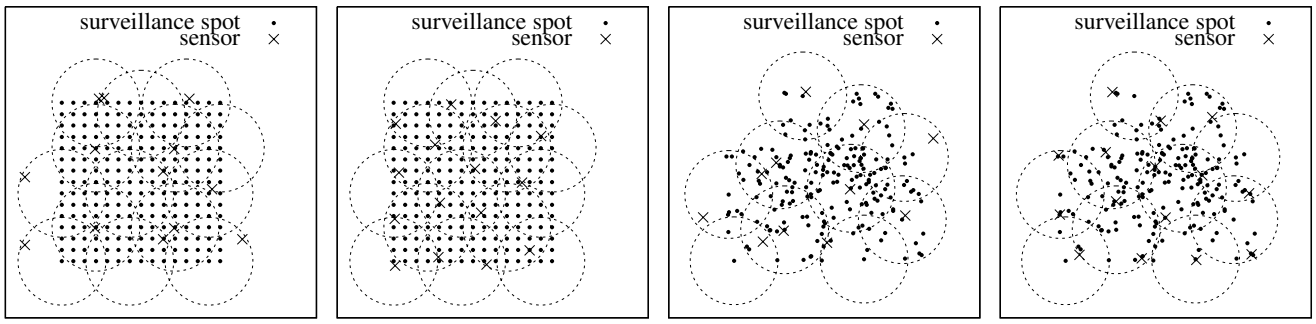
The curve labeled with "clustered" and "unclustered" represents the results computed by the divide-and-conquer placement algorithm with and without QT clustering, respectively. Figure 6(b) shows the results for random surveillance spots. We can see that the cluster-based placement algorithm can effectively reduce the number of required sensors. For instance, in Figure 6(a), when there are total $15 \times 15 = 225$ surveillance spots, 172 sensors are needed without clustering while only 13 sensors are needed when clustering is employed. Another interesting observation is that the number of sensors required does not increase considerably with the number of surveillance spots. For instance, in Figure 6(b), total 7 sensors are enough to cover 100 surveillance spots, and total 11 sensors are enough to cover 200 surveillance spots.

6.2 Sensor Placement Performance

We now compare our divide-and-conquer algorithm against the global optimal algorithm for small networks. Figure 7 shows the execution time of different algorithms versus the number of sensors placed for seven random surveillance spot layouts. We can see that the divide-and-conquer algorithm (labeled as D&C in Figure 7) is up to 7-fold faster than the global optimal algorithm. Meanwhile, Figure 8 shows that the divide-and-conquer algorithm places a comparable number of sensors as the optimal algorithm.

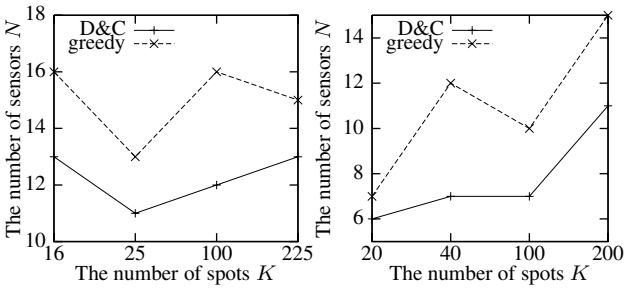
We also compare our divide-and-conquer algorithm against a greedy algorithm in large-scale networks. In the greedy algorithm, sensors are also organized into detection clusters centered at surveillance spots. In each iteration, we place a sensor randomly in the fusion region of the surveillance spot that has the minimum detection probability. The algorithm terminates when every surveillance spot is (α,β)-covered. This algorithm is similar to several greedy sensor placement algorithms employed in previous work [3, 6, 19].

In the first set of experiments, total $15 \times 15 = 225$ surveillance spots are regularly distributed, as shown in Figure 9(a) and Figure 9(b). The sensor placements computed by the cluster-based divide-and-conquer algorithm and the



(a) D&C: $K = 225$, $N = 13$ (b) Greedy: $K = 225$, $N = 15$ (c) D&C: $K = 196$, $N = 11$ (d) Greedy: $K = 196$, $N = 15$

Figure 9. Sensor placements using the cluster-based divide-and-conquer and the greedy algorithms. The dotted circles are the impact regions of clusters.



(a) Regular spots (b) Random spots

Figure 10. The number of sensors vs. the number of surveillance spots

greedy algorithm are also shown. Total 13 and 15 sensors are placed by the two algorithms, respectively. In the second set of experiments, total 196 surveillance spots randomly scatter in the field, as shown in Figure 9(c) and Figure 9(d). Total 11 and 15 sensors are placed, respectively.

Figure 10(a) shows the results of 4 random layouts with 4×4, 5×5, 10×10, and 15×15 spots. Figure 10(b) shows the results of four random layouts with 20, 40, 100 and 200 spots. We can see that our cluster-based divide-and-conquer algorithm consistently outperforms the greedy algorithm in all layouts. The average performance gain is about 30%.

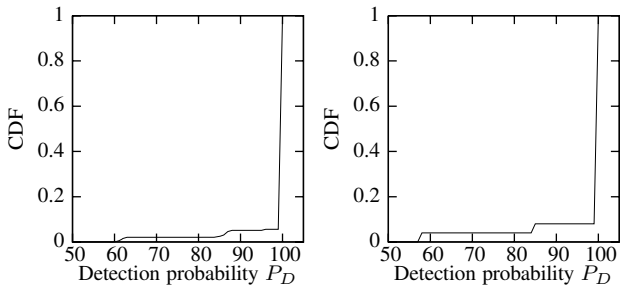
7 Trace-driven Simulations

We conduct extensive simulations using the real data traces collected in the DARPA SensIT vehicle detection experiments [8]. In the experiments, 75 WINS NG 2.0 nodes are deployed to detect military vehicles driving through several intersecting roads. We refer to [8] for detailed setup of the experiments. The dataset used in our simulations includes the ground truth data and the acoustic signal energy measurements recorded by 17 nodes at a sampling period of 0.75 seconds, when an Assault Amphibian Vehicle (AAV) drives through a road. The ground truth data include the positions of sensors and the track of the AAV recorded by a GPS device.

The data traces used in our simulations include the time series recorded for 9 vehicles (AAV3-11). We use the data trace of AAV3 as the training dataset for estimating the energy decay model. The estimated parameters of the signal decay and noise models are: $W_0 = 0.51$ (after normalization), $d_0 = 2.6\,\text{m}$, $k = 2$, $\sigma^2 = 0.05$. In our simulations, the surveillance field A is a $30 \times 30\,\text{m}^2$ square area. The bounds of false alarm rate and detection probability (*i.e.*, α and β) are set to be 1% and 90%, respectively.

As the real data are collected for moving targets, they can not be directly used in our simulations. For each energy measurement, we compute the distance between the sensor and the AAV from the ground truth data. When a sensor makes a measurement in simulation, the energy is set to be the real measurement gathered at the same distance to the AAV. We note that our simulations account for several realistic factors. For instance, there exists considerable deviation between the measurements of sensors and the theoretic signal decay model estimated by the training data. This deviation is due to various reasons including the changing noise levels caused by wind.

We evaluate the performance of the cluster-based sensor placement algorithm in two sets of simulations. First, 67 sensor are placed to cover 196 surveillance spots regularly distributed at 14×14 grid points. Second, 10 sensors are placed to cover 25 surveillance spots randomly scattered in the field. We evaluate the coverage of each surveillance spot as follows. For each surveillance spot, a target appears for 1000 times and the detection probability is calculated as the ratio of the number of successful detections to the number of appearances of the target. Figure 11(a) and Figure 11(b) show the Cumulative Distribution Function (CDF) of the detection probability under the two sensor placements, respectively. From the figures, we can see that over 90% surveillance spots are covered in both two placements, which satisfies the required lower bound of detection probability.



(a) 196 regular spots (b) 25 random spots

Figure 11. The CDF of P_D

8 Conclusion

Data fusion is an effective technique for improving system sensing performance by enabling efficient collaboration among sensors. In this paper, we present an optimal sensor placement algorithm and two divide-and-conquer heuristics for fusion-based target detection. Although the optimal algorithm can minimize the number of sensors in a placement, it incurs high computational complexity in large-scale networks. By exploiting the unique structure of the problem, the divide-and-conquer algorithms can find near-optimal solutions at significantly lower cost. We validate our approach through extensive numerical results as well as simulations based on the real data traces collected in a vehicle detection experiments. Our best algorithm runs up to 7-fold faster than the optimal algorithm while using in a comparable number of sensors in the placement.

Acknowledgment

The work described in this paper was partially supported by the Research Grants Council of Hong Kong under grant RGC 9041266, the Department of Energy, USA, under grant DOE ER-25737 and the National Science Foundation, USA, under grant NSF IIS-0713109. We thank Ke Shen for contributing to the early simulations of this work.

References

[1] K. Chakrabarty, S. Iyengar, H. Qi, and E. Cho. Grid Coverage for Surveillance and Target Location in Distributed Sensor Networks. *IEEE Trans. Comput.*, 51(12), 2002.

[2] H. Choi, J. How, and P. Barton. An Outer-Approximation Algorithm for Generalized Maximum Entropy Sampling. In *American Control Conference*, 2008.

[3] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. K. Saluja. Sensor deployment strategy for target detection. In *WSNA*, 2002.

[4] T. Clouqueur, K. K. Saluja, and P. Ramanathan. Fault tolerance in collaborative sensor networks for target detection. *IEEE Trans. Comput.*, 53(3), 2004.

[5] S. Dhillon, K. Chakrabarty, and S. S. Iyengar. Sensor placement for grid coverage under imprecise. In *FUSION*, 2002.

[6] S. S. Dhillon and K. Chakrabarty. Sensor placement for effective coverage and surveillance in distributed sensor networks. In *WCNC*, 2003.

[7] M. Duarte and Y. H. Hu. Distance based decision fusion in a distributed wireless sensor network. In *IPSN*, 2003.

[8] M. F. Duarte and Y. H. Hu. Vehicle classification in distributed sensor networks. *J. Parallel and Distributed Computing*, 64(7), 2004.

[9] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys*, 2004.

[10] L. Heyer, S. Kruglyak, and S. Yooseph. Exploring expression data: Identification and analysis of coexpressed genes. *Genome Research*, 9(11), 1999.

[11] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. In *IPSN*, 2006.

[12] D. Li and Y. H. Hu. Energy based collaborative source localization using acoustic micro-sensor array. *EUROSIP J. Applied Signal Processing*, (4), 2003.

[13] D. Li, K. Wong, Y. H. Hu, and A. Sayeed. Detection, classification and tracking of targets in distributed sensor networks. *IEEE Signal Process. Mag.*, 19(2), 2002.

[14] P. Liaskovits and C. Schurgers. Leveraging Redundancy in Sampling-Interpolation Applications for Sensor Networks. In *DCOSS*, 2007.

[15] M. Osborne, S. Roberts, A. Rogers, S. Ramchurn, and N. Jennings. Towards Real-Time Information Processing of Sensor Network Data Using Computationally Efficient Multi-output Gaussian Processes. In *IPSN*, 2008.

[16] X. Sheng and Y. H. Hu. Energy based acoustic source localization. In *IPSN*, 2003.

[17] D. Tian and N. D. Georganas. A coverage-preserved node scheduling scheme for large wireless sensor networks. In *WSNA*, 2002.

[18] P. Varshney. *Distributed Detection and Data Fusion*. Spinger-Verlag, 1996.

[19] L. F. M. Vieira, M. A. M. Vieira, L. R. Beatriz, A. A. F. Loureiro, D. C. da Silva Junior, and A. O. Fernandes. Efficient incremental sensor network deployment algorithm. In *Brazilian Symposium on Computer Networks*, 2004.

[20] B. W. Wah, Y. Chen, and T. Wang. Simulated annealing with asymptotic convergence for nonlinear discrete constrained global optimization. *J. Global Optimization*, 39, 2007.

[21] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. D. Gill. Integrated coverage and connectivity configuration in wireless sensor networks. In *Sensys*, 2003.

[22] T. Yan, T. He, and J. A. Stankovic. Differentiated surveillance for sensor networks. In *Sensys*, 2003.

[23] Z. Yuan, R. Tan, G. Xing, C. Lu, Y. Chen, and J. Wang. Fast sensor placement algorithms for fusion-based target detection. Technical Report MSU-CSE-08-29, Michigan State University, 2009.

[24] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration for tracking applications. *IEEE Signal Process. Mag.*, 19(2), 2002.