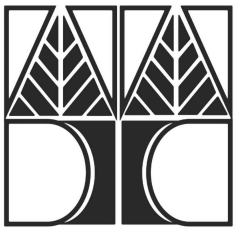
DISTRIBUTED EVENT DETECTION AND LOCALIZATION IN WIRELESS SENSOR NETWORKS

by

Michalis P. Michaelides

B.S. (1997) and M.S. (1998), in Electrical Engineering from Purdue Univ., Indiana, USA.

A Dissertation
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy
at the University of Cyprus



Recommended for Acceptance by the
Department of Electrical and Computer Engineering
University of Cyprus

March, 2009

ABSTRACT

of the dissertation of
Michalis P. Michaelides,
for the Doctor of Philosophy degree in Electrical Engineering
Title: Distributed Event Detection and Localization in Wireless Sensor Networks
Supervisor: Dr. Christos G. Panayiotou

This dissertation focuses on distributed event detection and localization in Wireless Sensor Networks (WSNs). Sensor nodes are usually small, simple and cheap devices, each equipped with a processor, radio transceiver and sensing probe, and powered by a battery. A WSN consists of a large number of such nodes that form an ad-hoc network in order to deliver the sensed data to the user. One of the common applications envisioned for WSNs is that of monitoring a large region for the presence of an event source that releases a certain signal or substance in the environment. The main objective of this dissertation is to detect and localize the event from the spatially distributed information provided by the sensor nodes in a simple, localized and fault tolerant manner. For the problem of distributed detection this dissertation proposes and analyzes two novel detection algorithms: the Covariance Detector (CD) and the Enhanced Covariance Detector (ECD) that exploit the spatial correlation between the measurements of sensor nodes in close proximity in order to improve the overall coverage of the sensor network. For the problem of distributed localization this dissertation proposes and analyzes two new algorithms: the SNAP (Subtract on Negative Add on Positive) algorithm and the Fault Tolerant Maximum Likelihood (FTML) algorithm. Both algorithms feature accuracy and robustness with respect to faults in the sensor network. In addition, SNAP has desirable properties such as low computational complexity and distributed implementation capability.

ΠΕΡΙΛΗΨΗ

Αυτή η διατριβή επικεντρώνεται στην κατανεμημένη ανίχνευση και εντοπισμό της εστίας ενός συμβάντος σε Ασύρματα Δίκτυα Αισθητήρων (ΑΔΑ). Οι κόμβοι αισθητήρες είναι συνήθως μικρές σε μέγεθος, απλές και φτηνές συσκευές, η κάθε μια εφοδιασμένη με επεξεργαστή, πομποδέκτη ραδιοκυμάτων, αισθητήρες, και αντλεί ενέργεια από μια μπαταρία. Ένα ΑΔΑ αποτελείται από ένα μεγάλο αριθμό τέτοιων κόμβων που σχηματίζουν ένα ad-hoc δίκτυο για την μετάδοση των μετρήσεων των αισθητήρων στο χρήστη. Μια από τις πιο διαδεδομένες εφαρμογές για ΑΔΑ είναι η παρακολούθηση μιας μεγάλης περιοχής για την ανίχνευση της εστίας ενός συμβάντος που απελευθερώνει κάποιας μορφής σήματος ή ουσίας στο περιβάλλον. Ο κύριος στόχος αυτής της διατριβής είναι να ανιχνεύσει και να εντοπίσει την εστία από τις κατανεμημένες στο χώρο πληροφορίες που συλλέγονται από τους κόμβους αισθητήρες με ένα τρόπο απλό, τοπικό και ανεκτικό σε σφάλματα. Για το πρόβλημα της κατανεμημένης ανίχνευσης αυτή η διατριβή προτείνει και αναλύει δύο καινούργιους αλγόριθμους: τον Covariance Detector (CD) και τον Enhanced Covariance Detector (ECD) οι οποίοι εκμεταλλεύονται τη συσχέτιση στο χώρο μεταξύ των μετρήσεων γειτονικών κόμβων αισθητήρων για να βελτιώσουν την συνολική κάλυψη του χώρου από το δίκτυο. Για το πρόβλημα του κατανεμημένου εντοπισμού αυτή η διατριβή προτείνει και αναλύει δύο καινούργιους αλγόριθμους: τον SNAP (Subtract on Negative Add on Positive) και τον Fault Tolerant Maximum Likelihood (FTML). Και οι δύο αλγόριθμοι παρουσιάζουν ακρίβεια και ευρωστία σε σφάλματα μέσα στο δίκτυο αισθητήρων. Επιπρόσθετα, ο SNAP παρουσιάζει επιθυμητά χαρακτηριστικά όπως χαμηλή υπολογιστική πολυπλοκότητα και δυνατότητα κατανεμημένης εφαρμογής.

ACKNOWLEDGMENTS

The road to academic excellence is not an easy road paved with rose petals, it is more of a roller coaster ride with ups and downs, unexpected turns and obstacles, that hopefully in the end reaches a higher level of learning, self-awareness and personal fulfilment. This dissertation is the result of such a five year journey and this section provides me with an opportunity to acknowledge all the people that have made it possible.

First and foremost, I would like to thank my academic advisor Dr Christos Panayiotou, for his guidance and support throughout these years, for always believing in me even at times when I found it hard to believe in myself, for influencing my decision (and making it possible) to pursue my dreams in academia, for being a good friend!

Furthermore, I would like to extend my gratitude to the members of my PhD committee, Professor Marios Polycarpou, Professor Christoforos Hadjicostis, Professor Andreas Pitsillides and Professor Cesare Alippi (Politechnico di Milano, Italy) for their time and effort and their valuable comments for improving this dissertation. Also, I would like to thank all the faculty, administrative staff, fellow colleagues and friends at the University of Cyprus.

Special thanks go to my family for their unconditional love, for showing patience during those days that research seemed like my only priority, for keeping me focused on my goals and what is truly important in life.

Michalis Michaelides

I gratefully acknowledge the grants that supported my research at UCY. This work is partly supported by the Cyprus Research Promotion Foundation under contracts ΠΛΗΡΟ/1104/10 and ΕΡΥΔΙ/0105/01.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Wireless Sensor Networks (WSNs) are a fairly new technology that can potentially provide an interface between the physical world and computers allowing the later to vanish into the background [1]. Recent advances in wireless communications and electronics have enabled the development of low-cost, low-power, multi-functional sensor nodes that are small in size and communicate untethered in short distances. These tiny sensor nodes which consist of sensing, data processing, and communicating components, leverage the idea of sensor networks. A sensor network is often composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it. They have a wide variety of applications including military sensing, infrastructure security, environment and habitat monitoring, industrial sensing, building and structure monitoring, and traffic control. This new promising technology however, also comes with unique challenges because of the vast number of sensor nodes, the limitations in terms of energy and bandwidth and the harsh conditions of operation. There are open research issues involving all layers of the protocol stack. Each layer needs to be modified or even rebuilt to combine power and routing awareness, integrate data with networking protocols, communicate power efficiently through the wireless medium and promote cooperative efforts of sensor nodes [2].

One of the common applications envisioned for WSNs is that of monitoring a large region for the presence of an event. The proposed sensor network can deal with a number of environmental monitoring and tracking applications including acoustic source localization, toxic source identification, and early detection of fire [2, 3, 4]. In this context, the sensor nodes are expected to infer useful information about the event under investigation like the position of the event source and the magnitude of the signal at the source location.

A sensor node is assumed to be a simple, cheap device with limited capabilities in terms of processing and communication. Moreover, it has limited energy resources and is expected to fail often or exhibit unpredictable behavior. For a sensor node alone it would be incon-

ceivable to perform the complex tasks of detection and estimation. The power of the sensor nodes lies in the collective efforts of the group. When a large number of these simple nodes collaborate together, they can achieve complex tasks that seemed impossible at first. Consider what happens when a sensor node becomes alarmed by measuring the signal intensity at its location and comparing it to a threshold. With its limited view, the sensor node cannot infer with certainty whether a target is really present, or this was just a false alarm due to noise. Moreover, it can say very little about the target position; at best, if the target signal propagation characteristics are completely known, it can place the target on a ring around its position. Without any information exchange with the other nodes, all it can do is report this information to the user. However, considering the energy cost of communication, this information would have very low utility.

Consider now the same scenario but with data fusion. After a sensor node becomes alarmed, it first contacts its neighbors so they can collectively detect and localize the event, and *then* report this information to the user. This approach is preferable in the context of sensor networks because it avoids redundant, and possibly false information, flowing end-to-end in the network, consuming valuable bandwidth and energy. There are a lot of questions, however, that have to be answered before we can implement this approach. How many neighbors does the alarmed sensor node need to contact? What type of information does it have to exchange? Do they have to be synchronized? How does it combine the information received from its neighbors with its own information in order to collectively decide detection? What algorithm does it implement based on the exchanged information in order to estimate the event location? These are just some of the important questions that this dissertation seeks to find answers to. In the same context, CSIP (Collaborative Signal Information Processing) [5] is a new research area that aims to determine the parameters of this energy constrained sensor collaboration i.e. who should sense, what needs to be sensed, whom the information should be passed on to.

This dissertation concentrates on developing distributed algorithms for detection and localization of an event in a WSN with the following main characteristics:

1. Simple - Due to the limited capabilities of the sensor nodes they cannot perform any complex computations. The algorithm should only perform simple operations.

2. Localized - Communication is considered to be the most expensive operation in terms of energy so the algorithm should only allow local information exchanges between the sensor nodes.

3. Fault tolerant - Sensor nodes can fail and exhibit unpredictable behavior at any time. The algorithm should retain its performance when a number of sensor nodes report

erroneous observations.

The performance of the detection algorithms is measured with respect to coverage, defined as the area where an event is detected with high probability (at least 0.5) subject to a fixed false alarm probability. The performance of the localization algorithms is measured in terms of the mean positioning error (in general, unless otherwise specified, the distance units are always assumed to be meters). Energy considerations and system lifetime are also of primal importance for the development of the algorithms.

1.1 Summary

The work done for this dissertation can be divided in two phases outlined below:

Phase I (Detection): For the problem of distributed detection we first use the Mean Detector (MD), the most common local detection scheme proposed in literature for WSNs. In this context, each sensor node decides its alarm status alone by comparing its calculated mean to a predefined threshold. To find this threshold, in Chapter 4, we solve an optimization problem that minimizes the error in detection (false alarms and misses) in a sensor network. When using the MD, the general intuition becomes to place sensor nodes apart to improve coverage. Consider though what happens if we have two sensor nodes that happen to fall next to each other. In the presence of an event, their measurements should look quite similar. This similarity, or spatial correlation between sensor nodes in proximity, can provide a powerful complementary detection scheme for WSNs to improve the overall coverage of the sensor network. We start by examining collaborative pairwise detection schemes for the case where we have just two sensor nodes in Chapter 5. We show that in order to increase their collective coverage area, the detection strategy becomes a function of the distance between the two sensors. For closely spaced sensor nodes, we propose the use of a Covariance Detector (CD) at the local detection level that exploits this similarity by calculating the sample covariance between them. Furthermore, we propose the Enhanced Covariance Detector (ECD) that combines the energy and the covariance information from two sensor nodes utilizing two different thresholds (one for the energy test statistic and another for the covariance). Next, we extend these results to the general case where we have a field with N randomly deployed sensor nodes in Chapter 6. The overall detection is decided at the base station using a fusion rule of the general form "event is detected if at least K out of N detection reports from alarmed sensors are received." We propose a Hybrid Detector (HD) where each sensor node

independently chooses between the MD and the CD based on the distance from its closest neighbor. This detector combines the benefits of the other two detectors: it uses the CD in situations where sensor nodes are close to each other to take advantage of possible spatial correlation between their measurements; in other situations where sensor nodes are far away from their neighbors they rely on their own measurements for detection and use the MD. Depending on the event distribution, the sensor node can decide which detector to use by solving an optimization problem involving the coverage of each detector. In many situations of interest though, the event characteristics are completely unknown. For those cases, we show how we can still evaluate a distance threshold from the "expected distance" between a sensor node and its closest neighbor. Finally, we derive analytical expressions for the system probability of false alarm for all proposed detectors and compare their performance in terms of coverage.

Phase II (Localization):

For the problem of distributed localization we first employ nonlinear least squares optimization techniques in Chapter 7 to estimate the event location. As already mentioned, a WSN consists of low-cost devices which have limited resources (processing capabilities, memory, and power), calibration mismatches (varying sensor sensitivities) and may fail frequently. With this in mind, we departed from least squares with the objective to develop a simple, efficient, fault tolerant algorithm that can quickly identify the event location using only binary data from the sensor nodes. SNAP (Subtract on Negative Add on Positive), is the algorithm developed in Chapter 8 and one of the main contributions of this dissertation. The main idea behind SNAP is to use the observations of *all* sensors to efficiently construct a likelihood matrix by summing contributions of ± 1. By bounding the contribution of each sensor, we do not allow any sensor measurement to dominate the overall estimation result which constitutes the basic reason for the algorithm's fault tolerant behavior. Furthermore, we show how the SNAP algorithm can be implemented in a distributed fashion (dSNAP). In this context, any alarmed node in the vicinity of the event can be elected as leader and perform the localization of the event by utilizing data from only the sensor nodes inside its neighborhood. The SNAP algorithm can be actually cast in a maximum likelihood (ML) estimator with respect to the binary data. Compared to ML, however, SNAP requires only simple operations and can retain its accuracy when a large percentage of the sensor nodes report erroneous observations. Next, in Chapter 9, we show how ML can be modified in order to accommodate erroneous observations coming from faulty sensor nodes. Specifically, by incorporating the fault probability when calculating the likelihood function we develop a fault tolerant maximum likelihood (FTML) estimator appropriate for sensor networks applications. Compared to the SNAP algorithm in the presence of faults, the two can achieve similar performance; FTML is slightly more accurate while SNAP is computationally less demanding and re-

quires fewer parameters. In Chapter 10, we study two different applications for the SNAP algorithm. First, we modify the SNAP algorithm to incorporate covariance information for performing the localization. Then, we apply the SNAP algorithm to the inverse localization problem of a plume source.

1.2 Main Contributions

The main contributions of this dissertation are summarized below:

Detection: For the problem of distributed detection this dissertation proposes and analyzes two novel detection algorithms: **the Covariance Detector (CD)** and the **Enhanced Covariance Detector (ECD)** that exploit the spatial correlation between the measurements of sensor nodes in close proximity in order to improve the overall coverage of the sensor network.

Estimation: For the problem of distributed localization this dissertation proposes and analyzes two new algorithms: the **SNAP (Subtract on Negative Add on Positive)** algorithm and the **Fault Tolerant Maximum Likelihood (FTML)** algorithm. Both algorithms feature accuracy and robustness with respect to faults in the sensor network. In addition, SNAP has desirable properties such as low computational complexity and distributed implementation capability.

1.3 Organization of the Dissertation

The first two chapters provide more details for the problem under investigation. Chapter 2 presents an extensive literature review of what is considered to be state of the art in event detection and localization using WSNs. Chapter 3 provides the assumptions and explains the different models used throughout this dissertation.

The next three chapters deal with distributed detection. In Chapter 4, we solve an optimization problem for the Mean Detector to obtain the optimal detection threshold at the sensor level. Chapter 5 examines collaborative pairwise detection schemes for the two sensor case and proposes the Enhanced Covariance Detector (ECD). In Chapter 6, we extend these re-

sults to the general case sensor network and propose the Hybrid Detector (HD).

The following four chapters deal with distributed localization. In Chapter 7, we investigate the use of nonlinear Least Squares techniques for estimating the event location. In Chapter 8, we introduce the SNAP algorithm and show how it can be implemented in a distributed fashion. Chapter 9 shows that SNAP can achieve fault tolerant localization and proposes the Fault Tolerant Maximum Likelihood (FTML) estimator. In Chapter 10, we further investigate applying the SNAP algorithm using covariance information and for the localization of plume sources.

Finally, this dissertation concludes with Chapter 11, where we also present our plans for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

This chapter presents an extensive literature review of what is considered to be state of the art in the problem of distributed event detection and localization in Wireless Sensor Networks. Special attention is given to the problems of coverage, spatial correlation, binary estimators, and fault tolerance since they are directly related to the issues addressed by this thesis. Finally, representative work is presented in the areas of target tracking and plume tracking because both of these areas are tightly coupled with the distributed localization algorithms developed as part of this dissertation.

2.2 Distributed Detection

Distributed detection using multiple sensors has been extensively investigated for radar and sonar applications (see [6, 7] and references therein). Interest in decentralized detection and estimation has re-surfaced with anticipated applications using multiple sensors which may be geographically dispersed. In classical multi-sensor detection and estimation, it is assumed that all the local sensors (such as radar, sonar, infrared) communicate all their data to a central processor that performs optimal detection and tracking of targets based on conventional statistical techniques. In decentralized processing, some preliminary processing of data (often lossy compression) is carried out at each sensor and condensed information is sent from each sensor to other sensors and ultimately the central processor which is often known as

the *fusion center*. In the terminology of sensor networks we say that the network has intelligence at each node. Unlike the central processor in centralized systems, the fusion center of a decentralized system has only partial information as communicated by the sensors. This results in a loss of performance in decentralized systems as compared to centralized systems. However, the performance loss can be made small by optimally processing the information at the sensors. The objective in most studies is to develop computationally efficient algorithms at the sensors and at the fusion center.

Optimality is usually studied under the Neyman-Pearson and Bayesian detection criteria [8, 9]. In general, the Neyman-Pearson formulation finds (optimum) local and global decision rules that minimize the global probability of miss (P_M) for a prescribed bound on the global probability of false alarm (P_F). On the other hand, the Bayesian formulation assigns costs to each error (miss and false alarm) and tries to minimize the overall probability of error. Both of these formulations however, require complete or partial knowledge of the joint densities (pdf) of the observations at the sensor nodes given the hypothesis H_l for $l = 0, 1$[1]. For conditionally independent observations, optimum fusion rules have been derived in [10, 11]. *Conditional independence* of sensor observations implies that the joint density of the observations obeys

$$p(y_1, ..., y_N|H_l) = \prod_{i=1}^{N} p(y_i|H_l), \text{ for } l = 0, 1. \tag{2.1}$$

For conditionally independent observations, the optimal tests at the sensors as well as the decision rules at the fusion center are threshold rules based on the appropriate likelihood ratios.

In large-scale wireless sensor networks, however, the signal generated by the event to be detected has unknown strength and varies spatially making sensor observations location-dependent and not identically distributed. The observations at the sensors are *conditionally dependent* when the joint density of the observations given the hypothesis, cannot be written as the product of the marginal densities as in (2.1). Such situations would arise if one detects a random signal in noise or if the sensor noise samples are correlated when detecting a deterministic signal in noise. It is shown in [12] that for the conditionally dependent case, the optimal tests at the sensors are no longer of the threshold type based solely on the likelihood ratio of the observations at the individual sensors. In general, without the conditional independence assumption, the optimal solution becomes mathematically intractable because it would require an exhaustive search of all possible decision rules. For example, even for

[1]In statistical hypothesis testing, H_1 refers to the signal present hypothesis while H_0 refers to the signal absent (noise only) hypothesis.

the simple case of 2 binary sensors with finite sets of observations Y_i for $i = 1, 2$, a solution by exhaustive enumeration could require the examination of us many as $2^{N_1+N_2}$ pairs of decision rules (where N_i denotes the cardinality of Y_i for $i = 1, 2$). It is shown in [13] that when the observations are discrete and conditionally dependent, the optimal solution is non-polynomial complete (NP-complete). When the joint distributions of the observations at the sensor have a certain structure, the performance of certain distributed decision rules can be determined. The authors in [14] design the optimum combining scheme at the fusion center when the local decision rules and the correlations between the local decisions are given. In [15], the optimal data fusion rule is developed for correlated local binary decisions in terms of the conditional correlation of all orders. In [16], the authors study Bayes-optimal binary quantization for the detection of a shift in mean in a pair of dependent Gaussian random variables.

In most previous work using correlated observations, the local sensor performances (in terms of the probability of detection and the probability of false alarm) and the correlation between their local decisions are assumed given. For the WSN under investigation, however, both the local sensor performance and the correlation between their measurements are unknown and can change dynamically with the location of the event, making it infeasible in most cases to obtain this information at the fusion center. Consequently, one needs to resort to suboptimal schemes and heuristics to achieve the desired objectives and the optimal decision rule for detection should be determined at the sensor node level sometimes even before deployment [17]. Such heuristics, are the detectors presented and analyzed in this dissertation. The Covariance Detector (CD) and the Enhanced Covariance Detector (ECD) aim to exploit the spatial correlation between the measurements of sensor nodes in close proximity in order to improve the overall coverage of the sensor network.

2.3 Coverage

One of the fundamental issues in sensor networks is the coverage problem, which reflects how well an area is monitored by sensors. In general, coverage can be considered as the measure of quality of service of a sensor network [18]. For example, consider a sensor network deployed in a forest for providing early warnings against the presence of a fire. In this context, one may ask how well the network can observe a given area and what the chances are that a fire starting in a specific location will be detected in a given time frame. Furthermore, coverage formulations can try to find weak points in a sensor field and suggest

future deployment or reconfiguration schemes for improving the overall quality of service.

Coverage is perhaps more well known in the literature from a computational geometry standpoint. A famous example is the *Art Gallery Problem* [19] that deals with determining the number of observers necessary to cover an art gallery room such that every point is seen by at least one observer. This problem has found many applications in many domains such as the optimal antenna placement problems for wireless communication. The Art Gallery Problem has a linear time solution in 2D and was shown to be NP-hard in the 3D case. A polynomial time approximation solution for the 3D version of the Art Gallery Problem is presented in [19]. Coverage has been extensively studied for sensor networks in the last few years using mostly computational geometry techniques for developing algorithms for worst / best case coverage [18]. In worst / best case coverage, attempts are made to quantify the quality of service by finding areas of lower / higher observability from sensor nodes and detecting breach / support regions. The key idea is to combine computational geometry, specifically the Voronoi diagram and the Delaunay triangulation, together with graph search algorithms for establishing optimal polynomial time worst and average case algorithm for coverage calculation. In 2D, the Voronoi diagram of a set of discrete sites partitions the plane into a set of convex polygons such that all points inside the polygon are closest to only one site. This construction effectively produces polygons with edges that are equidistant from neighboring sites. The Delaunay triangulation can be obtained by connecting the sites in the Voronoi diagram whose polygons share a common edge. It has been shown that among all possible triangulations, the Delaunay triangulation maximizes the smallest angle in each triangle. In fact, since sites that are close together are connected, the Delaunay triangulation can be used to find the two closest sites by considering the shortest edge in the triangulation. The problem of coverage has also been formulated in various other ways. For example, in [20], the authors deal with exposure, which is directly related to coverage in that it is a measure of how well an object, moving on an arbitrary path, can be observed by the sensor network over a period of time. An algorithm is developed for determining a path with the least exposure. In [21], the authors present polynomial time algorithms for determining whether an area is sufficiently k-covered, in the sense that every point in the monitored area is covered by at least k sensors. In addition to developing algorithms for calculating coverage, scheduling schemes have also been investigated in the literature for turning off some redundant nodes while still preserving a complete coverage of the monitored area [22].

Most of the aforementioned work assumes that the sensing coverage of a sensor node can be represented by a disc inside which an event is always detected. In other words, an event that occurs within the sensing radius of a node is always assumed detected with probability 1 while any event outside the sensing disk is assumed not detected. This idealized model might be convenient for developing algorithms that calculate coverage in polynomial time

but it fails to capture several important characteristics of sensor nodes in the real world. In fact, sensor node measurements are expected to be noisy so we need to probabilistically account for the noise in the sensor readings. This means that there is always a nonzero probability of incorrectly detecting a target when there is actually no target in the field (i.e., false alarm) and therefore a tradeoff we need to consider between the probability of false alarm and the probability of detection. For the purposes of this dissertation, we directly associate coverage to the probability of detection. There have been a few other attempts in the literature to deal with coverage in a probabilistic way. In [23], the authors propose a probabilistic coverage algorithm based on a computational geometry approach that evaluates the degree of confidence in coverage provided by a randomly deployed sensor network. The vulnerability of sensor networks to unauthorized traversal and monitoring is analyzed in [24] by considering the tradeoff between a target passing undetected through the field and the false alarm probability. In [25], the authors address the problem of finding the critical density of sensors for complete coverage. Assuming the presence of a threshold for dealing with noise, they concentrate on finding the number of sensors for detecting a moving target.

Coverage area, however, is not only influenced by noise. Note that even when we probabilistically account for the presence of noise, the coverage area of a sensor node remains a disc around the node inside which the target will be detected above a certain probability. A main contribution of this dissertation is to show that the coverage area of a sensor node also depends on the detection strategy (fusion rule). For example, we will be showing that a Covariance Detector for a pair of sensor nodes produces an ellipse-shaped coverage area whose size depends on the separating distance between the two nodes. Therefore, in order to maximize the coverage area of a sensor network, it becomes imperative to investigate collaborative detection schemes between the sensor nodes. This is further investigated in Chapters 5.

2.4 Spatial Correlation

The physical phenomena monitored by sensor networks, e.g. forest temperature, water contamination, usually yield sensed data that are strongly correlated in space [26]. With this in mind, researchers have designed a large number of sensor network protocols and algorithms that attempt to exploit such correlations. Spatial correlation has been used in data aggregation and routing algorithms, data storage and querying, MAC protocol design, localization, data compression and encoding, and calibration. In [26], the authors present a simple and

accurate model of spatially correlated sensor network data. The model can capture correlation in data irrespective of the node density, the number of source nodes or the topology. In [27] the authors develop a theoretical framework to model the spatial and temporal correlations in a WSN and use it for designing efficient communication protocols. The authors of [28] develop a decision fusion Bayesian framework for detecting and correcting sensor measurement faults in the event region by exploiting the fact that measurement errors are uncorrelated while environmental conditions are spatially correlated.

Typical WSN applications require spatially dense sensor deployment in order to achieve satisfactory coverage. As a result, multiple sensors record information about a single event in the sensor field. Due to high density in the network topology, spatially proximal sensor observations are highly correlated with the degree of correlation increasing with decreasing internode separation. For the purpose of this dissertation, we model the spatial correlation in the actual measurements that the sensor nodes get based on the distance from the event source and the distance from each other. In other words, measurements of sensors that are close to each other and close to the source are correlated. On the other hand, sensor nodes that are located far away from the source do not receive any signal information; so even if they happen to fall next to each other their signal measurements become uncorrelated. For the development of efficient communication protocols, intuitively, due to the spatial correlation, data from spatially separated sensors is more useful to the sink than highly correlated data from nodes in proximity [27]. For distributed event detection, however, counter-intuitively, spatially correlated sensor data from nodes in proximity can become quite useful in increasing the coverage of the sensor network through a covariance detector scheme, as the one proposed in Chapter 5.

2.5 Distributed Localization

Localization of the event position has been extensively studied in the last 20 years using arrays of sensors for radar, sonar and acoustic target tracking applications (see [29, 30, 31] and references therein) and in the context of wireless cellular networks to support location-based services (see [32]). A variety of techniques have been proposed to solve the localization problem that can be classified into 3 main categories: 1. DOA (Direction of Arrival) 2. TDOA (Time Difference of Arrival) 3. Energy-based. DOA can be estimated by exploiting the phase difference measured at the receiving sensors and is applicable in the case of a coherent, narrow band source. In practice, DOA measurements typically require a costly

antenna array on each node. TDOA is more suitable for broadband sources but typically require accurate measurement or estimation of time delay. In contrast, energy-based methods are based on received sensor signal strength which is much easier, requires simpler hardware, and is less costly to obtain from the time series recordings from each sensor. Furthermore, they do not require any synchronization between the sensor nodes. On the other hand, they do require the communication of the energy readings from the sensor nodes to a central processing unit with enough processing power to solve this non-linear optimization problem using iterative search techniques. For an estimation algorithm to be implementable in the context of WSNs, it has to be simple enough so it can be performed by any sensor node in a distributed fashion, energy-efficient and fault-tolerant. Classical estimation techniques like least squares and maximum likelihood usually give accurate results but at the expense of heavy message exchanges and computations. Therefore, it is not clear how any of these methods could be implemented in a distributed fashion by a sensor node. Moreover, the fault tolerance of these schemes has not been demonstrated.

2.6 Binary Estimators

More recently, the use of binary data for localization has been proposed in the literature. Sensor nodes are expected to be low-cost, simple devices with limited resources (processing capabilities, memory, and power). Binary decisions are simple problems a sensor node can solve by just comparing its measurements to a predefined threshold. Binary decisions are also less sensitive to calibration mismatches and varying sensor sensitivities. Moreover, using binary observations limits the bandwidth usage and conserves energy; only single-bit information needs to be transmitted from the sensor nodes to the fusion center (sink) or, in the case of decentralized implementation, to the leader node that will perform the localization.

For event localization in WSNs using binary data two different methods have been proposed: The Centroid Estimator (CE) [33] and the Maximum Likelihood (ML) [34] estimator. Since both of these estimators are used for the performance evaluation of the algorithms developed in this dissertation, we provide their details below.

2.6.1 Centroid Estimator (CE)

The centroid of a finite set of points can be computed as the arithmetic mean of each coordinate of the points. Let (x_n, y_n), $n = 1, \cdots, P$ ($P \leq N$) denote the positions of all alarmed sensor nodes. Then, the event location estimated by CE is the centroid of these positions:

$$\hat{\theta}_{CE} = [\hat{x}_s, \hat{y}_s] = \left[\frac{1}{P} \sum_{n=1}^{P} x_n, \frac{1}{P} \sum_{n=1}^{P} y_n \right].$$

This simple technique works well under conditions of dense uniform sensor deployments and when the events are not located close to the field boundaries.

2.6.2 Maximum Likelihood (ML)

Maximum Likelihood is overwhelmingly the most popular approach to obtaining practical estimators. It has the distinct advantage of being a "turn-the-crank" procedure, allowing it to be implemented for complicated estimation problems. Additionally, in most cases of practical interest its performance is optimal for large enough data records. Specifically, it is approximately a minimum variance unbiased estimator due to its approximate efficiency [35].

For the problem under investigation, the Maximum Likelihood estimator finds the most likely source location based on the binary data received from the sensor nodes. It essentially evaluates a likelihood function for every possible source location in the field and then the maximum of this likelihood function points to the estimated source location. First, let us formulate the problem by defining the indicator function for sensor node $n = 1, \cdots, N$ at time $t = 1, \cdots, M$:

$$I_{n,t} = \begin{cases} 0, & \text{if } z_{n,t} < T \\ 1, & \text{if } z_{n,t} \geq T \end{cases} \tag{2.2}$$

where $z_{n,t}$ is the measurement of sensor node n at time t and T is a prescribed threshold. Thus, the sensor data can be represented as $\mathbf{I} = \{I_{n,t} : n = 1, \cdots, N, t = 1, \cdots, M\}$. The goal is to estimate the source location $\theta = [x_s, y_s]$ using the collected data $\mathbf{I}$.

The Maximum Likelihood Estimator has the form

$$\hat{\theta}_{ML} = \max_{\theta} \log p(\mathbf{I} \mid \theta) \tag{2.3}$$

where the log-likelihood function, as derived in [34], is given by

$$\log p(\mathbf{I} \mid \theta) = \sum_{n=1}^{N} \sum_{t=1}^{M} I_{n,t} \times \log \left[Q\left(\frac{T - s_n(\theta)}{\sigma_w} \right) \right] + (1 - I_{n,t}) \times \log \left[1 - Q\left(\frac{T - s_n(\theta)}{\sigma_w} \right) \right]. \quad (2.4)$$

In this expression, $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt$ is the complementary distribution function of the standard Gaussian distribution. Also, $s_n(\theta)$ is the signal that would have been measured by sensor n if the source was at location θ and there was no noise. The ML estimator is shown to be optimal (achieves the Cramer-Rao bound) in [34] when enough sensor measurements are used.

For the purposes of this dissertation we implement a discrete version of the algorithm described above. Specifically, we divide the area in $G \times G$ equal size grid-cells and evaluate the log-likelihood function G^2 times using (2.4) assuming the source is located at the center of each cell. The maximum of the resulting matrix points to the event location. For the rest of this dissertation we will refer to this discrete version of the algorithm as simply ML (Maximum Likelihood).

SNAP (Subtract on Negative Add on Positive), one of the main contributions of this dissertation, can be actually cast in a ML estimator with respect to the binary data. Compared to ML, the SNAP algorithm can achieve similar accuracy using less computations and making fewer assumptions. For more information see Chapter 8.

2.7 Fault Tolerance

Sensor nodes are often envisioned to operate unattended for long periods of time in hostile environments, e.g. airdropped inside an area for tracking potential targets and intruders. In terms of security, this means that there is an increased probability that the sensor readings are altered by a malicious attacker. For example, an intruder that would like to move inside the area without being tracked, may try to "fool" the system by compromising some of the sensors and modifying their readings to divert the attention of the monitoring personnel. A possible attacker strategy would be to tamper with sensors away from the attacker's true position to become falsely alarmed and prevent sensors close to the attacker from reporting detection. For instance in [36], the authors present a scenario with malicious actuator nodes

deployed to perturb or distort the readings of neighboring sensor nodes within their actuation radius. Another possible attacker strategy would be to deploy various decoys inside the monitored area to cause confusion through a series of false alarms. In applications such as environmental monitoring of large areas, false alarms may incur a significant cost because a response crew may have to travel to the suspected area. Also, in many cases, frequent false alarms may make the users simply ignore *all* alarms and as a result even important detected events may go unnoticed. For example, in 1980 the rebels in Afghanistan threw rabbits over base fences that caused the motion detectors to generate a series of false alarms [37]. As a result the base turned off completely the monitoring system allowing surreptitious attacks! In addition to the case of malicious intervene, sensor nodes can provide erroneous observations for a variety of other reasons: noise, energy depletion, environmental harsh conditions of operation and software problems. For example, in [38] the authors report real experiments where a node would constantly report false detections when its program board was overheated, or scenarios in which nodes were programmed in an incorrect manner that yielded unpredictable (Byzantine type) behavior. Another source of faults is the network unreliability that can result in a high percentage of dropped packets. These can be attributed to the uncertain nature of the communication medium (mainly due to radio frequency interference and obstacle presence), as well as collisions due to the dense deployment of the sensor networks.

Fault tolerant event detection in WSNs has received considerable attention over the last few years. Specifically, the problem of fault tolerant event region detection using binary sensors was first considered in [28]. Event region refers to a region of the environment with a distinguishable characteristic, for example a region having a chemical concentration of some chemical agent that exceeds a certain threshold. The proposed solution, in the form of Bayesian fault recognition algorithms, exploits the notion that the measurement errors due to the faulty equipment are likely to be uncorrelated, while environmental conditions are spatially correlated. The work in [28] has been followed by three other publications dealing with the same problem of fault tolerant event region detection. In [39], the authors provide comments on the original paper and correct some of the mistakes in the theoretical analysis section. In [40], the authors extend the model to account for the fact that the sensor errors can have two different sources; an error could be noise-related or coming from a sensor fault. The model is further extended in [41] by considering the case where nodes can have different failure probability levels. The authors also propose two error models that are particularly suitable for applications where the event is highly localized. The first model takes into account the fact that nodes that are closer to each other have a higher spatial correlation than nodes that are further apart, while the second model accounts for the relative geographical distributions of the two voting quorums (the two subsets of neighbors determining the

presence or absence of the event).

Fault tolerant event localization in WSNs has received limited attention so far from the research community. In [33], the authors propose a median detector to filter out extreme measurements followed by a Centroid Estimator (defined in Section 2.6) to achieve fault-tolerant localization. The binary estimators proposed in the literature, however, can yield significant estimation errors when the sensor nodes start failing; the CE is sensitive to false positives, i.e., sensor nodes far away from the source becoming falsely alarmed. On the other hand, the Maximum Likelihood (defined in Section 2.6) is extremely sensitive to false negatives, i.e., when a node located close to the event does not become alarmed. The SNAP algorithm is one of the first attempts to provide a simple, fault tolerant localization algorithm suitable for WSNs. It is essentially a maximum likelihood estimator that can achieve similar accuracy in a much simpler way. Furthermore, in Chapter 9, by incorporating the fault probability when calculating the likelihood function we develop FTML, a fault tolerant maximum likelihood estimator also appropriate for sensor networks applications. Compared to FTML, the SNAP algorithm can achieve similar accuracy and fault tolerance but it is computationally more efficient and makes fewer parameter assumptions.

2.8 Target Tracking

Mobile object tracking is one of the most fundamental collaborative information processing problems in WSNs. In this section we also review representative work on target tracking because it is tightly coupled with the distributed localization algorithms developed in this dissertation. Tracking techniques, combine the localization techniques with a target mobility model and assume a probability distribution for the sensor measurement errors to achieve superior performance. They rely on Bayesian filtering variants [42], such as Kalman Filter or Particle Filter, to mitigate the effect of measurement noise and alleviate high positioning errors that do not reflect the target's mobility pattern.

Most of the work on target tracking uses a decentralized approach. When a target is detected, a cluster is formed and all the alarmed sensors in the vicinity of the target forward their measurements to the cluster head (leader). The cluster head estimates the target location, based on the queried sensor measurements and predicts the next location according to the filtering scheme. Then it multicasts wake-up information to the predicted (forwarding) area, in order to initiate the next cluster head election and propagate tracking information and filter specific parameters. In [43], the cluster head performs the target position estimation by

triangulating the distances from three closest nodes. Subsequently, a simple filtering scheme that uses only the previous two locations to calculate the target's speed and moving direction, is employed to linearly predict the next location. A combination of a geometric algorithm and Particle Filter is proposed in [44] to track the trajectory, estimate the velocity of the target and yield economical path descriptions. In [45], a dynamic clustering scheme is proposed for accurate tracking, while localization is based on non-linear optimization methods. Authors assume a hierarchical topology and utilize a Voronoi diagram together with back-off timers to elect the cluster-head closest to the target with high probability. In [46], the authors suggest a distributed group management algorithm for electing a single leader and resolving contention via message exchange. They use a time stamp to elect the leader that first witnessed the event. In [47], the sensor nodes are aggregated into collaborative groups for a target counting task. For each distinct target a single leader is elected via one hop information exchange by identifying the sensor node that received the highest signal power. In [48, 49], the authors have addressed the problem of dynamically querying sensors and routing data in the network so information gain is maximized while latency and bandwidth consumption is minimized. They concentrate on selecting the next best sensor for a vehicle tracking application and updating the belief state in order to maximize information content. Authors in [50], propose a distributed approach where each node exchanges messages only with its neighbors in the network and runs the tracking algorithm locally by using available measurements from all neighboring nodes. In this fashion, the tracking information is duplicated and distributed among all nodes in the vicinity of the target. The position estimates are given by the known node positions disturbed by additive noise while the estimated target positions are processed by a Kalman Filter algorithm.

2.9 Plume Tracking

Tracing contaminant transport to source can be used against terrorist attacks to rapidly respond to a chemical, biological or radiological event and provide the proper authorities the necessary information to deal with such events [51, 52]. Such a sensor network can also be used to provide early warning against accidental spillage of toxic waste by ships, factories etc. Other examples include early detection of fires and other environmental monitoring. Underwater environmental monitoring programs are under way both in USA [53, 54, 55] and in Europe [56, 57]. A lot of research has been done in the area of plume tracking using underwater autonomous unmanned vehicles trying to locate a chemical source. They use bio-mimetic robotic plume-tracing algorithms based on olfactory sensing. Usually a single

sensor on the robot is capable of sensing the chemical and sensing or estimating fluid velocity. Subsequently, this information is used to determine the speed and heading direction of the vehicle such that the motion of the vehicle is likely to locate the odor source. Farrell [58] has done significant work in locating a chemical source underwater using an autonomous vehicle operating in a fluid flow.

Locating an event source using autonomous vehicles works fine when we know that an event exists. However, in many instances, for example in environmental monitoring, an event occurs very rarely. In this case, it would be very expensive to have an autonomous vehicle continuously searching until it finds an event. A more preferable alternative is to deploy a stationary, low cost, low power sensor network that would continuously monitor for the existence of the event source. Once the network detects the existence of an event, it generates an initial estimate of the position of the event source. Subsequently, this initial estimate can be used by a group of mobile robots which will then move closer to the source in order to accurately determine its position.

As part of this dissertation, we first investigate using least square methods and then applying the SNAP algorithm for determining an initial estimate of the event source position using binary measurements from stationary sensors. In [59] the author also proposes the use of binary sensors for a plume tracking application. He tracks the correlation between the sensor nodes that have detected the target and constructs a likelihood function whose maximum points to the event location. The main difference between SNAP and this work is that we also consider the sensors that did not detect the event when forming the likelihood function.

CHAPTER 3

EVENT SOURCE MODEL

This chapter presents the details of the event source model used throughout this dissertation. We start by making the following assumptions for the sensor network that detects and estimates the position of an event:

1. A set of N sensor nodes is randomly spread (in a uniform manner) over a square field of area A. The nodes are static. Their position is denoted by (x_n, y_n), $n = 1, \cdots, N$ and it is assumed that it is known (e.g., a small fraction of the sensor nodes uses GPS, while the rest estimate their location using localization algorithms).

2. A single source of the event is located at a position (x_s, y_s) which is also a random location generated by a uniform distribution inside A. The source emits a continuous signal that attenuates inside A.

3. The network is connected in the sense that each node has at least one path to the fusion center (sink). The sensors that "detect" the event (i.e., they are alarmed) send a packet to the sink or the leader node; otherwise they remain silent.

All assumptions are quite common and reasonable for sensor networks. Assumption 1 is primarily needed for the localization algorithms and it is a rather weak assumption. In fact, as we will be further demonstrating in Chapter 8, the SNAP algorithm only requires approximate sensor node positions in order to accurately estimate the event location. Regarding Assumption 2, the results of this thesis can be readily extended to situations where we have multiple sources; this is a subject of on-going research. Finally, Assumption 3 is rather technical; it is only used to ensure that the sensor network is fully connected. We should point out that the localization algorithms developed in this thesis are fault tolerant, so the performance of the WSN does not deteriorate even when a significant number of packets do not arrive at the sink.

In general, the t-th sample measurement of any sensor n located at (x_n, y_n) is composed of a signal component $S_{n,t}$ and a noise component $W_{n,t}$ and is given by

$$Z_{n,t} = \min\left\{V_{max}, \gamma S_{n,t} + W_{n,t}\right\}, \tag{3.1}$$

for $n = 1, \cdots, N$, $t = 1, \cdots, M$. In this model V_{max} and γ are sensor specific parameters: the first reflects the maximum measurement that a sensor can register while the second is a scaling factor corresponding to the sensor gain. In order to facilitate the theoretical analysis of the proposed algorithms and without any loss of generality, for the remaining of this dissertation we assume $\gamma = 1$ and take V_{max} to be the signal amplitude that would be measured one meter away from the source location.

$S_{n,t}$ is the realization of a space-time process $s(t, x, y)$ that describes the signal propagation dynamics. Specifically, we restrict ourselves to the xy-plane and assume a general model of the form

$$s(t, x, y) = f(t, x, y, x_s, y_s), \tag{3.2}$$

where $s(t, x, y)$ is the signal measurement at any location (x, y) and time t while (x_s, y_s) is the source location. The function $f(\cdot)$ describes the signal propagation dynamics and can be chosen according to the application. For this dissertation, we use three different signal propagation models explained in detail in the subsequent sections: 1. A uniform signal propagation model that is appropriate for acoustical or electromagnetic sources where the signal attenuates uniformly around the source location, 2. A stochastic model that applies to environmental monitoring conditions where sensor node measurements are expected to be highly correlated in the space domain and 3. A directed propagation model that is more suitable for smoke or plume sources where the wind can push the substance released from the source in a certain direction.

Finally, unless otherwise specified, $W_{n,t}$ is assumed to be additive white Gaussian noise, i.e. $W_{n,t}$ is a sequence of `iid` (independent and identically distributed) Gaussian random variables $W_{n,t} \sim \mathcal{N}(0, \sigma_W^2)$ for $n = 1, \cdots, N$ and $t = 1, \cdots, M$.

3.1 Basic signal propagation model

First, we present a deterministic model where the source signal attenuates uniformly in all directions. Similar models have been extensively used in the literature (see [30, 31, 33, 34]) for acoustic source localization using sensor networks.

Specifically, we assume that the measured intensity at the source is c and as we move away from the source, the measured intensity is inversely proportional to the distance from the source raised to some power $\alpha \subset \mathbb{R}^+$ which depends on the environment. As a result, the t-th signal measurement of any sensor n located at (x_n, y_n) is given by

$$S_{n,t} = \frac{c}{r_n^{\alpha}}, \tag{3.3}$$

for $i = 1, \cdots, N$, $t = 1, \cdots, M$. In addition, r_n is the radial distance from the source, i.e.,

$$r_n = \sqrt{(x_n - x_s)^2 + (y_n - y_s)^2}. \tag{3.4}$$

For the purposes of this dissertation, unless otherwise specified, we use $\alpha = 2$.

3.2 Stochastic signal model with spatial correlation

This section describes a stochastic signal model that incorporates spatial correlation between the signal measured at the sensor nodes locations. This signal energy model is appropriate for a variety of problems where we use a WSN to monitor the environment, since sensor observations are expected to be highly correlated in the space domain [27].

The signal is generated at the source location according to a normal distribution $\mathcal{N}(c, \sigma_S^2)$. The signal attenuates uniformly in all directions from the source and is modeled by a Gaussian space-time-varying process $\mathbf{s}(x, y, t)$. $S_{n,t}$ becomes the realization of a space-time-varying process $\mathbf{s}(x, y, t)$, i.e., it is the sample at sensor n located at a position (x_n, y_n) at time t. Since only spatial correlation is considered, the samples received at each sensor node location are temporally independent (since they are only influenced by Gaussian white noise). Specifically, we assume that

$$\mathbb{E}\left[S_{n,t}\right] = \frac{c}{r_n^2} \tag{3.5}$$

$$\mathbb{E}\left[S_{n,t}^2\right] = \sigma_S^2 C^2(\lambda_v, r_n) \tag{3.6}$$

$$\mathbb{E}\left[S_{n,t} S_{m,t}\right] = \sigma_S^2 C(\lambda_v, r_n) C(\lambda_v, r_m) C(\lambda_c, d_{nm}) \tag{3.7}$$

for $n, m = 1, \cdots, N$ and $t = 1, \cdots, M$. In the above equations, d_{nm} is the Euclidean distance between the sensor nodes n and m and r_n is the distance of node n from the source, i.e., $r_n = \sqrt{(x_n - x_s)^2 + (y_n - y_s)^2}$. Furthermore, $|C(\lambda, r)|$ is a decreasing function of the distance r such that $\lim_{r \to \infty} C(\lambda, r) = 0$. For the purposes of this thesis we assume that

$$C(\lambda, r) = e^{-\frac{r}{\lambda}} \tag{3.8}$$

$\lambda > 0$, however we point out that other functions are also possible, e.g., see [27]. The constants λ_v and λ_c in (3.6), (3.7) can be chosen according to the physical event propagation model. The first, reflects the rate at which the signal energy (variance) attenuates as a function of the radial distance from the source r. The second, reflects the expected correlation between the signals received (excluding noise) by two sensor nodes n and m based on the separation distance between them d_{nm}.

The signal propagation model used in this section is chosen to reflect the expected "similarity" between measurements of sensor nodes in proximity. In other words, measurements of sensors that are close to each other and close to the source are correlated. On the other hand, sensor nodes that are located far away from the source do not receive any signal information; so even if they happen to fall next to each other their signal measurements become uncorrelated.

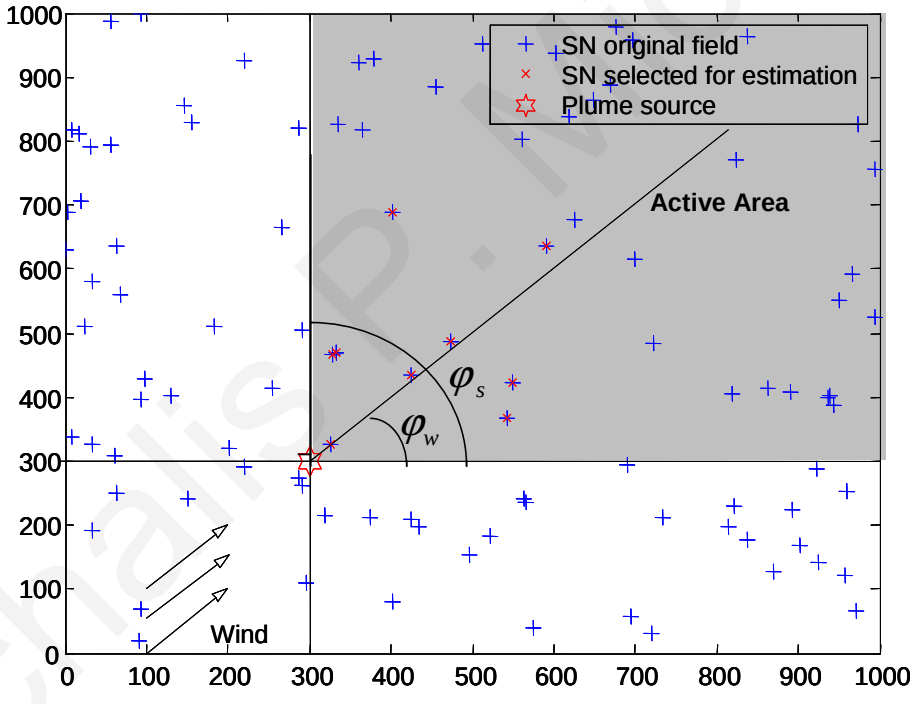


Figure 3.1: A field with 100 randomly placed sensor nodes.

3.3 Directed signal propagation model

Uniform propagation models may be accurate for sources that emit sound or electromagnetic waves, but they are not very accurate for problems where an actual substance is released in the environment (for example in problems of environmental pollution). In this section, we introduce a simplified signal/substance propagation model which takes into consideration environmental conditions like wind or current flow.

Specifically, we assume that there is a draft in a constant direction ϕ_w. As a result, the substance is spread in a limited angle ϕ_s as shown in Fig. 3.1. We assume no environmental changes throughout the propagation, therefore these angles are fixed throughout the experiment.

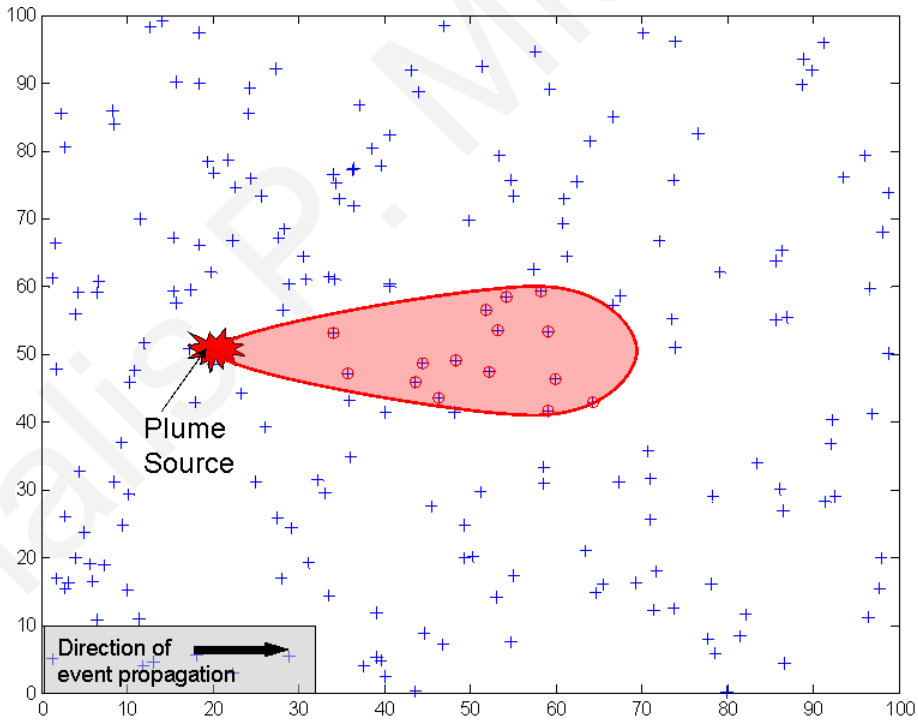


Figure 3.2: Gaussian plume event source.

Definition 3.3.1. *A sensor is said to be in the Active Area $\mathcal{A}$ if it is located in the area defined by the source position, the wind direction and the angles ϕ_w and ϕ_s (see Fig. 3.1).*

Only the sensor nodes located inside $\mathcal{A}$ can receive signal information. Specifically,

$$S_{n,t} = \begin{cases} S_{n,t}, & \text{if } n \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

for $n = 1, \cdots, N$, $t = 1, \cdots, M$. $S_{n,t}$ can be chosen according to the signal propagation characteristics; for acoustical sources we can use the model of Section 3.1, for environmental monitoring applications we could use the stochastic model described in Section 3.2. Alternatively, for creating a plume signal we can use the Gaussian Plume Model [60]. As an example, for one-dimensional spreading, assuming a steady wind direction along the x-axis (i.e., $\phi_w = 0$) and $\phi_s = \pi$, the signal propagation equation becomes

$$S_{n,t} = \frac{c}{\sqrt{2\pi}\sigma_{y_n}} \exp\left[-\frac{1}{2}\left(\frac{y_n - y_s}{\sigma_{y_n}}\right)^2\right], \tag{3.10}$$

where $\sigma_{y_n} = b(x_n - x_s)$ for $n = 1, \cdots, N$, $t = 1, \cdots, M$. The resulting drop-shaped plume spreading is depicted in Fig 3.2.

CHAPTER 4

THRESHOLD OPTIMIZATION FOR EVENT DETECTION

4.1 Overview

This chapter investigates the use of a sensor network for detecting the presence of an event. The sensors monitor the signal emitted from the source and report the existence of the event when the received signal strength is above a certain threshold. In this chapter we derive analytical expressions for the probability of false alarm and the probability of no detection as functions of the threshold. Subsequently, we determine the optimal threshold that trades off the probability of false alarm and the probability of no detection.

4.2 Introduction

This chapter investigates the use of a Wireless Sensor Network (WSN) for detecting the presence of an event source that releases a certain signal or substance in the environment which is then propagated over a large area. The concentration of the substance at the source location is assumed unknown. The sensor nodes are able to measure the substance concentration at their own locations but the measurements are noisy. Based on these concentration readings the sensor nodes use a threshold to decide whether they have detected the event or not.

The question that naturally arises is what threshold to use. Choosing the "right" threshold is

an optimization problem involving the probability of false alarm and the probability of miss. The sensor nodes are continuously monitoring the environment and their measurements are highly uncertain so there will be situations where they will be triggered by just noise. To avoid these highly undesirable situations of false alarms the threshold has to be large enough to minimize their probability of occurrence. Note that in a real time detection system a series of false alarms can be as bad if not worse than no detection. For example, in applications such as environmental monitoring of large areas, false alarms may incur a significant cost because a response crew may have to travel to the suspected area. Also, in many cases, frequent false alarms may make the users simply ignore *all* alarms and as a result even important detected events may go unnoticed. On the other hand we do not want to compromise the possibility of an event going completely undetected so the threshold needs to be small enough to maximize the detection probability of our sensor nodes.

Another motivation for having a threshold stems from the general problem of estimating the location of an event given the various sensor measurements. In source location estimation we are not interested in the noise so it is often beneficial to simply ignore measurements that are below a certain threshold. As we will be showing in Chapter 7, applying a threshold significantly improves the accuracy of the location estimate. Furthermore, in the context of wireless sensor networks it conserves energy since only a fraction of the sensors will report to the sink.

The main contribution of this chapter is the analytical evaluation of the probability of false alarm and the probability of no detection (event miss) in a wireless sensor network in terms of the local detection threshold used by each sensor node. The determination of this threshold is then solved as an optimization problem that minimizes a cost function involving the overall probability of error (either due to missed events or false alarms).

The chapter is organized as follows. First, in Section 4.3, we present the model we have adopted. Then, in Section 4.4, we present the cost function we seek to minimize and the analytical evaluation of the probability of false alarm and the probability of missed detection. In Section 4.5, we present several simulation results. We conclude with a summary of the main results of this chapter in Section 4.6.

4.3 Problem Formulation

For the sensor network that detects an event we use the uniform propagation model described in Section 3.1 of Chapter 3. For deriving the analytical results, we only consider non-boundary sources. In other words, the event source is assumed to be located at a position inside the smaller area $B \subset A$ as shown in Figure 4.1. We point out that for large areas the probability of having a source on the boundary is very small and thus the effect of this assumption is negligible.

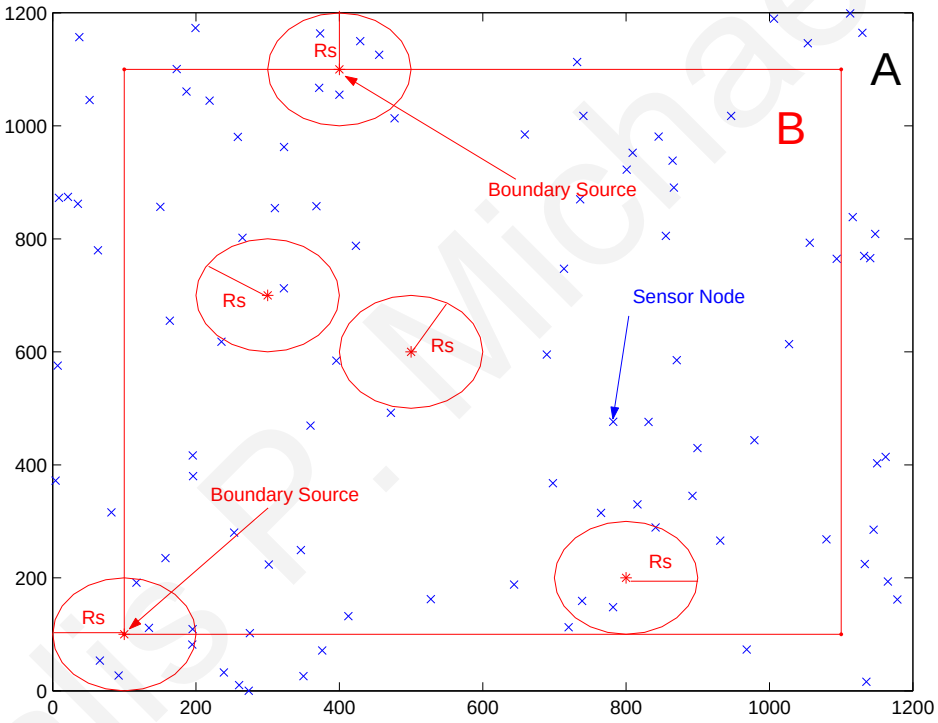


Figure 4.1: A field with 100 randomly placed sensor nodes and 5 sources.

The sensor nodes are assumed to be in an energy-conserve state until triggered by the presence of the signal. When a sensor node i detects something, i.e., it receives a measurement $z_{i,t} > T$, where $i = 1, \cdots, N$, $t = 1, \cdots, M$ and T is a threshold, it wakes up and takes a number of discrete measurements M over a time interval and takes the mean of these measurements

$$\overline{z_i} = \frac{1}{M} \sum_{t=1}^{M} z_{i,t}. \tag{4.1}$$

Then, it compares this value to the threshold T to decide whether to communicate this information to the sink and continue measuring or go back to sleep.

Definition 4.3.1. *Let the **Mean Detector** (MD) denote the distributed detection scheme in which each sensor node independently computes the mean test statistic in (4.1) and compares this to a threshold T in order to determine its alarm status.*

This threshold will be the same for all sensor nodes in the field and will work in a distributed fashion in that each sensor node will use this threshold to decide whether it has detected the event or not. The mean is a sufficient statistic of the sensor data and is the uniformly most powerful test (UMP) in the Neyman-Pearson formulation for a single sensor in composite hypothesis one-sided detection [8].

Taking the mean before communicating the information to the sink is also justified as a way of reducing the amount of data flowing in the sensor network and saving both bandwidth and energy and therefore prolonging the lifetime of the network. Moreover, it is shown in Chapter 7 that there is no loss in accuracy when it is used to estimate the source position compared to the case that every sensor measurement is used instead. Finally, we assume a centralized approach where sensor readings from the alarmed sensors are gathered at the sink using a communication paradigm like directed diffusion [61]. The overall detection of the event will then be decided at the sink if at least one sensor node reports detection.

4.4 Analytical Evaluation of Threshold

To solve for the optimal threshold we seek to minimize the following cost function J representing the overall error in detection as a function of the threshold T:

$$J(T) = w \times P_{fa}(T) + (1 - w) \times P_{nd}(T) \tag{4.2}$$

where P_{fa} is the probability of false alarm, P_{nd} is the probability of no detection and $0 \leq w \leq 1$ is a user specified weight that should be chosen with care according to the application. A small w implies that the application can tolerate more false alarms but it cannot tolerate any missed events. For example, in networks that monitor for toxic terrorist attacks in crowded areas, w could be set close to zero. On the other hand, larger values of w imply that some missed events may be tolerated to reduce the cost of false alarms. For example, in applications such as environmental monitoring of large areas, false alarms may incur a significant cost because a response crew may have to travel to the suspected area. Also, in many cases, frequent false alarms may make the users simply ignore *all* alarms and as a

result important events may go undetected. Finally we point out that alternatively one could also formulate the problem as a constrained optimization problem. For example, minimize the probability of false alarm subject to the probability of no detection being less than some value.

Definition 4.4.1. *Let H_0 represent the noise-only hypothesis and H_1 represent the signal-present detection hypothesis using the statistical hypothesis testing formulation [8].*

Below we go into the details of evaluating the probability of false alarm and the probability of no detection as a function of the threshold T.

4.4.1 Probability of false alarm

In the absence of a source the sensors are measuring just noise which is Gaussian with distribution $\mathcal{N}(0, \sigma_W^2)$. Therefore, the sample mean at each sensor node (4.1) also has a Gaussian distribution, $\overline{z_i} \sim \mathcal{N}(0, \frac{\sigma_W^2}{M})$. The probability of false alarm is the probability that at least one of the sensors mistakenly reports the presence of a source and is given by the following equation:

$$P_{fa}(T) = 1 - \Phi^N \left(\frac{T \times \sqrt{M}}{\sigma_W} \right) \tag{4.3}$$

where $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-\frac{y^2}{2}) dy$ is the probability the Standard Normal Gaussian random variable $\mathcal{N}(0,1)$ is less than x and can be calculated from tables or using Matlab.

The derivation is outlined below:

$$\begin{aligned} P_{fa}(T) &= \Pr\{H_1 | H_0\} \\ &= \Pr\{(\overline{z_i} \geq T) \text{ for at least one sensor } i\} \\ &= 1 - \Pr\{(\overline{z_i} < T) \forall \text{ sensors } i = 1, ..., N\} \\ &= 1 - \prod_{i=1}^{N} \Pr\{\overline{z_i} < T\} \\ &= 1 - \Phi^N \left(\frac{T \times \sqrt{M}}{\sigma_W} \right). \end{aligned}$$

In the above derivation we use independence between the probabilities that the sensor nodes' measurements jointly remain below a certain threshold T. This follows from the fact that noise samples are assumed to be uncorrelated between the sensor nodes [62].

4.4.2 Probability of no detection

In Fig. 4.2 we see a graphical interpretation of the probability of no detection in a randomly created field of 100 sensor nodes. Each circle denotes the detection area of each sensor node as defined by the threshold in the absence of noise and the shaded area represents the locations where an event would go completely undetected. The sensor measurements have a Gaussian distribution $\overline{z_i} \sim \mathcal{N}(\frac{c}{r_i^\alpha}, \frac{\sigma_W^2}{M})$ and they are spatially correlated based on the sensor positions and the distance from the source.

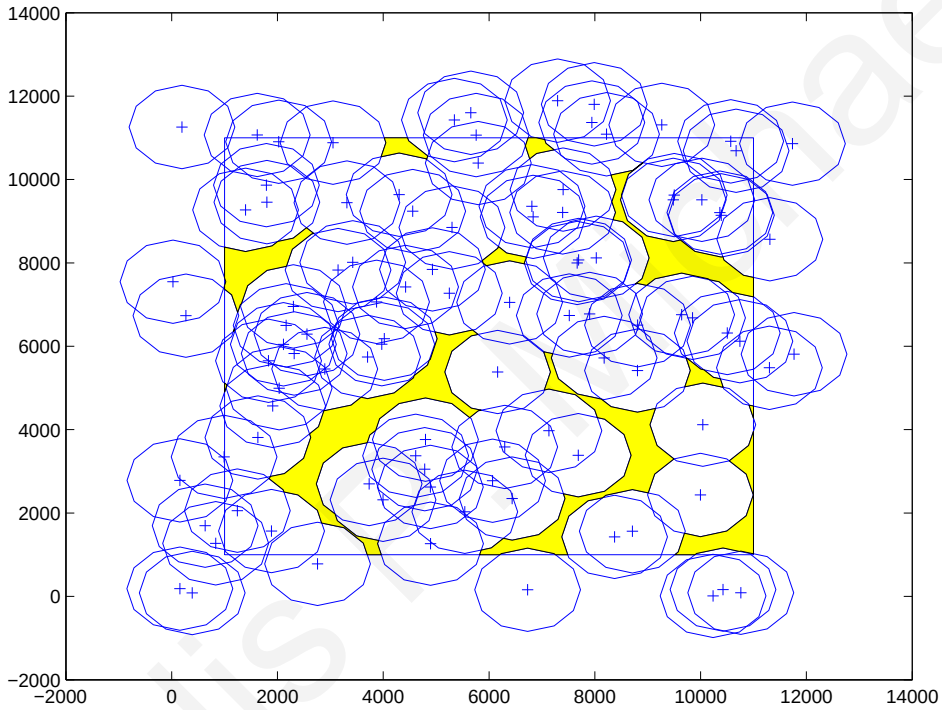


Figure 4.2: Graphical interpretation of probability of no detection.

Definition 4.4.2. *Let $R_s(T)$ denote the radius of a disk centered at the source inside which a sensor node will be alarmed with high probability,* at least *0.5*[1] *(see Fig. 4.1).*

Note that R_s is a function of the prescribed threshold T and the source attenuation model. For the uniform propagation model of Section 3.1, R_s is given by

$$R_s(T) = \sqrt[\alpha]{\frac{c}{T}}. \tag{4.4}$$

[1]Using 0.5 is a convenient way to define $R_s(T)$ when dealing with noise that has symmetric pdf (e.g. Gaussian) because its size becomes independent of the noise variance.

Definition 4.4.3. *Let S_i denote the event that sensor node i falls inside the disk of radius $R_s(T)$ centered at the source (see Fig. 4.1).*

Using independence, the probability of no detection is given by

$$P_{nd}(T) = \prod_{i=1}^{N}(1 - P_{D_i}) \tag{4.5}$$

where P_{D_i} is the probability of detection by sensor i and is computed by conditioning on the event S_i:

$$P_{D_i} = \Pr\{H_1|S_i\} \times \Pr\{S_i\} + \Pr\{H_1|\overline{S_i}\} \times \Pr\{\overline{S_i}\} \tag{4.6}$$

where $\Pr\{\overline{S_i}\} = 1 - \Pr\{S_i\}$. Now,

$$\Pr\{S_i\} = \frac{\pi R_s^2}{A} \tag{4.7}$$

since the sensor nodes are uniformly distributed so all points in area A are equally probable [62] (note that by assumption no sources are located on the boundaries). $\Pr\{H_1|\overline{S_i}\}$ is the probability that sensor i detects the source given that the source is outside its coverage area. For values of α that are large enough, the signal measurement $\left(\frac{c}{r^\alpha}\right)$ away from the source becomes zero and thus, this probability can be approximated with the probability of false alarm for sensor node i which is given by (4.3). Finally, we can derive $\Pr\{H_1|S_i\}$ by conditioning on the radial distance from the source. In the derivation below $R \in [0, R_s(T)]$ is a random variable representing the radial distance from the source with `pdf` $f_{R|S_i}(r|S_i) = \frac{2r}{R_s^2}$. This choice of `pdf` makes all points in the disk around the source equally probable.

$$\begin{aligned}
\Pr\{H_1|S_i\} &= \int_0^{R_s(T)} \Pr\{H_1|S_i, R = r\} f_{R|S_i}(r|S_i) dr \\
&= \int_0^{R_s(T)} \left[1 - \Phi\left(\left(T - \frac{c}{r^\alpha}\right)\frac{\sqrt{M}}{\sigma_W}\right)\right] \frac{2r}{R_s^2} dr
\end{aligned} \tag{4.8}$$

Substituting (4.8),(4.7),(4.3) and (4.4) in (4.6) yields P_{D_i}. Then by substituting this in (4.5) we get the probability of no detection as a function of the threshold T. It is worth pointing out that in high `SNR` situations $\Pr\{H_1|S_i\} = 1$ so the probability of detection P_{D_i} is given by the following equation:

$$P_{D_i} = \Pr\{S_i\} + (1 - \Pr\{S_i\})\, P_{fa_i} \tag{4.9}$$

The $P_{nd}(T)$ requires knowledge of c the amplitude at the source which is usually unknown. Since this parameter is needed in advance to determine the optimal threshold for detection we decided to use the smallest value that would cause an alarm for the specific substance we aim to detect. This ensures that our optimal threshold T_{opt} will be small enough to detect concentrations equal or greater than c. This is also justified in composite hypothesis testing where it is shown that the probability of detection is proportional to the magnitude of the unknown parameter (c) we aim to detect [8].

4.5 Simulation Results

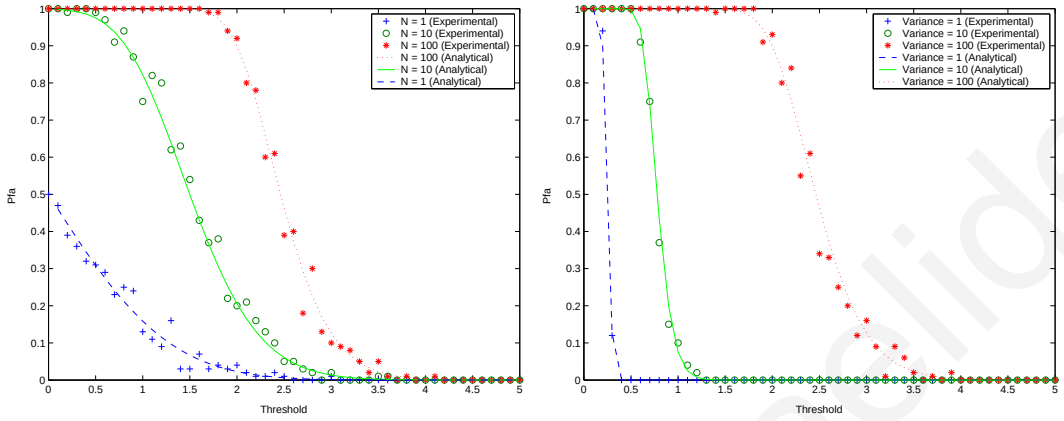
For all subsequent experiments we used a square sensor field of $1km \times 1km$ and assume that the sensor measurements were given by:

$$z_{i,t} = \min\left\{10^6, \frac{10^6}{r_i^2}\right\} + w_{i,t} \tag{4.10}$$

where $i = 1, \cdots, N, t = 1, \cdots, M, r_i^2 = (x_s - x_i)^2 + (y_s - y_i)^2$ and $w_{i,t} = \mathcal{N}(0, \sigma_W^2), \forall i, \forall t$. The experimental results reported were obtained by taking the average over 100 randomly created sensor fields. For obtaining the experimental probability of false alarm (P_{fa}) the sensor nodes were simply exposed to noise and we counted the times that at least one sensor node reported the presence of an event source. For obtaining the probability of no detection (P_{nd}) we randomly placed a source in each sensor field 100 times and counted the number of times that no sensor node reported the presence of the source. For all the experiments we used Matlab.

4.5.1 Probability of false alarm

In the first set of experiments we investigated the validity of (4.3) derived in Section 4.4. In Fig. 4.3(a) we plot the analytical and experimental P_{fa} vs. the threshold T for different numbers of sensor fields- N = 1, 10 and 100. In Fig. 4.3(b) we do the same but this time we use a fixed sensor field with 100 sensor nodes and investigate the effect of changing the noise variance- σ_W^2 =1,10,100. It is evident from the 2 plots that our experimental results are very well in agreement with the analytical ones for all situations tested proving the validity of (4.3).



(a) Vary N (b) Vary σ_W^2

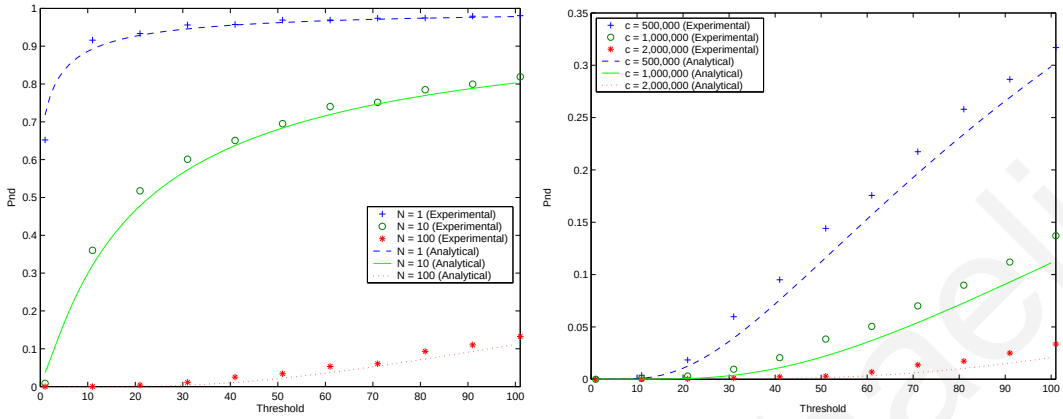
Figure 4.3: Probability of false alarm vs. threshold.

As it can be observed from Fig. 4.3(a), P_{fa} increases with the number of sensors involved. This is expected because the more sensors we have the more likely it becomes for one of the sensors to mistakenly report the presence of an event. Also from Fig. 4.3(b), it is evident that P_{fa} increases with noise variance. This is also expected because noise is what causes the false alarms in the first place. In both of the above situations we would need to increase the threshold to minimize the probability of false alarm. Another way to compensate for the effects of noise would be to increase the number of measurements M before averaging because as it can be seen from (4.3) this acts as a counterweight to noise variance σ_W^2.

4.5.2 Probability of no detection

In the second set of experiments we investigated the validity of (4.5) derived in Section 4.4. In Fig. 4.4(a) we plot the analytical and experimental P_{nd} vs. the threshold T for different numbers of sensor fields- N = 1, 10 and 100. In Fig. 4.4(b) we do the same but this time we use a fixed sensor field with 100 sensor nodes and investigate the effect of changing the initial concentration at the source c. It is evident from the 2 plots that our experimental results are very well in agreement with the analytical ones for all situations tested proving the validity of (4.5). The small deviation in the order of 2% observed in the plots can be attributed to not having enough samples in the experiments to achieve truly uniform distribution of the sensor

nodes[2].



(a) Vary N (b) Vary c

Figure 4.4: Probability of no detection vs. threshold.

As it can be observed from Fig. 4.4(a), the probability of no detection depends largely on the number of sensor nodes in the sensor field. This is expected because the more sensor nodes we have the better coverage we achieve. Noise variance on the other hand does not affect the P_{nd}. Even under very noisy conditions our sensor nodes have a big probability of detecting a source if they are located in the source neighborhood. Increasing the threshold T increases the P_{nd} in all cases.

In Section 4.4 we made the claim that even if we do not know the initial concentration c we can use the minimum concentration we wish to detect in our optimization problem. In Fig. 4.4(b) it is evident that by doing so we will pick a threshold that will allow us to detect all larger concentrations as well. For example let's say that we wanted to detect a minimum $c = 500,000$ with a $P_{nd} \simeq 0.1$. From the plot we would choose $T = 40$ for this problem. This threshold value allows us to detect $c = 1,000,000$ with $P_{nd} \simeq 0.01$ and anything above $c = 2,000,000$ with $P_{nd} \simeq 0$.

[2]Note that Fig. 4.4(a) and Fig. 4.4(b) use different scales for the y-axis. So the deviation results are consistent between the two plots.

4.6 Summary

In this chapter, we investigate a sensor network that monitors for the presence of an event. The network uses a threshold at the sensor level to decide for the existence of the event. We obtain the optimal threshold that minimizes the error involving the probability of false alarm and the probability of no detection. The correctness of the derived equations is shown through simulations of different test case scenarios.

CHAPTER 5

IMPROVED COVERAGE BY EXPLOITING SPATIAL CORRELATION

PART I: THE TWO SENSOR CASE

5.1 Overview

One of the main applications of Wireless Sensor Networks (WSNs) is area monitoring (e.g., environmental monitoring). In such problems, it is desirable to maximize the area coverage which can be achieved by appropriately positioning the sensors (if possible) and/or by increasing the detection range of the sensors. This chapter considers the latter. The emphasis is on pairs of closely spaced sensors that can collaborate in order to increase their collective area coverage. The main contribution of this work is the Enhanced Covariance Detector (ECD) that combines the energy and the covariance information from two sensor nodes utilizing two different thresholds (one for the energy test statistic and another for the covariance). This distributed detector can significantly improve the collective coverage when the two sensors are situated close to each other while maintaining a constant probability of false alarm.

5.2 Introduction

The main objective of this chapter is to investigate collaborative detection schemes at the local sensor level for increasing the area coverage of each sensor and thus increasing the coverage of the entire network. Instead of having each sensor node separately decide its alarm status and report it to the fusion center, in this chapter we focus on *pairs* of nodes that are closely spaced and can exchange information to decide their *collective* alarm status in a decentralized manner, and report that to the fusion center. The aim is for the pair to achieve a larger area coverage than the two individual sensors acting alone.

In this chapter, we consider various collaboration schemes that can be employed by a pair of nodes. A straightforward solution would be for each sensor node to use the energy detector (ED) so that each individual node would determine its alarm status and then the pair would determine its collective decision using an *AND* or an *OR* rule. Another possibility is for the pair to exchange all of their measurements and decide its alarmed status based on the sample covariance. This is referred to as the covariance detector (CD). In this chapter, we also propose a hybrid detection scheme (the Enhanced Covariance Detector (ECD)) that combines the strengths of the Energy Detector (ED) and the Covariance Detector (CD) for closely spaced sensor nodes. By utilizing two different thresholds, one for each detector used, the ECD can improve the overall coverage while attaining the same probability of false alarm as any individual detector.

An important outcome of this work is that it shows that the area coverage achieved by each collaborative detection scheme depends on the distance between the two sensors. When the two sensors are located relatively close to each other, ECD achieves better coverage whereas when the two sensors are spaced further apart, the ED with an *OR* fusion rule can achieve better results. Therefore, for monitoring applications one can organize the sensors of the field into pairs (e.g., closest neighbors) and each pair will decide its alarm status using the best detection algorithm given their relative distance. The cornerstone of this approach is that closely spaced sensors can take advantage of the possible correlation in their measurements to reduce the false alarm probability and extend their coverage.

In summary, the contribution of this chapter is to propose a hybrid detection scheme (the Enhanced Covariance Detector (ECD)) that combines the strengths of the Energy Detector (ED) and the Covariance Detector (CD) for closely spaced sensor nodes. Furthermore, it investigates collaborative detection schemes between pairs of closely spaced sensor nodes and shows that the scheme to be used depends on the distance between the nodes. Finally, the performance of each detector is analyzed.

The chapter is organized as follows. First, in Section 5.3, we present the details of the Optimal Detector (OD), the Energy Detector (ED), the Covariance Detector (CD) and the Enhanced Covariance Detector (ECD) as they apply to a pair of sensor nodes. Then, Section 5.4 analyzes the area coverage for each of the detectors. Section 5.5 presents several simulation results. We conclude with a summary of the main results of this chapter in Section 5.6.

5.3 Collaborative Pairwise Detection Schemes

For this chapter we use the stochastic propagation model described in Section 3.2 of Chapter 3 with $c = 0$. We concentrate on a single pair of sensor nodes that without loss of generality are assumed to be located on the horizontal axis in the middle of the field A and are placed at a distance d apart (at points $(-\frac{d}{2}, 0)$ and $(\frac{d}{2}, 0)$). Under the modeling assumptions used in this chapter, the detection problem can be mathematically described as,

$$\begin{aligned} H_0 &: \mathbf{z} = \mathbf{w} \\ H_1 &: \mathbf{z} = \mathbf{s} + \mathbf{w} \end{aligned}$$

where $\mathbf{s} \sim \mathcal{N}(\mathbf{0}, \mathbf{C_s})$, $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma_W^2 \mathbf{I})$, and $\mathbf{s}$ and $\mathbf{w}$ are independent. The signal covariance matrix $\mathbf{C_s}$ can be calculated using (3.5)-(3.8) as,

$$\mathbf{C_s} = \sigma_S^2 e^{-\frac{r_1+r_2}{\lambda_v}} \begin{pmatrix} e^{-\frac{r_1-r_2}{\lambda_v}} & e^{-\frac{d}{\lambda_c}} \\ e^{-\frac{d}{\lambda_c}} & e^{-\frac{r_2-r_1}{\lambda_v}} \end{pmatrix}. \tag{5.1}$$

For detecting the presence of an event in the field using the pair of sensors, we investigate the following collaborative detection schemes: Optimal Detector (OD), Energy Detector (ED) with either the *AND* or the *OR* fusion rules, Covariance Detector (CD) and Enhanced Covariance Detector (ECD). For each detector, one of the two sensors (referred to as the leader) collects the required information and computes the test statistic.

Definition 5.3.1. *A sensor is "alarmed" if the value of the test statistic $\mathcal{T}$ (depending on the detection algorithm) exceeds a pre-determined threshold.*

Next, we present the specifics of each detector below and derive analytical expressions that approximate their performance (in terms of probability of false alarm and detection).

5.3.1 Optimal Detector (OD)

Assuming the two nodes are synchronized and all signal measurements are available at the leader, the modeling assumptions of this chapter lead to the *general Gaussian detection problem*. The test statistic for the Optimal Detector (OD) for this problem is given in [8] as:

$$\mathcal{T}_{OD} = \frac{1}{M}\sum_{t=1}^{M} \mathbf{z}[t]^T \mathbf{C_s}(\mathbf{C_s} + \sigma_W^2 \mathbf{I})^{-1}\mathbf{z}[t] \geq \gamma_o \tag{5.2}$$

where $\mathbf{z}[t]^T = [Z_{1,t}, Z_{2,t}]$ are the sensor measurements and γ_o is the threshold calculated in a Neyman-Pearson formulation to achieve a pre-specified probability of false alarm constrain. The detection performance of the optimal detector (also known in the literature as estimator-correlator or Wiener filter) is in general difficult to obtain analytically [8]. However, for a large number of samples M, using the Central Limit Theorem (CLT), the test statistic in (5.2) has a Gaussian distribution that depends on the underlying hypothesis. Under the H_0 hypothesis, the probability of false alarm is given by

$$P_{f|OD} = \Pr\{\mathcal{T}_{OD} \geq \gamma_o | H_0\} = Q\left(\frac{\gamma_o - \mu_{0|OD}}{\sigma_{0|OD}}\right) \tag{5.3}$$

where

$$\mu_{0|OD} = \sigma_W^2(b_{11} + b_{22}) \tag{5.4}$$

$$\sigma_{0|OD}^2 = \frac{2}{M}\sigma_W^4\left(b_{11}^2 + b_{22}^2 + \frac{1}{2}(b_{12} + b_{21})^2\right) \tag{5.5}$$

are the mean and the variance of the OD test statistic under H_0 and $[b_{ij}]$ for $i, j = 1, 2$ are the entries of the $\mathbf{B} = \mathbf{C_s}(\mathbf{C_s} + \sigma_W^2\mathbf{I})^{-1}$ matrix in (5.2). Using the above equations, the threshold γ_o can be calculated such that the pair's probability of false alarm constrain $P_{f|OD} = \alpha$ for a specific source location and distribution as,

$$\gamma_o = \sigma_{0|OD}Q^{-1}(\alpha) + \mu_{0|OD} \tag{5.6}$$

The probability of detection can then be obtained numerically using this threshold.

The drawback of the optimal detector is that it requires complete knowledge of the signal distribution (through the matrix $\mathbf{C_s}$) and it is thus impractical for the problem under investigation. Even if we use a grid based exhaustive search method to detect a source at all possible source locations on the grid, we still have to assume knowledge of the signal variance σ_S^2 and calculate a different threshold for each possible source location. Nevertheless, we still consider the OD since its performance can be used to obtain an upper bound on the probability of detection of any other detection scheme.

5.3.2 Energy Detector (ED)

For the Energy Detector (ED) each sensor independently decides first its alarm status based on its own measurements. Then, the 1-bit decisions are gathered at the leader where the detection decision of the pair is decided using an *AND* or an *OR* fusion strategy. Using the *AND* fusion rule, the pair decides that it has detected the event if *both* sensors are alarmed, while using the *OR* strategy detection is decided if *at least one* of the sensor nodes becomes alarmed.

The test statistic used by each sensor is the sample variance[1] of the measurements compared to a constant threshold γ_e,

$$\mathcal{T}_{ED} = \frac{1}{M}\sum_{t=1}^{M} Z_{i,t}^2 \geq \gamma_e \tag{5.7}$$

In the sequel, we will be showing that a different threshold γ_e applies for each fusion rule. Strictly speaking, the test statistic is χ-distributed [8], however, for large enough M, the CLT applies and so the distribution of the test statistic is approximated by a normal distribution which can simplify the computation of the appropriate threshold γ_e such that the false alarm requirement is satisfied. Using the CLT, the probabilities of false alarm $p_{f|ED}$ and detection $p_{d|ED}$ of the ED for a *single* node are given by

$$p_{f|ED} = Q\left(\frac{\gamma_e - M\sigma_{0|ED}^2}{\sqrt{2M}\sigma_{0|ED}^2}\right) \tag{5.8}$$

$$p_{d|ED} = Q\left(\frac{\gamma_e - M(\sigma_{0|ED}^2 + \sigma_{1|ED}^2)}{\sqrt{2M}(\sigma_{0|ED}^2 + \sigma_{1|ED}^2)}\right) \tag{5.9}$$

where

$$\sigma_{0|ED}^2 = \frac{\sigma_W^2}{M} \tag{5.10}$$

$$\sigma_{1|ED}^2 = \frac{\sigma_S^2 e^{-\frac{2r}{\lambda_v}}}{M} \tag{5.11}$$

and $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty \exp\left(-\frac{y^2}{2}\right)dy$ is the right-tail probability of the normalized Gaussian random variable $\mathcal{N}(0,1)$ [8].

[1] Note that the unbiased estimator of the variance is given by $\frac{1}{M-1}\sum_{t=1}^{M} Z_{i,t}^2$ but for large M the difference between the two becomes insignificant.

Fusion Rules for the ED

Next, we consider the case where the decisions of the two sensor nodes are combined. Under H_0 the decisions of the two sensor nodes are independent and the pair's probability of false alarm for the two fusion rules $AND(\wedge)$ and $OR(\vee)$ are:

$$P_{f|ED}^{\wedge} = p_{f|ED}^2 \quad (5.12)$$

$$P_{f|ED}^{\vee} = 1-(1-p_{f|ED})^2 \quad (5.13)$$

Using a Neyman-Pearson formulation we set $P_{f|ED}^{(\cdot)} = \alpha$ and using (5.8) we can derive the threshold that each node in the pair should use depending on the fusion rule.

$$\gamma_e^{\wedge} = \sqrt{\frac{2\sigma_W^4}{M}} Q^{-1}\left(\sqrt{\alpha}\right) + \sigma_W^2 \quad (5.14)$$

$$\gamma_e^{\vee} = \sqrt{\frac{2\sigma_W^4}{M}} Q^{-1}\left(1-\sqrt{1-\alpha}\right) + \sigma_W^2 \quad (5.15)$$

Note that $\sqrt{\alpha} \geq 1-\sqrt{1-\alpha}$ for all $0 \leq \alpha \leq 1$ and since $Q^{-1}(y)$ is a decreasing function of y, to achieve a probability of false alarm α, we need to have $\gamma_e^{\wedge} < \gamma_e^{\vee}$. In other words, the *AND* rule requires a smaller threshold than the *OR* rule. This observation will become significant when we study the coverage of the detectors in Section 5.4.

Under H_1, the test statistics of the two sensor nodes 1 and 2 for large M become 2 correlated Gaussian random variables $\mathcal{T}_{ED|1}$ and $\mathcal{T}_{ED|2}$. To derive the system probability of detection for the energy detector for the two fusion rules we first make the following observation. The *OR* fusion rule can be thought of as $\max\left\{\mathcal{T}_{ED|1}, \mathcal{T}_{ED|2}\right\} \geq \gamma_e^{\vee}$ while the *AND* fusion rule is $\min\left\{\mathcal{T}_{ED|1}, \mathcal{T}_{ED|2}\right\} \geq \gamma_e^{\wedge}$. The exact distribution of the Max and Min of two correlated Gaussian random variables is given in [63] which can be used to obtain the probability of detection for the pair of nodes under the different fusion rules.

5.3.3 Covariance Detector (CD)

For the Covariance Detector (CD), we assume that the two sensor nodes can synchronize their measurements over the next time interval. For the synchronization we are assuming a lightweight scheme like the one proposed in [64] where a pair-wise synchronization is achieved with only three messages. Then, the leader node receives the measurements of the

other sensor and computes the following test statistic:

$$\mathcal{T}_{CV} = \frac{1}{M}\sum_{t=1}^{M}\left\{(Z_{1,t} - \overline{Z_1}) \times (Z_{2,t} - \overline{Z_2})\right\} \geq \gamma_c \tag{5.16}$$

where $\overline{Z_i} = \frac{1}{M}\sum_{t=1}^{M} Z_{i,t}$. The test statistic used is the sample covariance of the measurements between the two sensor nodes compared to a constant threshold γ_c. Note that (5.16) exploits the correlation between the measurements of two sensors that are located close to each other.

For large M, again using the CLT, the test statistic in (5.16) has a Gaussian distribution that depends on the underlying hypothesis:

$$\mathcal{T}_{CV} \sim \begin{cases} \mathcal{N}(0, \sigma^2_{0|CD}), \text{under } H_0 \\ \mathcal{N}(\mu_{1|CD}, \sigma^2_{1|CD}), \text{under } H_1 \end{cases}$$

For the model under investigation,

$$\sigma^2_{0|CD} = \frac{\sigma^4_W}{M} \tag{5.17}$$

$$\mu_{1|CD} = \sigma^2_S e^{-(\frac{r_1+r_2}{\lambda_v}+\frac{d}{\lambda_c})} \tag{5.18}$$

while $\sigma^2_{1|CD}$ is obtained numerically.

Under the H_0 hypothesis, the probability of false alarm for the pair of sensor nodes 1 and 2, is given by

$$P_{f|CD} = \Pr\{\mathcal{T}_{CV} \geq \gamma_c | H_0\} = Q\left(\frac{\gamma_c}{\sigma_{0|CD}}\right) \tag{5.19}$$

where $\sigma_{0|CD}$ is given by (5.17). Using the above equation, the threshold γ_c can be calculated to attain a probability of false alarm constrain $P_{f|CD} = \alpha$,

$$\gamma_c = \sqrt{\frac{\sigma^4_W}{M}} Q^{-1}\left(\sqrt{\alpha}\right) \tag{5.20}$$

It is worth pointing out that the threshold obtained by the CD may be much lower (depending on the noise variance σ^2_W) than the one obtained for the ED in the previous section to attain the same P_f - compare the above equation with (5.14) and (5.15). Under H_1, again using the CLT, the probability of detection for the pair of sensor nodes is given as a function of the threshold γ_c by

$$P_{d|CD} = \Pr\{\mathcal{T}_{CV} \geq \gamma_c | H_1\} \approx Q\left(\frac{\gamma_c - \mu_{1|CD}}{\sigma_{1|CD}}\right) \tag{5.21}$$

where $\mu_{1|CD}$ is given by (5.18) and $\sigma_{1|CD}$ is obtained numerically.

5.3.4 Enhanced Covariance Detector (ECD)

The proposed ECD uses two test statistics; the ED test statistic (5.7) and the CD test statistic (5.16) using the following fusion rule.

$$\{\mathcal{T}_{CD} \geq \gamma_{c_2}\} \wedge \{(\mathcal{T}_{ED|1} \geq \gamma_{e_2}^{\vee}) \vee (\mathcal{T}_{ED|2} \geq \gamma_{e_2}^{\vee})\}. \tag{5.22}$$

In other words, a pair of sensors will become alarmed only if the sample covariance measured by the pair exceeds a threshold γ_{c_2} (different than the threshold used by the CD alone) *and* if either of the sensors becomes alarmed using the ED (i.e., if the recorded sample variance exceeds $\gamma_{e_2}^{\vee}$, different from the corresponding ED threshold). The test statistic is computed by anyone of the two sensor nodes. The two thresholds, γ_{c_2} and $\gamma_{e_2}^{\vee}$ are computed using $P_{f|ED}^{\vee} = \sqrt{\alpha}$ and $P_{f|CD} = \sqrt{\alpha}$ for the individual detectors ED and CD respectively using (5.15) and (5.20). This ensures that the pair's probability of false alarm for the ECD will be $P_{f|ECD} = \sqrt{\alpha} \times \sqrt{\alpha} = \alpha$ and we can directly compare its performance with the other detectors in a Neyman-Pearson formulation. The performance of the ECD in terms of probability of detection can be approximated assuming that the two decisions are independent or can be obtained through simulation.

5.4 Coverage Area Analysis

In this section we formally define the coverage area of the pair of sensors in terms of the P_f and the P_d. We show that the coverage area shape and size depends on the underlying fusion rule.

Definition 5.4.1. *Given the acceptable false alarm probability for each pair is $P_f = \alpha$, "**Coverage Area**" denotes the area around the sensor locations where if a source is present it will be detected by the pair with probability $P_d \geq 0.5$.*

This area is a function of the detection algorithm and the threshold used. Using 0.5 is a convenient way to define the Coverage Area when dealing with test statistics that have symmetric distributions (e.g. Gaussian) because its size becomes independent of the variance of the test statistic. Specifically, when the test statistic has a Gaussian distribution $\sim \mathcal{N}(\mu_1, \sigma_1^2)$ under the H_1 hypothesis, the coverage area can also be represented by

$$P_d = Q\left(\frac{\mu_1 - \gamma}{\sigma_1}\right) \geq \frac{1}{2} \Rightarrow \mu_1 \geq \gamma \tag{5.23}$$

where γ is the appropriate detection threshold. Next we investigate the coverage area for each detector.

5.4.1 ED

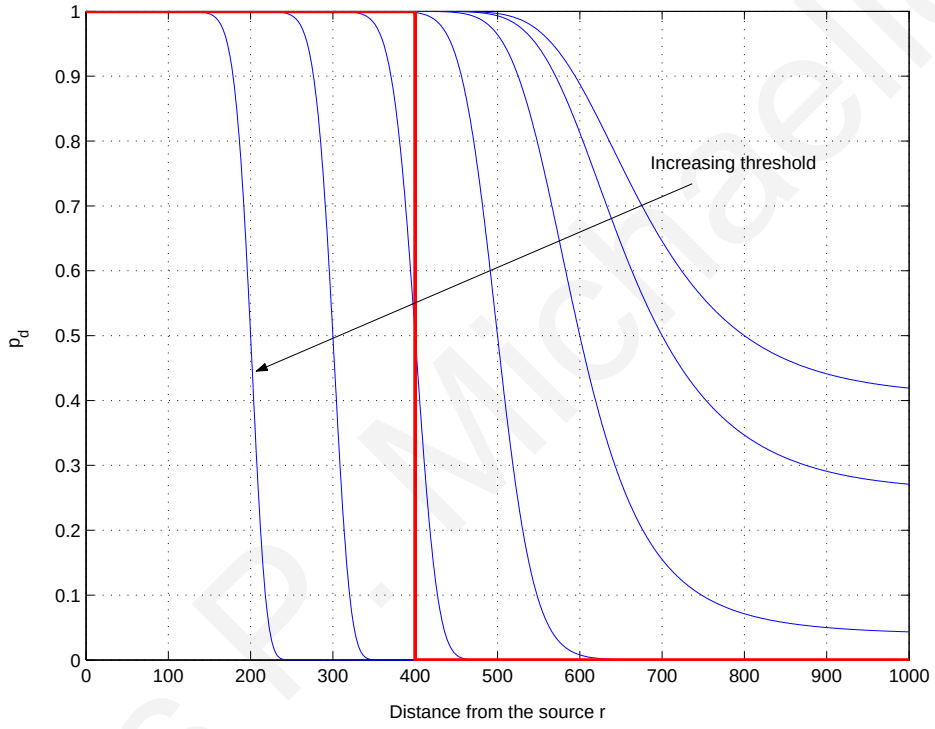


Figure 5.1: Probability of detection vs. distance from the source r for a single sensor using the ED for different values of the threshold γ_e.

From (5.23) and using (5.9), we can calculate the coverage area of a single sensor using the ED which becomes a disc around the sensor node location with radius R_e given by

$$p_{d|ED} = \frac{1}{2} \Rightarrow \sigma_S^2 e^{-\frac{2R_e}{\lambda_v}} + \sigma_W^2 = \gamma_e \Rightarrow R_e = \frac{\lambda_v}{2} \ln\left(\frac{\sigma_S^2}{\gamma_e - \sigma_W^2}\right). \tag{5.24}$$

Note that R_e is a function of the detection threshold γ_e. Fig. 5.1 displays $p_{d|ED}$ versus the distance from the source r for different values of the threshold γ_e. From the figure, it becomes evident that as the threshold γ_e is increased, the $p_{d|ED}$ curve can be approximated by a step function; $p_{d|ED}$ is close to one when the source falls inside the sensor coverage disc while it sharply falls to zero as the source moves outside. Choosing a large threshold is desirable in the context of monitoring applications in order to achieve a fairly small false

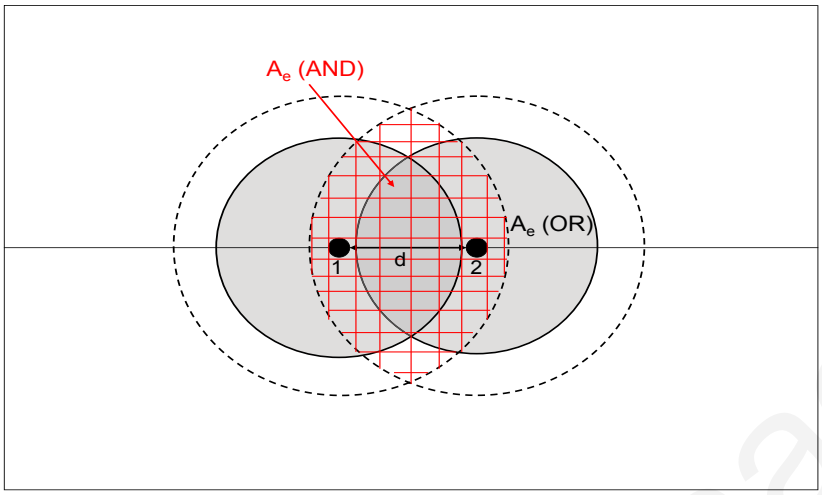


Figure 5.2: Graphical representation (not drawn to scale) of the coverage area of 2 sensor nodes separated by a distance d when using the Energy Detector (ED) with different fusion rules. Using the $OR(\vee)$ fusion rule the coverage area is the union of the two smaller circles (indicated with shaded region) while using the $AND(\wedge)$ the coverage area becomes the intersection of the two larger circles (indicated with a grid).

alarm probability. Assuming that $p_{d|ED}$ takes the form of the step function (see Fig. 5.1), then the coverage area of the pair depends on the fusion rule used. The coverage area is given by the union and intersection between two circles for the $AND(\wedge)$ and $OR(\vee)$ fusion rules respectively (see Fig. 5.2[2]). Note from the figure that the discs for the $AND(\wedge)$ have a larger radius than the ones for the $OR(\vee)$ fusion rule. The reason comes from (5.14) and (5.15) where we clearly see that given $P_f = \alpha$ we get $\gamma_e^{\wedge} < \gamma_e^{\vee}$.

Next we argue that the fusion rule to be used by a pair depends on the distance d between the two sensors. Let $A_e = \pi R_e^2$ denote the coverage area of a single sensor node where R_e is given by (5.24). Also, let $A_e^{\wedge}$ denote the combined coverage area of 2 sensor nodes using the *AND* fusion rule and $A_e^{\vee}$ the coverage area of 2 sensor nodes using the *OR* fusion rule. As argued above, $A_e^{\wedge} > A_e^{\vee}$. When the distance between the two sensors is zero, both the

[2]Note that due to the difference of the actual $p_{d|ED}$ from the step function during the transition from one to zero (see Fig. 5.1) the coverage area approximation in Fig. 5.2 is less accurate near the areas where the two discs intersect. For those situations, the result obtained from the intersection of the two circles (*AND*) overestimates the true coverage while the result obtained from the union (*OR*) underestimates the actual coverage area. Nevertheless, for the calculation of the total coverage area this graphical representation method provides a reasonably accurate approximation.

union and intersection of the circles are the circles themselves, thus the coverage area of the *AND* rule ($A_e^{\wedge}$) is larger. On the other hand, as the distance is increased, there is a distance where the two circles become disjoint and coverage area of the pair becomes zero, while the coverage area of the pair that uses the *OR* rule achieves its maximum equal to $2A_e^{\vee}$. In fact, there exists a distance $\bar{d}$ where the two fusion rules have identical performance. For $d < \bar{d}$ the *AND* rule achieves better coverage whereas for $d > \bar{d}$ the *OR* rule becomes superior.

5.4.2 CD

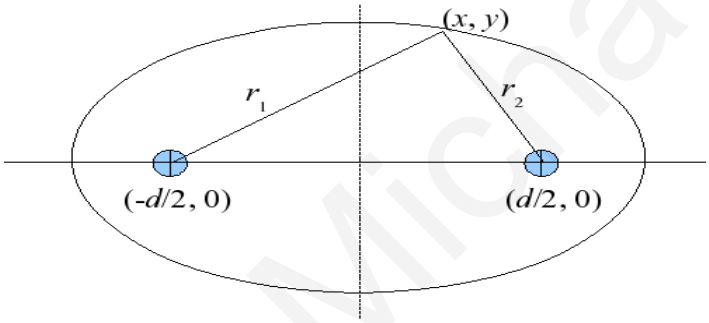


Figure 5.3: CD coverage area.

According to (5.23) and (5.18) and given the threshold γ_c, the perimeter of the coverage area by the two sensors is given by

$$\sigma_S^2 e^{-(\frac{r}{\lambda_v}+\frac{d}{\lambda_c})} = \gamma_c.$$

where $r = r_1 + r_2$. Note that it is necessary that $\sigma_S^2 > \gamma_c$ since $e^{-(\frac{r}{\lambda_v}+\frac{d}{\lambda_c})} \leq 1$ for any $r, d \geq 0$, $\lambda_v, \lambda_c > 0$. Taking logarithms on both sides and rearranging terms,

$$r = \lambda_v \left(\ln \frac{\sigma_S^2}{\gamma_c} - \frac{d}{\lambda_c} \right) = 2a \tag{5.25}$$

Eq. (5.25) is an ellipse with general equation

$$\frac{x^2}{a^2} + \frac{y^2}{a^2 - \frac{d^2}{4}} = 1. \tag{5.26}$$

and therefore the area covered by a sensor that uses the CD is given by

$$A_c = \pi a \sqrt{a^2 - \frac{d^2}{4}}. \tag{5.27}$$

Note that for (5.27) it is necessary that $d < 2a$. If $d = 0$, i.e., the two sensors are located at the same point, then the coverage area is a circle with radius a. Also note that the maximum

coverage area is achieved when $d = 0$. In other words two sensors that use the CD can achieve their maximum coverage when they are located at exactly the same point.

5.4.3 ECD

The ECD essentially takes the intersection of the coverage areas of two detectors: the CD (ellipse shown in Fig. 5.3) and the ED using the *OR* fusion rule (union of 2 circles shown in Fig. 5.2). This intersection operation allows the threshold of each detector to decrease and the individual coverage area to increase without affecting the system probability of false alarm. Since the coverage areas of the 2 detectors have similar shape for closely spaced sensor nodes, taking the intersection of the increased individual coverage areas of the two detectors can improve the coverage area when using the ECD.

5.5 Simulation Results

For all subsequent experiments, we use a square field of 500×500 with 2 sensors placed in the middle of the field separated by a horizontal distance d. We assume that the sensor measurements are given by the propagation model in Section 3.2 of Chapter 3, with $\lambda_v = \lambda_c = 200$, $\sigma_W^2 = \sigma_S^2 = 10$ and $M = 100$. The thresholds for all detectors are calculated using the equations derived in Section 5.3, to obtain a probability of false alarm $P_f = \alpha$ in a Neyman-Pearson formulation. To obtain the experimental probability of detection (P_d), we take the average over a grid of possible source locations that cover the entire field. For each source location we use 500 Monte-Carlo simulations. For all experiments we use Matlab.

Fig. 5.4 shows the performance of the different detectors for $P_f = 0.01$ as we vary the horizontal separation distance d between the 2 sensor nodes. From the plot it is evident that for all detectors, the analytical approximations for the probability of detection -derived in Section 5.3- are very close to the experimental results obtained. The Enhanced Covariance Detector (ECD) outperforms the other distributed schemes for $d < 120$ while for greater separation distances d the Energy Detector (ED) with the *OR* fusion rule becomes the best option. The Optimal Detector (OD) is also shown on the same plot for comparison purposes. To calculate the performance of the OD, we first used (5.3-5.5) to calculate the threshold for each different source location. Then, the probability of detection was obtained numerically using these thresholds. It is interesting to note that the hybrid detection scheme ECD pro-

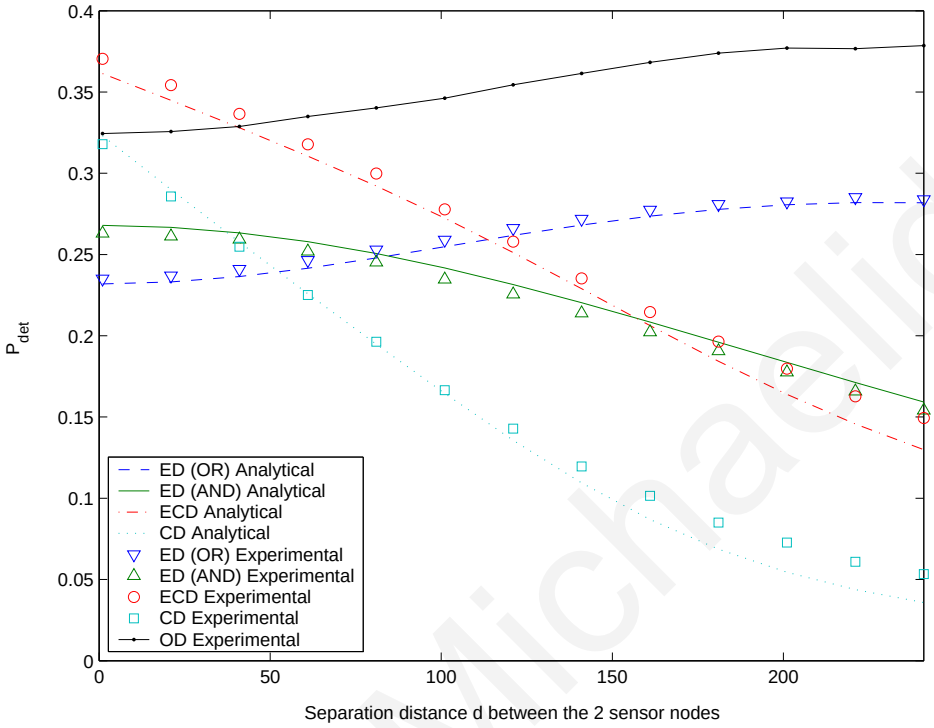
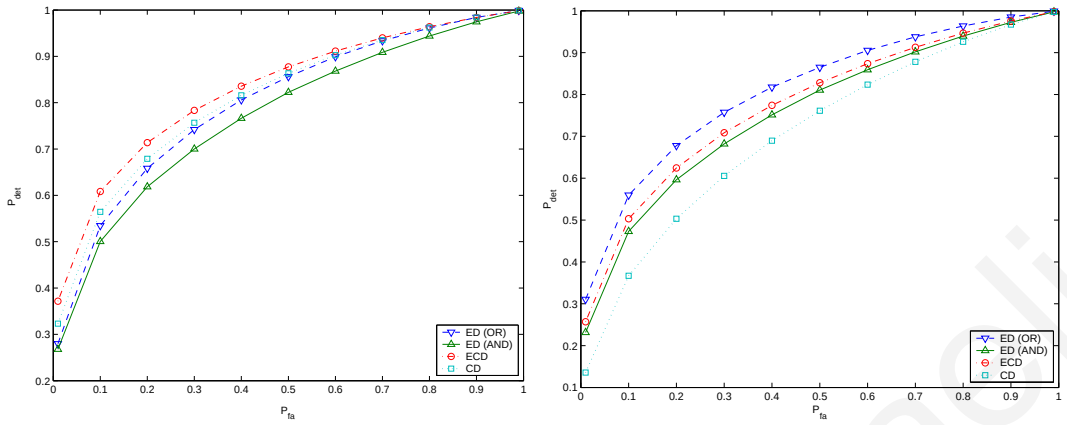


Figure 5.4: Probability of detection vs. separation distance d between the 2 sensor nodes for different detectors given $P_F = 0.01$.

posed in this chapter outperforms the OD for $d < 40$. Remember that the optimal detector refers to a single test statistic compared to a single threshold but assumes full knowledge of the event location and distribution while ECD uses two thresholds.

Next, Fig. 5.5 displays the ROC curves for the different pair detectors for two different separation distances d between the 2 sensor nodes. For small d, the ECD achieves the better results while for large d the ED with the *OR* fusion rule is the best option.

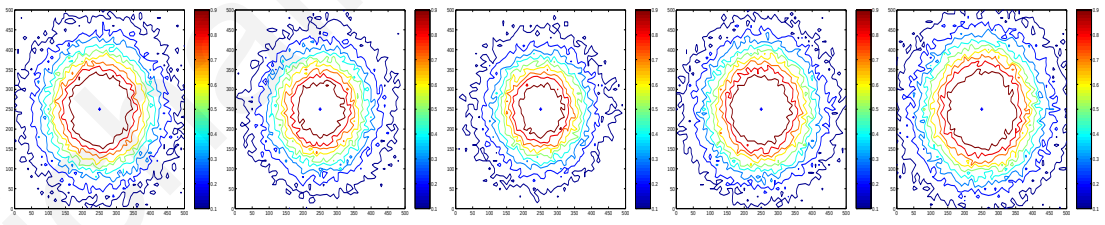
Finally, Figs. 5.6-5.9 show snapshots of the coverage of the different detectors for the specified values of d for the test scenario displayed in Fig. 5.4. There are several things to notice from these plots that are consistent with the analysis in Section 5.4: 1. When the sensor nodes are very close to each other (see Fig. 5.6), the coverage area for all detectors is a circle around the location of the sensor nodes. For this case the hybrid detector ECD has the best coverage followed by CD that essentially achieves the optimal performance (OD). It is also interesting to note that for this case, ED(*AND*) achieves slightly better coverage than ED(*OR*). 2. As the separation distance between the two sensor nodes is increased (see Fig. 5.7-5.8), the coverage area of the CD becomes an ellipse around the sensor nodes' loca-



(a) $d = 1$ (b) $d = 141$

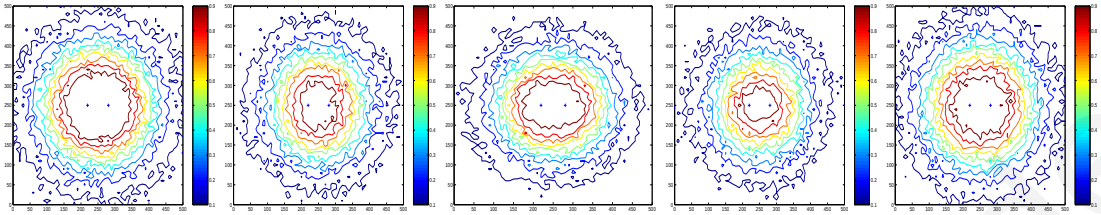
Figure 5.5: Probability of detection vs. probability of false alarm for different detectors given the 2 sensor nodes are separated by distance d.

tions and looks very similar to the one of ED(*OR*)- this explains the motivation behind using the ECD. Please note that while the coverage area of the OD and the ED(*OR*) increases, the coverage area of all other detectors decreases since they depend on either covariance information -CD, ECD-, or simultaneous detection by the two sensor nodes- ED(*AND*). 3. When the sensor nodes are sufficiently apart (see Fig. 5.9) the optimal coverage area becomes two circles around the individual sensor nodes' positions. This is closely resembled by ED(*OR*) which achieves the best coverage out of the distributed detectors. The other detectors do not perform well for this case- this is expected because their performance is based on closely spaced sensor nodes.



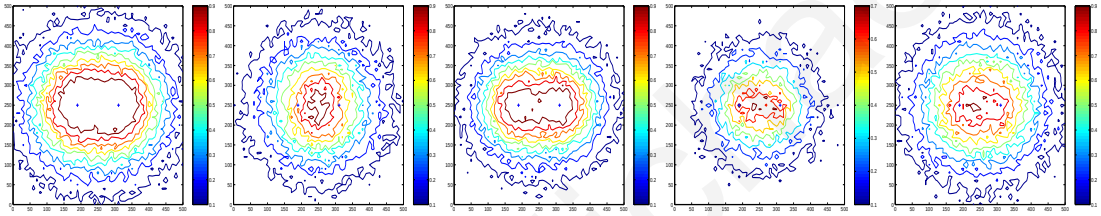
(a) OD (b) ED (*AND*) (c) ED (*OR*) (d) CD (e) ECD

Figure 5.6: Detection snapshots between 2 sensor nodes separated by d=1m.



(a) OD (b) ED (*AND*) (c) ED (*OR*) (d) CD (e) ECD

Figure 5.7: Detection snapshots between 2 sensor nodes separated by d=61m.

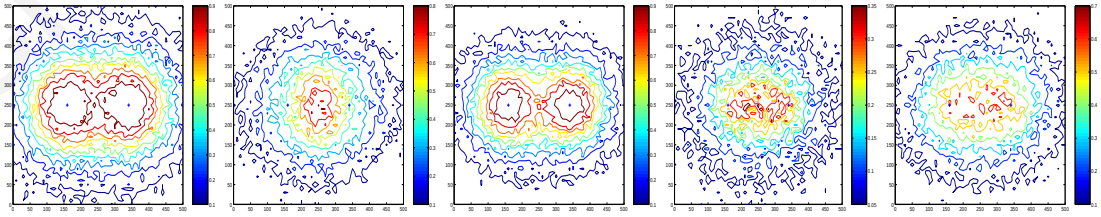


(a) OD (b) ED (*AND*) (c) ED (*OR*) (d) CD (e) ECD

Figure 5.8: Detection snapshots between 2 sensor nodes separated by d=121m.

5.5.1 Preliminary Field Results

In this section we present some preliminary results for the case where we have 100 randomly deployed sensor nodes to cover a 1000×1000 area. Other than that we use the simulation parameters of the previous section. Furthermore, we assume that the fusion center uses a counting rule, thus it decides detection if at least K sensors/pairs become alarmed. Fig. 5.10 displays the ROC curves for the different detectors. $s1 - ED$ refers to the case where each sensor node uses the ED and reports its alarm status to the fusion center which decides detection if at least $K = 1$ nodes become alarmed. $s2 - ED$ is similar to $s1 - ED$ but the



(a) OD (b) ED (*AND*) (c) ED (*OR*) (d) CD (e) ECD

Figure 5.9: Detection snapshots between 2 sensor nodes separated by d=181m.

fusion center decide detection if at least $K = 2$ nodes are alarmed. For $p1 - CD$ and $p1 - ECD$, each sensor node utilizes information from its *closest neighbor* for computing the test statistics ($\mathcal{T}_{CD}$ and $\mathcal{T}_{ECD}$ respectively) and the fusion center decides detection if at least $K = 1$ pairs become alarmed. From the plot it becomes evident, that utilizing collaborative local detection schemes (ECD) can significantly improve the coverage of the WSN especially for small system probabilities of false alarm P_F by exploiting sensor nodes that happen to fall close to each other. Reducing the false alarm rate can preserve valuable energy and extend the lifetime of our WSN while achieving the required coverage performance. We plan to investigate this further as part of our future work.

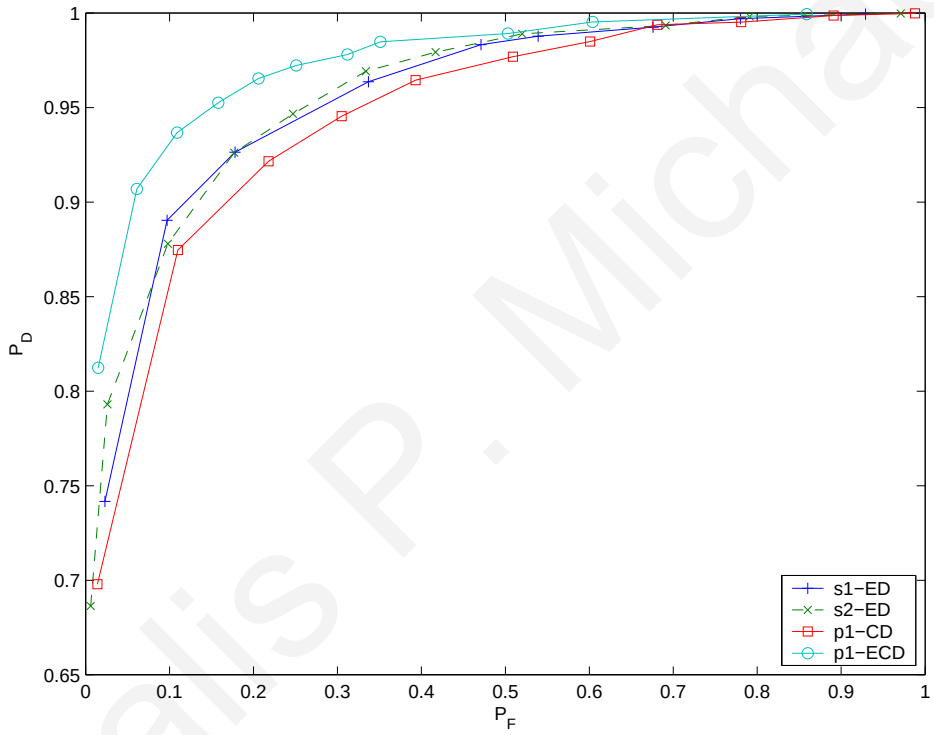


Figure 5.10: ROC plots for different detectors for 100 randomly deployed sensor nodes.

5.6 Summary

In this chapter we investigate distributed detection strategies for improving the coverage (detection performance) of two sensor nodes as we vary the separation distance between them. For closely spaced sensor nodes the proposed Enhanced Covariance Detector (ECD)

can significantly improve the coverage while attaining the same probability of false alarm as any other single distributed detection scheme. For sensor nodes that are further apart using the Energy Detector (with an *OR* fusion rule between the 2 sensor nodes) achieves the best coverage out of the distributed detector schemes tested. As part of our future work, we will be investigating distributed detection strategies for the case where we have three or more sensor nodes collaborating for improving their coverage.

CHAPTER 6

IMPROVED COVERAGE BY EXPLOITING SPATIAL CORRELATION

PART II: THE NETWORK CASE

6.1 Overview

This chapter continues to investigate the problem of event detection using a Wireless Sensor Network (WSN). In Chapter 5, we studied the case with two sensor nodes. In this chapter, we extend these results to the general case with N randomly deployed sensors where the overall detection is decided at the base station using a fusion rule of the general form "event is detected if *at least* K out of N detection reports from alarmed sensors are received". In Chapter 5, we proposed a Covariance Detector (CD) where a pair of sensor nodes would collectively decide detection by evaluating the sample covariance between their measurements. In this chapter, we compare the CD's performance to the Mean Detector (MD) where the test statistic is the sample mean of each sensor node by itself. We show that under certain conditions the CD can exploit the measurement correlation between neighboring sensor nodes to achieve significantly better coverage. The main contribution of this chapter is to propose a Hybrid Detector (HD) for improving the overall coverage of a sensor network. This detector combines the benefits of both the MD and the CD by allowing each sensor to choose which detector to use based on the distance from its closest neighbor. Finally, we show that the results of this chapter have implications on the network topology.

6.2 Introduction

This chapter continues to investigate a Wireless Sensor Network for detecting the presence of an event source that releases a certain signal or substance in the environment which is then propagated over a large area. In Chapter 5, we examined collaborative pairwise detection schemes for the two sensor case. We showed that in order to increase their collective coverage area, the detection strategy became a function of the distance between the two sensors. For closely spaced sensor nodes, we showed that using covariance information as the detection criterion can significantly improve their coverage area. In this chapter, we extend these results to the more general case where we have a field with N randomly deployed sensor nodes. The overall detection is decided at the base station using a fusion rule of the general form "event is detected if *at least* K out of N detection reports from alarmed sensors are received."

The main objective of this chapter is to investigate detection schemes for increasing the overall coverage of the sensor network. Traditionally, the sensing coverage of a sensor node has been represented by a uniform disc around the sensor node location inside which an event was detected with high probability- this is equivalent to the Mean Detector (`MD`) - see Chapter 4. Therefore, the intuition has been to have sensor nodes as far apart as possible in order to limit the overlap between their coverage areas. In many detection scenarios that require random deployment however, it is often the case that some sensor nodes fall close to each other. For those situations, in Chapter 5, we provided the sensor node with an alternative local detection strategy: to compute the measurement covariance with another sensor node located nearby and use this as the deciding factor for detection- we called this the Covariance Detector (`CD`). In this case, the sensing coverage of a sensor node becomes an ellipse whose area depends on the degree of correlation present and the distance from the node's neighbor. Thus, in situations of high correlation for closely spaced sensor nodes the `CD` can achieve a better coverage than the `MD`. The two thresholds for the detectors `MD` and `CD`, can be readily calculated from the false alarm constraint assuming a Neyman-Pearson formulation. Each sensor node in the field can then be programmed with two thresholds and decide which detector to use based on the distance from its closest neighbor- we call this the Hybrid Detector (`HD`). When the problem parameters are known, a sensor node can solve an optimization problem to find the optimal distance threshold for choosing between the `MD` and the `CD`. On the other hand, in situations where the event distribution is completely unknown, we can still evaluate a distance threshold from the "expected distance" between a sensor node and its closest neighbor. Note, that in this context the sensor nodes have no information about the signal monitored and they simply decide which detector to use based on the distance from their closest neighbor. This is a great advantage for sensor networks

because they are often envisioned to monitor areas for the presence of a variety of sources of different strengths and variances.

The main contribution of this chapter is to propose a Hybrid Detector (HD) for improving the overall coverage of a sensor network. This detector combines the benefits of both the MD and the CD by allowing each sensor to choose which detector to use based on the distance from its closest neighbor. Another contribution is to show how the CD has direct implications on the network topology. Specifically, our results indicate that for the CD the best placement would be pairs of sensor nodes placed on a grid configuration. Finally, for all the detectors considered we derive the system probability of false alarm in terms of the detection threshold. This allows a direct comparison of their performance in a Neyman-Pearson formulation.

The chapter is organized as follows. First, in Section 6.3, we present the details of the Mean Detector (MD), the Covariance Detector (CD) and the Hybrid Detector (HD). Then, in Section 6.4, we present several simulation results. We conclude with a summary of the main results of this chapter in Section 6.5.

6.3 Detection

For this chapter we use the stochastic propagation model described in Section 3.2 of Chapter 3 with $\lambda_v = \lambda_c = \lambda$ (for symmetry). Assuming that all signal measurements from the sensor nodes are available centrally at the fusion center, the modeling assumptions of this chapter lead to the *general Gaussian detection problem* which can be mathematically described as

$$\begin{aligned} H_0 : &\quad \mathbf{z} = \mathbf{w} \\ H_1 : &\quad \mathbf{z} = \mathbf{s} + \mathbf{w} \end{aligned}$$

where $\mathbf{s} \sim N(\mu_s, C_s)$, $\mathbf{w} \sim \mathcal{N}(0, \sigma_W^2 \mathbf{I})$, and $\mathbf{s}$ and $\mathbf{w}$ are independent. The signal mean μ_s is an $N \times 1$ vector whose entries can be calculated using (3.5) and C_s is an $N \times N$ signal covariance matrix whose entries can be calculated using (3.6) and (3.7). Since the optimal detector for this problem requires complete knowledge of the signal distribution (see [8]) it cannot be applied to the problem under investigation. So we investigate three heuristic distributed detectors for detecting the presence of an event in the field at the local sensor level:

1. Mean Detector (MD): each sensor independently uses its computed mean to decide

whether it has detected the event or not.

2. Covariance Detector (CD): covariance information between the closest neighbors is the deciding factor for detecting the event.

3. Hybrid Detector (HD): each sensor decides independently between the MD and the CD based on the distance from its closest neighbor.

The overall detection is decided at the base station using a fusion rule of the general form "event is detected if *at least* K out of N detection reports from alarmed sensors are received." Next, we present the specifics of each detector below and then compare their performance (in terms of probability of miss event) in Section 6.4.

6.3.1 Mean Detector (MD)

The test statistic used is the sample mean of the measurements compared to the constant threshold T_m,

$$\overline{Z_n} = \frac{1}{M}\sum_{t=1}^{M} Z_{n,t} \geq T_m \tag{6.1}$$

for $n = 1, \cdots, N$, $t = 1, \cdots, M$. The sensor nodes are assumed to be in an energy-conserve state until triggered by the presence of the signal. When a sensor node detects something, i.e., it receives a measurement $Z_{n,t} > T_m$, it wakes up and takes a number of discrete measurements M over a time interval. Then it makes a decision whether it has detected the event or not using the mean test statistic shown in (6.1). An alarmed node sends a message to the sink which decides that an event is present if *at least* K such messages are received.

The performance of the MD using an N choose K fusion rule has been analyzed in [65] for the case with $\sigma_S^2 = 0$, $\sigma_W^2 = 1$ and $M = 1$. From the analysis presented, one can obtain the probability of false alarm (P_F) as shown below for the general case MD:

For N randomly deployed sensors using the MD, the *system* probability of false alarm is given by

$$P_F = \sum_{i=K}^{N} \binom{N}{i} p_{f_{\text{MD}}}^{i} [1 - p_{f_{\text{MD}}}]^{N-i} \tag{6.2}$$

where,

$$p_{f_{\text{MD}}} = Q\left(\frac{T_m}{\sigma_{0_{\text{MD}}}}\right) \tag{6.3}$$

In the above equation, $\sigma^2_{0_{\text{MD}}} = \frac{\sigma^2_W}{M}$ is the variance of the test statistic (6.1) under the H_0 hypothesis.

6.3.2 Covariance Detector (CD)

Definition 6.3.1. *Let $d(n,m)$ define the Euclidean distance between two sensor nodes n and m for $n,m = 1...N$. Sensor node m is defined to be the "closest neighbor" of sensor node n if no other sensor node can be found located inside a disc of radius $d(n,m)$ centered at the location of sensor n.*

The test statistic used is the sample covariance of the measurements between two sensor nodes that are closest neighbors compared to a constant threshold T_c,

$$\overline{CV}_{n,m} = \frac{1}{M}\sum_{t=1}^{M}\{(Z_{n,t} - \overline{Z_n}) \times (Z_{m,t} - \overline{Z_m})\} \geq T_c \tag{6.4}$$

for $n,m = 1,\cdots,N$, $t = 1,\cdots,M$. The test statistic is computed by sensor node n and sensor node m is its closest neighbor.

When sensor node n is alarmed by the presence of the event, it also wakes up its closest neighbor m if it is still asleep and they synchronize their measurements over the next time interval. For the synchronization we are assuming a lightweight scheme like the one proposed in [64] where a pair-wise synchronization is achieved with only 3 messages. Then sensor node n receives all of m's measurements and computes the test statistic in (6.4) to determine whether to become alarmed and send a signal to the sink. The overall detection of the event will then be decided at the sink if it receives *at least K independent* detection reports from any of the sensor nodes. Note that if two sensors are the closest neighbors of each other, then the test statistic evaluated by the two sensors will be identical. In this case, the fusion center will consider the reports from such a pair as a single report.

This is also a distributed detector since each node only needs local information (its own measurements and measurements of its closest neighbor). Note that (6.4) exploits the correlation between the measurements of two sensors that are located close to each other.

Definition 6.3.2. *We define $\mathcal{N}_2$ as the set of pairs of sensor nodes that are closest neighbors of each other. In other words,*
$\{$pair $(n,m) \in \mathcal{N}_2\} \Leftrightarrow \{n$ closest neighbor of m AND *m closest neighbor of $n\}$.*
We define $\mathcal{N}_1$ as the set of all remaining sensor nodes.

Note that $|\mathcal{N}_1| + 2|\mathcal{N}_2| = N_1 + 2N_2 = N$. For pairs of sensor nodes in $\mathcal{N}_2$ the test statistic (6.4) is exactly the same for both involved sensor nodes, so we have to account for this when we derive the P_F for the CD. In other words, there are only $N_1 + N_2$ "unique pairs" of sensor nodes that we need to consider.

Lemma 6.3.1. *For N randomly deployed sensors using the CD, the* system *probability of false alarm is given by*

$$P_F = 1 - \sum_{i=0}^{K-1} \binom{N_1 + N_2}{i} p_{f_{CD}}^i [1 - p_{f_{CD}}]^{N_1+N_2-i} = \sum_{i=K}^{N_1+N_2} \binom{N_1 + N_2}{i} p_{f_{CD}}^i [1 - p_{f_{CD}}]^{N_1+N_2-i} \tag{6.5}$$

where $p_{f_{CD}}$ is the probability of false alarm for a single pair of sensors derived in Chapter 5 as

$$p_{f_{CD}} = Q\left(\frac{T_c}{\sigma_{0_{CD}}}\right) \tag{6.6}$$

In the above equation, $\sigma^2_{0_{CD}} = \frac{\sigma_W^4}{M}$ is the variance of the test statistic (6.4) under the H_0 hypothesis.

Proof: Under the H_0 hypothesis the sensor nodes are measuring noise, so their measurements are uncorrelated and therefore independent. P_F is the probability that at least K sensor nodes report the presence of the event. The proof follows if we only consider the unique pairs of closest neighbors in the derivation (see Def. 6.3.2).

Using (6.5) and given a maximum false alarm constraint ($P_F \leq q$), the threshold T_c can be derived. This threshold will be the same for all sensor nodes in the field and will work in a distributed fashion in that each unique pair of sensor nodes will use this threshold to decide whether to become alarmed or not.

6.3.3 Hybrid Detector (HD)

From the simulation results presented in Chapter 5, it is evident that the CD performs well if two sensors are located close to each other. On the other hand, if a sensor is "isolated", i.e., its *closest* neighbor is located at a distance that is greater than a threshold D_h, then the MD becomes a better option[1]. The Hybrid Detector (HD) attempts to combine the strengths of these two detectors in a way such that the false alarms are not adversely affected. Thus,

[1]Isolated from a detection perspective. Assumption 4 in Chapter 3 still holds, i.e., the node is within the communication range of at least one of its neighbors.

it uses the CD only when the sensor nodes are close to each other to take advantage of the possible spatial correlation between sensor measurements. In situations where a sensor node is isolated from its neighbors, it relies on its own measurements for detection so it uses the MD. For the HD, we only consider the N_1+N_2 "unique pairs" of sensor nodes as per definition 6.3.2. This choice, facilitates the comparison between the different detectors. Each sensor node i, $n = 1, ..., N_1 + N_2$, compares the distance to its *closest* neighbor d_n, to D_h to decide which detector to use. If $d_n \leq D_h$ the sensor node uses the CD, otherwise it uses the MD.

Lemma 6.3.2. *For the HD the probability of false alarms P_F is given by:*

$$P_F = 1 - \sum_{i=0}^{K-1} \sum_{x=0}^{i} \binom{N_m}{x} \binom{N_c}{i-x} p_{f_{MD}}^{x} p_{f_{CD}}^{i-x} [1 - p_{f_{MD}}]^{N_m - x} [1 - p_{f_{CD}}]^{N_c - i + x} \tag{6.7}$$

where N_m is the number of sensor nodes using the MD, N_c is the number of sensor nodes using the CD. $p_{f_{MD}}$ is given by (6.3) and $p_{f_{CD}}$ is given by (6.6).

The proof is included in the appendix.

Corollary 6.3.1. *Using $T_c = \sigma_W \times T_m$ keeps the probability of false alarm the same for all three detectors: MD , CD and HD.*

Proof: We equate $p_{f_{MD}}$ and $p_{f_{CD}}$ given by (6.3) and (6.6) respectively and the result follows. This choice of thresholds keeps the probability of false alarm the same for all 3 detectors and facilitates the comparison between them.

Finally the distance threshold for the Hybrid Detector (D_h) is what essentially determines N_m and N_c and its value is an optimization problem.

Lemma 6.3.3. *To maximize the area covered by itself, a sensor node will use the CD if the distance from its closest neighbor is less than D_h or the MD otherwise. The optimal distance threshold D_h is determined by solving the following equation,*

$$d^3 - 5\Lambda d^2 + 8\Lambda^2 d - 4\Lambda^3 + \frac{4c^2}{\Lambda T_m^2} = 0, \tag{6.8}$$

where $\Lambda = \frac{\lambda}{2} \ln \frac{\sigma_S^2}{T_c}$.

The proof is included in the appendix.

Thus, if a single sensor has estimates of the parameters T_m, T_c, c, σ_S and λ it can solve equation (6.8) to determine the distance threshold D_h. For a variety of monitoring applications,

however, the event distribution is completely unknown - in terms of c, σ_S and λ. For those situations, our results indicate that the optimal D_h can be approximated as the "expected" distance from the closest neighbor given by the following equation,

$$\hat{D}_h = \mathbb{E}[D] = \int_0^{\sqrt{\frac{A}{\pi}}} u f_D(u) du \tag{6.9}$$

where $f_D(.)$ is given by the following Lemma:

Lemma 6.3.4. *In a randomly deployed sensor network with N sensors over an area A the distance of a sensor node from its closest neighbor can be described by a random variable D with:*

$$cdf: F_D(u) = 1 - \left(1 - \frac{\pi u^2}{A}\right)^{N-1} \tag{6.10}$$

$$pdf: f_D(u) = \frac{2\pi(N-1)u}{A}\left(1 - \frac{\pi u^2}{A}\right)^{N-2} \tag{6.11}$$

The proof is included in the appendix. We know that on average, the CD is expected to have a better performance when sensor nodes are close to each other and the MD when a sensor node is more "isolated" from its neighbors. Since the event distribution is completely unknown, we can only base our decision of which detector to use on the distance from the closest neighbor the distribution of which is given by Lemma 6.3.4 for a randomly deployed sensor field. So on average, we should use the MD when the distance from the closest neighbor is larger than the mean distance and the CD otherwise.

Using D_h a sensor node can essentially maximize the area covered by itself. However, we point out that the above solution provides only a local maximum (i.e., the coverage area of a *single* sensor) and it does not mean that it can achieve the maximum coverage of the entire field which is also a function of the topology since there may be overlap between the areas of neighboring sensors and the amount of overlap differs based on the detection algorithm used (in MD the coverage area of every sensor is always a circle whereas for CD it is an ellipse). In principle, one can identify various scenarios and derive the optimal D_h based on them. However, the simulation results indicate that the solution of (6.8) provides a fairly good lower bound to the global optimal threshold.

6.4 Simulation Results

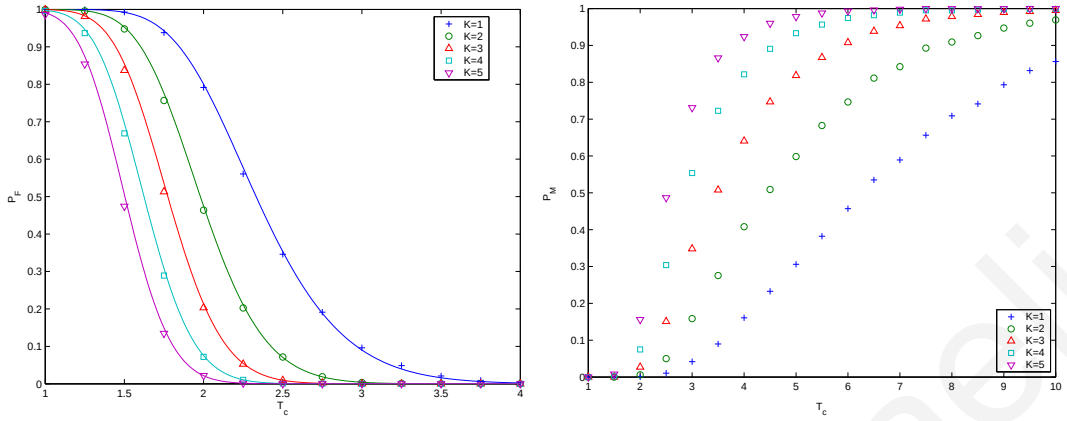
For all subsequent experiments, we use a square sensor field of $1km \times 1km$ with N randomly deployed sensors and assume that the sensor measurements were given by the propagation model in Section 3.2. Also, it is assumed unless otherwise specified, that $c = 1000$, $\lambda = 100$, $\sigma_W^2 = \sigma_S^2 = 10$ and $M = 100$. To obtain the experimental probability of false alarm (P_F) the sensor nodes were simply exposed to noise and we counted the times that at least K sensor nodes reported the presence of an event source. For obtaining the experimental probability of miss (P_M) we counted the number of times that fewer than K sensor nodes reported the presence of the source. We took the average over a grid of possible source locations that covered the entire field and for each source location we used 500 Monte-Carlo simulations. For all the experiments we used Matlab.

6.4.1 CD Performance Evaluation

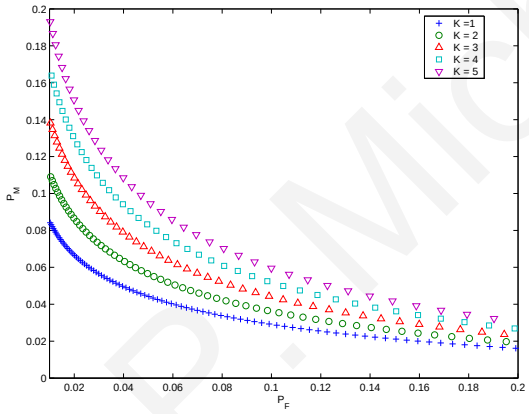
In the first set of experiments we investigate the validity of Lemma 6.3.1 for the performance evaluation of the Covariance Detector. In Fig. 6.1(a) we plot the analytical and experimental P_F vs. the detection threshold T_c for different fusion rules- $K = 1 - 5$. It is evident from the plots that the experimental results match the analytical. Fig. 6.1(b) displays the experimental P_M vs. the detection threshold T_c for different fusion rules- $K = 1 - 5$. In Fig. 6.1(c), we plot the experimental P_M vs. the P_F for the same test scenario. From the plot it is evident that the fusion rule $K = 1$ achieves the best coverage. Similar results were obtained when varying the different problem parameters. For this reason, from now on for the remaining simulations we decided to use the OR fusion rule ($K = 1$).

6.4.2 CD vs. MD

In this section we attempt to give better insight on why the `CD` outperforms the `MD` in situations of High Correlation with a specific test case study of a randomly deployed field with 100 sensors displayed in Fig. 6.2. We set the false alarm constraint to $P_F = 0.001$ and use equations (6.2) and (6.5) to calculate the thresholds of the 2 detectors (we use $T_m = 1.35$ and $T_c = 4.19$). We use the correlation model (3.8) in Section 3.2 with $\lambda = 200$. We then run a series of experiments where each possible source location is chosen from a grid of points that covers the entire area (i.e., there is a grid point every 10m). For each different source location we run 100 simulations and calculate the experimental P_M at that location.



(a) P_F vs. Threshold (b) P_M vs. Threshold

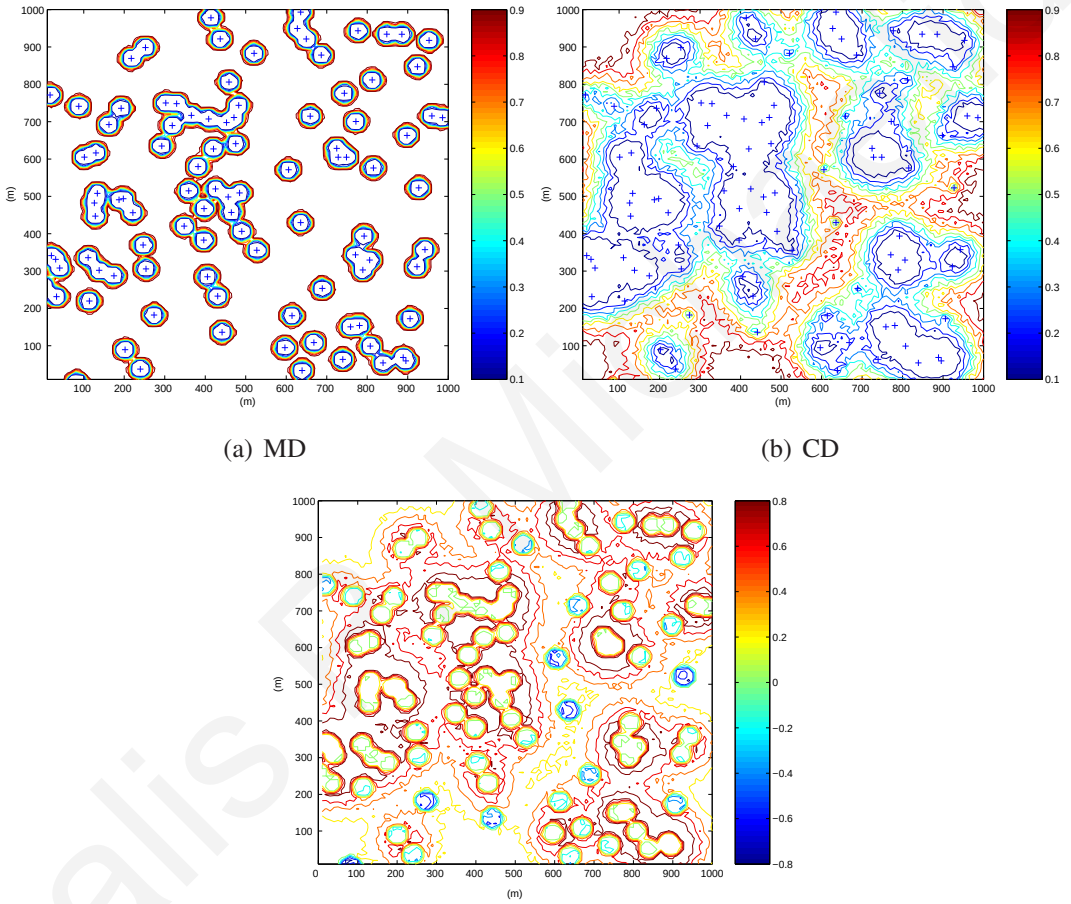


(c) P_M vs. P_F

Figure 6.1: Performance evaluation for CD using N choose K fusion rule. We use $c = 1000$, $\lambda = 150$, $\sigma_S^2 = 20$ and $\sigma_W^2 = 10$.

The resulting contour plot for the P_M of the MD is displayed over the sensor field in Fig. 6.2(a) while for the CD is displayed in Fig. 6.2(b). The red color on the first 2 contour plots corresponds to the value 1 and the blue color to the value 0. The blue crosses are the sensor nodes. From Fig. 6.2(a) we see that using the MD the strict P_F constraint only allows detection in the immediate neighborhood of the sensor nodes and if the source is placed anywhere else we have a miss event with probability 1. On the other hand from Fig. 6.2(b) it is evident that CD has significantly better detection in places with high sensor density and allows the sensor nodes to exploit the close distances to their neighbors for better detection results. The reason is that highly correlated data from nodes in proximity is very useful in reducing P_F. The average probability of miss for the mean detector is $P_{M_{\text{MD}}} = 0.789$ while for the covariance detector $P_{M_{\text{CD}}} = 0.349$ so the probability of miss was reduced by almost 50%!. Finally the

difference of the two $(P_{M_{\text{MD}}} - P_{M_{\text{CD}}})$ is displayed in Fig. 6.2(c). On the difference contour plot red color means positive values and blue negative. From the plot it is evident that while for densely deployed sensor nodes CD achieves significantly better results, for isolated sensors the MD is better. This explains the motivation behind the hybrid detector HD which is further investigated in the next section.



(a) MD (b) CD

(c) MD-CD

Figure 6.2: Probability of miss for different detectors.

6.4.3 Hybrid Detector

This section investigates the use for the Hybrid Detector HD as a way to improve the performance of the MD and the CD in terms of the probability of miss for the test case given in the previous Section 6.4.2. Fig. 6.3 shows the probability of miss vs the threshold D_h for

different values of parameter constant λ. Based on the algorithm described in Section 6.3.3 the distance threshold D_h is what determines the number of sensor nodes that use the MD and the number that use the CD. So $D_h = 0$ means that all sensor nodes use the MD and $D_h = 150$ essentially means that all sensor nodes use the CD- (this was found experimentally for the particular sensor field under investigation).

Table 6.1: Optimal D_h for different λ

	$\lambda = 50$	$\lambda = 100$	$\lambda = 150$	$\lambda = 200$
Solution of (6.8)	-12	29	59	84
Experimental	0	35	65	100

From Fig. 6.3 one can determine the optimal D_h threshold for the different λ parameters. As seen from the figure, for $\lambda = 50$, the MD (i.e., $D_h = 0$) achieves the minimum probability of miss. For the remaining values of λ, the optimal D_h is summarized in Table 6.1. The table also shows the real solution of (6.8) (the other two roots are complex numbers). For $\lambda = 50$ the optimal distance is negative which indicates that there is no valid solution to (6.8) and thus, $D_h = 0$ should be used which is equivalent to saying that the MD should always be preferred in low correlation conditions. For all other cases tested, it is evident from the table that the analytical results are close to the experimental. So when the event distribution is known we can solve the optimization problem given by Lemma 6.3.3 to derive the optimal D_h.

Next, let's consider what happens when the event distribution is completely unknown. To simulate this, we assume that both c and λ are uniform random variables and find the average probability of miss for 1,000 instances of the variables in 100 different randomly deployed fields. Note, that by changing c we only affect the performance of MD, the same way λ only affects the CD. The distributions where chosen to cover the entire range of P_M for both detectors from 0 to 1. We used $c \sim U[500 - 8, 500]$ and $\lambda \sim U[100 - 250]$. Fig. 6.4 shows the results of this test. From the plot it is evident that the optimal experimental $D_h = 50$. Note that if we use (6.9) we get $\hat{D}_h = 50.1$ which is in agreement with the result obtained experimentally.

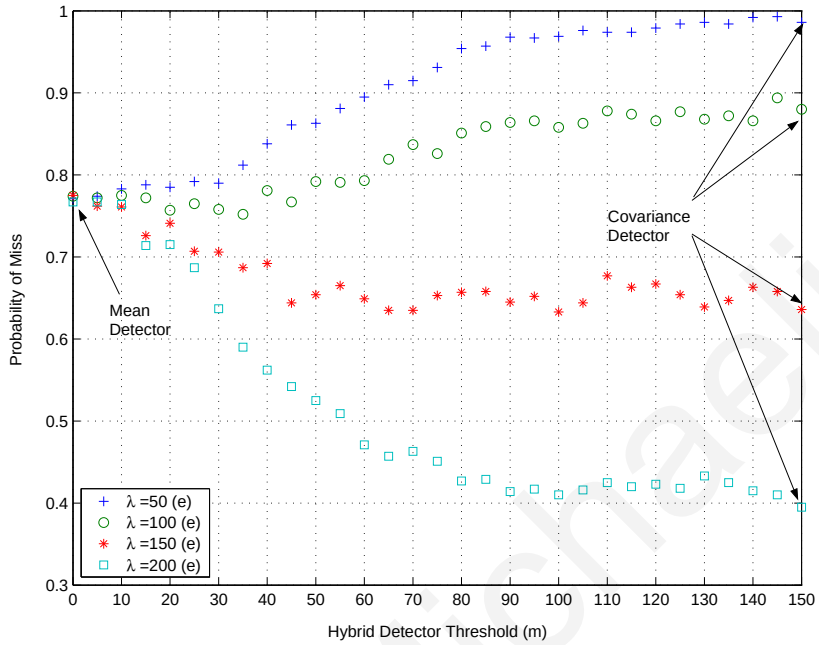


Figure 6.3: Probability of miss vs. threshold for HD.

6.4.4 Topology Implications

In this section we investigate the performance of the CD with respect to the sensor node topology. In papers in the literature that do not consider spatial correlation, one can usually find a grid deployment that outperforms any random deployment. The intuition is that random deployment may have significant overlap between the coverage areas of two neighboring sensors which in general does not provide any new information. On the other hand a grid deployment tries to minimize the overlap.

In this experiment we consider a 1000×1000 field with a grid in the middle. This grid consists of 10 rows and 10 columns spaced at a distance of g meters apart, where g is the grid size. At every intersection of a row with a column we place a sensor, i.e., there are 100 sensors. In other words, every sensor -except the ones on the perimeter of the grid- has four neighboring sensors at a distance g (north, south, east, west). Fig. 6.5 shows the probability of miss of the MD and CD for a fixed probability of false alarm as a function of the grid side. In this figure, the black dots indicate the performance of the MD for grid deployments of various g; the black semi-dotted horizontal line indicates the performance of the MD when the sensors are randomly deployed (average of 100 deployments). From these two plots it is evident that a grid deployment with a grid side $g \geq 50$ outperforms

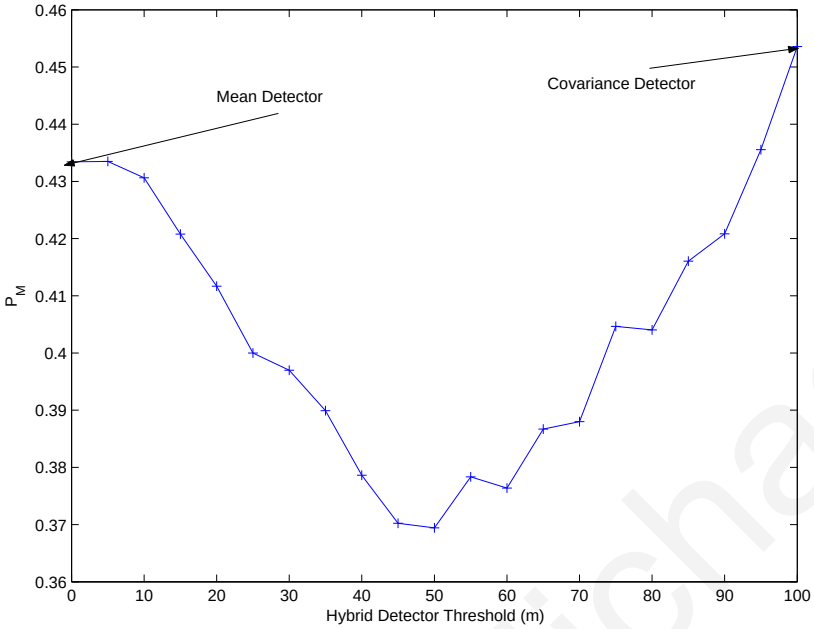


Figure 6.4: Distance threshold optimization for HD.

the random deployment. Similar experiments are performed for the CD for different values of λ. For $\lambda = 50$ (blue crosses and blue horizontal line) the minimum probability of miss is achieved when the grid side $g = 40$m. Increasing the grid side beyond 40m deteriorates the detector's performance since the correlation between neighboring sensors decreases. The blue horizontal line corresponds to the probability of miss for random sensor deployment and we see the if the grid side $25 \leq g \leq 55$, then the grid deployment outperforms the random deployment. Similar results are recorded for $\lambda = 100$ (green circles and horizontal line) and we also see that the grid deployment outperforms the random deployment if $45 \leq g \leq 60$. The results for high correlation, $\lambda = 150$ (red stars and horizontal line) and $\lambda = 200$ (cyan squares and line) are different. The grid deployment achieves a minimum probability of miss at $g \simeq 70$ for $\lambda = 150$ and at $g \simeq 85$ for $\lambda = 200$. In both cases however, note that a random deployment can also achieve the minimum probability of miss. In other words, for high correlation scenarios, there is no need to place the sensors on a grid since a random deployment can perform as good or better than any grid deployment.

Next, motivated by (6.14) where it is indicated that the maximum area coverage of a sensor is achieved when its closest neighbor is located at a distance $d = 0$ (i.e., the two sensors are located at exactly the same point) we experiment with topologies that involve two sensors at exactly the same point. So, for this experiment we place a 7×7 grid in the middle of the field and place 2 sensors at each grid point (i.e., a total of 98 sensors). The grid side was

selected such that the entire field was covered. Fig. 6.6 shows the contour plot for the P_M of the CD over the sensor field with the sensor node pair positions indicated with blue crosses for different values of correlation constant λ. For all cases tested, for the CD, placing pairs of sensor nodes in a grid configuration outperforms random and regular grid deployments (see Fig. 6.5). The most significant improvement was observed for the $\lambda = 150$ scenario, where the average $P_M = 0.28$, which translates to about 50% improvement when compared to the random deployment case.

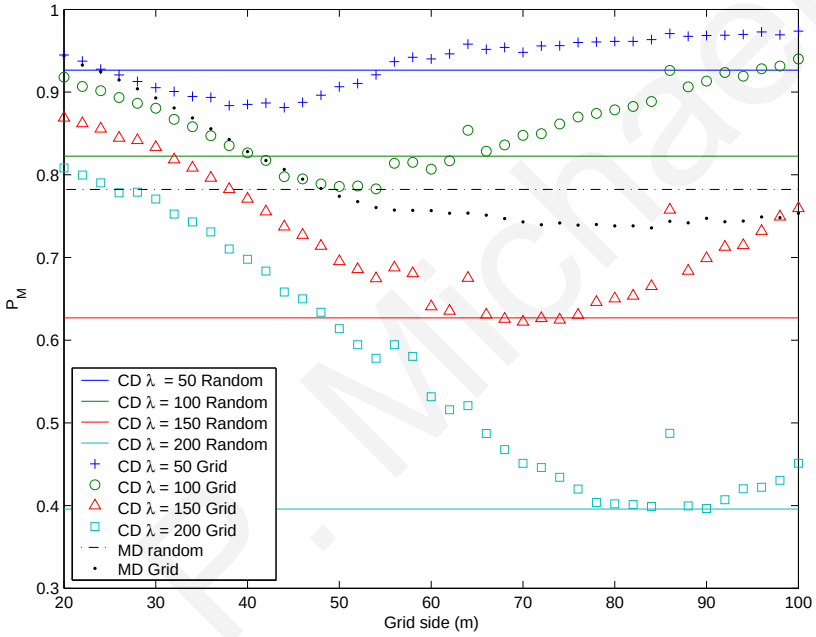
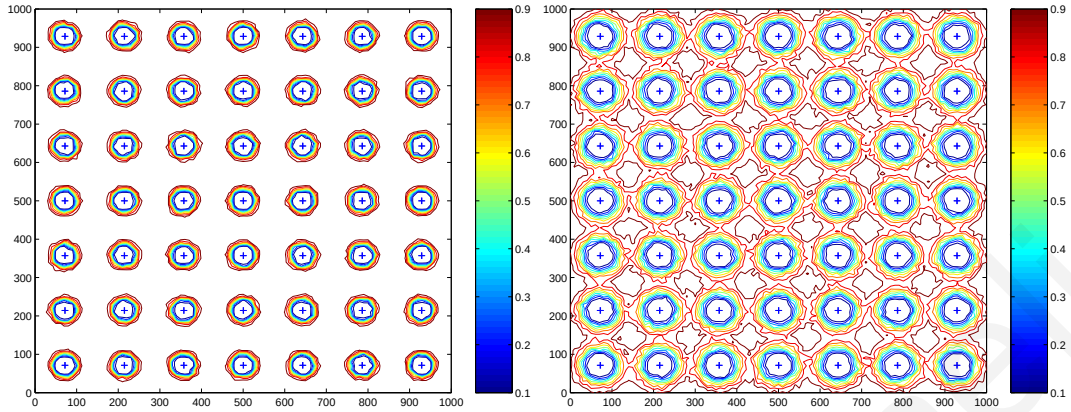


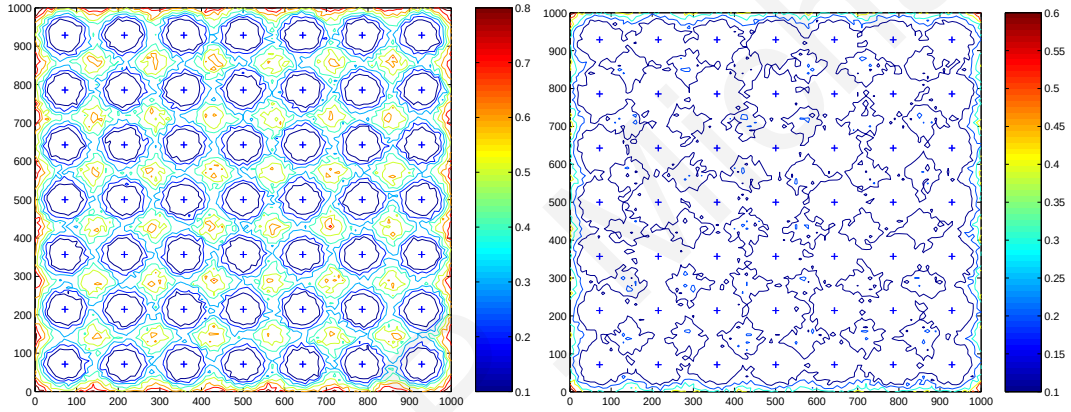
Figure 6.5: Grid vs. random topology.

6.5 Summary

In this chapter we evaluate the performance of three algorithms for event detection in a WSN when the fusion rule at the base station is of the general form "event is detected if *at least* K detection reports from alarmed sensors are received". Firstly, the Mean Detector (MD) where the test statistic is the sample mean of each sensor node by itself. Secondly, the Covariance Detector (CD) that evaluates the sample covariance between pairs of neighboring sensor nodes. If the estimated sample covariance is above a threshold, then the CD reports that an event is present. Finally, the Hybrid Detector (HD) where each sensor decides in-



(a) $\lambda = 50, P_M = 0.85$ (b) $\lambda = 100, P_M = 0.60$



(c) $\lambda = 150, P_M = 0.28$ (d) $\lambda = 200, P_M = 0.09$

Figure 6.6: Probability of miss for CD using grid of sensor node pairs.

dependently between the MD and the CD based on the distance from its *closest* neighbor. Our results indicate that in situations of densely deployed sensor networks with high spatial correlation the CD can drastically improve the coverage (detection performance) of the sensor network compared to the MD. Furthermore the HD can always achieve the best detection results when using an optimal distance threshold to choose between the MD and the CD. In situations where the event distribution is completely unknown, the optimal distance threshold can be derived based on the density of sensor nodes in the field. Finally, our results have implications on the topology of the network as well.

A Proof of Lemma 6.3.2

We can think of a sensor node being alarmed as a red ball and a sensor being non-alarmed as a blue ball. Then, this problem becomes equivalent to the problem of having two containers of red and blue balls and we want the probability of picking a number of red balls from both containers. Suppose that in the first container C_1 we have N_1 balls and in the second container C_2 we have N_2 balls. The probability of a ball being red in the first container is P_{R_1} and in the second P_{R_2}. The probability of choosing i red balls from both containers is given by the following equation:

$$\Pr\{i \text{ red balls}\} = \sum_{x=0}^{i} \binom{N_1}{x} P_{R_1}^{x}[1 - P_{R_1}]^{N_1 - x} \binom{N_2}{i-x} P_{R_2}^{i-x}[1 - P_{R_2}]^{N_2 - i + x} \tag{6.12}$$

To find the total probability of choosing i red balls from both containers we condition on the event of choosing x red balls from C_1 and $i - x$ red balls from C_2 for $x = 0, ..., i$. The probability of choosing x red balls from the first container is given by $\binom{N_1}{x} P_{R_1}^{x}[1 - P_{R_1}]^{N_1 - x}$ and similarly the probability of choosing $i - x$ red balls from the second container by $\binom{N_2}{i-x} P_{R_2}^{i-x}[1 - P_{R_2}]^{N_2 - i + x}$. Since the two events are independent, we multiply them together to get the joint probability of choosing x red balls from the first container and $i - x$ red balls from the second container.

$$\begin{aligned}
\Pr\{i \text{ red balls}\} &= \sum_{x=0}^{i} \Pr\{x \text{ red balls from } C_1 \text{ AND } i - x \text{ red balls from } C_2\} \\
&= \sum_{x=0}^{i} \Pr\{x \text{ red balls from } C_1\} \Pr\{i - x \text{ red balls from } C_2\} \\
&= \sum_{x=0}^{i} \binom{N_1}{x} P_{R_1}^{x}[1 - P_{R_1}]^{N_1 - x} \binom{N_2}{i-x} P_{R_2}^{i-x}[1 - P_{R_2}]^{N_2 - i + x}
\end{aligned}$$

This completes the proof. ∎

B Proof of Lemma 6.3.3

For the MD, the test statistic (6.1) under the H_1 hypothesis has a Gaussian distribution with mean $\mu_{1_{\text{MD}}}(r) = \frac{c}{r^2}$. So using Def. 5.4.1 from Chapter 5, the coverage area becomes a disc around the sensor node location whose radius depends on the event amplitude and detection threshold and is given by

$$R_c = \sqrt{\frac{c}{T_m}} \tag{6.13}$$

where $c \gg T_m$.

For the CD, the test statistic (6.4) has a Gaussian distribution with mean $\mu_{1_{\text{CD}}}(r, d) = \sigma_S^2 e^{-\frac{r+d}{\lambda}}$. Using the analysis of Chapter 5, the perimeter of the coverage area by the two sensors is an ellipse with the sensor node and its closest neighbor located at the ellipse's foci. Therefore the area covered by a sensor that uses the CD can be described by,

$$A_c(\Lambda, d) = \pi \left(\Lambda - \frac{d}{2}\right) \sqrt{\Lambda(\Lambda - d)}. \tag{6.14}$$

where $\Lambda = \frac{\lambda}{2} \ln \frac{\sigma_S^2}{T_c}$.

A sensor will use the CD as long as the coverage area of the ellipse (6.14) is greater than the coverage area of the MD -a circle with radius given by (6.13) , i.e.,

$$A_c(\Lambda, d) = \pi \left(\Lambda - \frac{d}{2}\right) \sqrt{\Lambda(\Lambda - d)} \geq \pi R_c^2 = \frac{\pi c}{T_m}.$$

The proof follows by rearranging terms. ∎

C Proof of Lemma 6.3.4

We start by deriving the cdf of D, $F_D(u)$ as a function of the radial distance u from a given sensor node position.

$$\begin{aligned} F_D(u) &= \Pr\{D \leq u\} = 1 - \Pr\{D > u\} \\ &= 1 - \Pr\{\text{no sensor within disc of radius } u \text{ around sensor node location}\} \\ &= 1 - \Pr\{N - 1 \text{ remaining sensor nodes fall outside disc}\} \\ &= 1 - \left(1 - \frac{\pi u^2}{A}\right)^{N-1} \end{aligned}$$

Note that for $1 - F_D(u) > 0$ we have to have $\pi u^2 \leq A \Rightarrow u \leq \sqrt{\frac{A}{\pi}}$. The pdf of D, $f_D(u)$, is given by

$$f_D(u) = \frac{dF}{du} = \frac{2\pi(N-1)u}{A}\left(1 - \frac{\pi u^2}{A}\right)^{N-2}. \tag{6.15}$$

■

CHAPTER 7

EVENT LOCALIZATION USING NONLINEAR LEAST SQUARES

7.1 Overview

In this chapter, we begin our investigation of using a Wireless Sensor Network (WSN) for estimating the location of an event source. More specifically, we use nonlinear Least Squares (LS) optimization to estimate the source position based on the concentration readings at the sensor nodes. Our simulation results indicate that the LS method performs significantly better than the Closest Point Approach (CPA) where the source location is assumed to be the sensor node with the highest measurement. Furthermore, our results indicate that in the presence of a draft that pushes the substance in certain direction, a threshold is necessary for the LS method to yield accurate results. Finally, the use of unconstrained optimization or the existing knowledge of the wind direction can further improve the location estimate.

7.2 Introduction

This chapter proposes the use of a sensor network for estimating the location of a source that releases a certain substance in the environment which is then propagated over a large area. The concentration of the substance at the source location is assumed unknown. The sensor nodes are able to measure the substance concentration at their own locations but their measurements are noisy. Based on these concentration readings we use nonlinear Least

Squares (LS) optimization [66] to estimate the event source position.

We first present a simulation study for determining the event source location when the substance spreads uniformly in all directions around the source. In our simulations, we vary the number of sensors in the sensor field and the variance of the noise at each measurement. We show that in high uncertainty environments it pays off to use a large number of sensors in the estimation whereas in low uncertainty scenarios a few sensors achieve satisfactory results. We also show the importance of choosing the appropriate parameters for the least squares optimization especially the start position for our algorithm. We compare our results to the Closest Point Approach (CPA) where the source location is assumed to be the sensor node with the highest measurement.

Next, we use the directed propagation model that pushes the substance to a specific direction. Under this scenario, we show that for the Least Squares (LS) approach to yield accurate results, it is necessary to introduce a threshold, such that only measurements that are above this threshold are used for the event source location estimation. Moreover, our results indicate that using unconstrained optimization or our knowledge of the wind direction in constraining the search area of the LS optimization can further improve our estimation results.

The chapter is organized as follows. First, in Section 7.3, we provide the details of the nonlinear least squares approach. Then, in Section 7.4, we analyze the results of the simulation study using the uniform propagation model. Section 7.5 presents the results of the simulation study using the directed propagation model. We conclude with a summary of the main results of this chapter in Section 7.6.

7.3 Nonlinear Least Squares Optimization

The sensor nodes are assumed to be in an energy-conserve state until triggered by the presence of the substance released by the event source. When a sensor node detects the event, i.e., it receives a measurement $z_{i,t} > T$, where T is a threshold, it wakes up and takes a number of discrete measurements M over a time interval and takes the mean of these measurements $\bar{z}_i = \frac{1}{M} \sum_{t=1}^{M} z_{i,t}$. Then, it compares this value to a predefined threshold T to decide whether to communicate this information to the sink and continue measuring or go back to sleep.

Definition 7.3.1. *A sensor i is said to be in the Energized Sensor Set $\mathcal{E}$ if $\bar{z}_i > T$.*

$$\mathcal{E} = \{i \ : \ \bar{z}_i > T\}$$

After the sink receives the information from the sensor nodes $\in \mathcal{E}$, it employs the nonlinear least squares method [66] using the received information. It computes an estimate of the source location $(\hat{x}_s, \hat{y}_s)$ by minimizing the following function:

$$J = \sum_{i \in \mathcal{E}} \left(\frac{\hat{c}}{[(\hat{x}_s - x_i)^2 + (\hat{y}_s - y_i)^2]^{\frac{\alpha}{2}}} - \bar{z}_i \right)^2 \tag{7.1}$$

where $\hat{c}$ is an estimate of the substance concentration at the event source.

Since the initial plume concentration c is also assumed unknown, we first use separable least squares [35] techniques to minimize J with respect to c in terms of the other unknown parameters. This can be accomplished in closed form because J is a quadratic function of c. After taking the partial derivative $\frac{\partial J}{\partial c}$ of (7.1) and set it to zero we obtain:

$$\hat{c} = \frac{\sum_{i \in \mathcal{E}} \frac{\bar{z}_i}{r_i^{\alpha}}}{\sum_{i \in \mathcal{E}} \frac{1}{r_i^{2\alpha}}} \tag{7.2}$$

where r_i is the radial distance from the source, i.e.,

$$r_i = \sqrt{(x_i - x_s)^2 + (y_i - y_s)^2} \tag{7.3}$$

for $i = 1, \cdots, N$. Using the expression for $\hat{c}$ in (7.1) we go back to the usual nonlinear minimization problem with two unknowns, $\hat{x}_s$ and $\hat{y}_s$.

By taking the mean before communicating this information to the sink, we reduce the amount of data flowing in the sensor network and save both bandwidth and energy and therefore prolong the lifetime of the network. Furthermore, as shown in the following lemma, minimizing (7.1) gives exactly the same results as minimizing an objective function that includes all sensor measurements, in other words there is no loss of accuracy in sending only $\bar{z}_i$.

Lemma 7.3.1.

$$\begin{aligned} &\arg\min_{x,y} \sum_{i \in \mathcal{E}} \left(\frac{\hat{c}}{[(\hat{x}_s - x_i)^2 + (\hat{y}_s - y_i)^2]^{\frac{\alpha}{2}}} - \bar{z}_i \right)^2 \\ &= \arg\min_{x,y} \sum_{i \in \mathcal{E}} \sum_{t=1}^{M} \left(\frac{\hat{c}}{[(\hat{x}_s - x_i)^2 + (\hat{y}_s - y_i)^2]^{\frac{\alpha}{2}}} - z_{i,t} \right)^2 \end{aligned}$$

where $\bar{z}_i = \frac{1}{M} \sum_{t=1}^{M} z_{i,t}$.

The proof is included in the appendix.

7.4 Uniform Propagation Simulation Results

For the initial implementation of our sensor network that traces the contaminant transport to source we use the uniform propagation model described in Section 3.1 of Chapter 3 - i.e. the propagation of the contaminant transport is uniform in all directions and there are no environmental changes throughout the propagation.

For all subsequent experiments we used a square sensor field of $1km \times 1km$ and assume that the sensor measurements were given by:

$$z_{i,t} = \min\left\{10^6, \frac{10^6}{r_i^2}\right\} + w_{i,t} \tag{7.4}$$

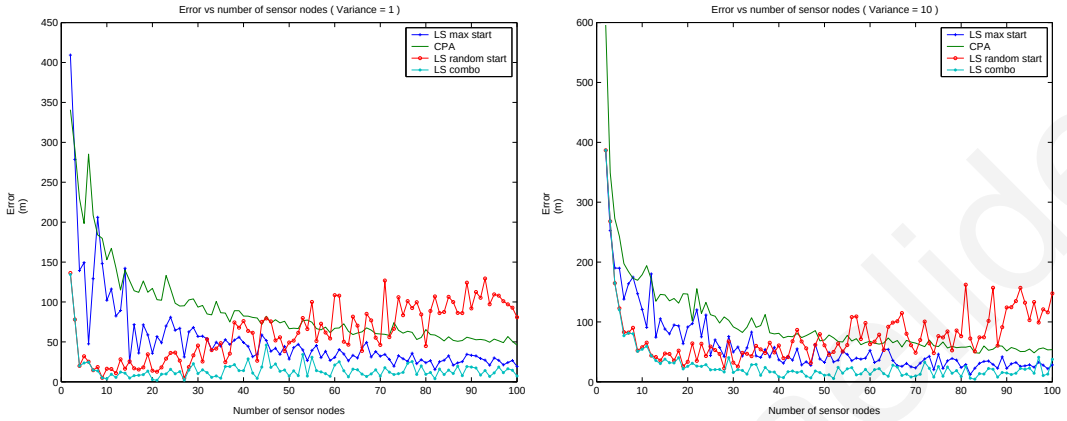
where $i = 1, \cdots, N$, $t = 1, \cdots, M$, $r_i^2 = (x_s - x_i)^2 + (y_s - y_i)^2$ and $w_{i,t} = N(0, \sigma^2)$, $\forall i$, $\forall t$. The error reported is the average over K experiments where we assume that the plume source is randomly placed at points $(x_{s,k}, y_{s,k})$ and we solve the problem (7.1) K-times to obtain $(\hat{x}_{s,k}, \hat{y}_{s,k})$, $k = 1, \cdots, K$. In other words, the error shown in our results is given by

$$Error = \frac{1}{K}\sum_{k=1}^{K}\sqrt{(x_{s,k} - \hat{x}_{s,k})^2 + (y_{s,k} - \hat{y}_{s,k})^2} \tag{7.5}$$

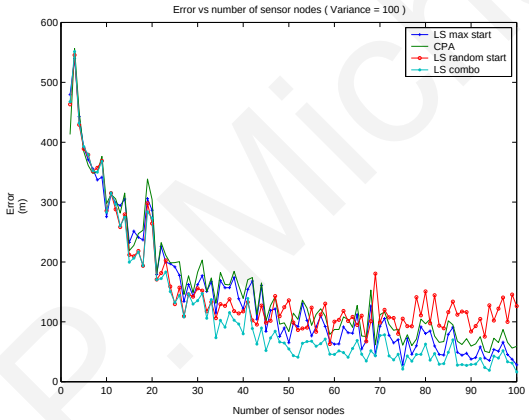
At the beginning of these K experiments we randomly initialize the sensor field but it remains fixed for all K experiments. For the following experiments we used Matlab and $K = 100$.

7.4.1 Effect of varying the number of sensors

In the first set of experiments we investigated the effect of the number of sensors N used in the calculation and compared the results against the CPA. We varied the number of sensor nodes in the sensor field from 2 to 100 and used three different approaches for evaluating the start point of the least squares algorithm. In the first approach, for the starting point, we used a point close to the sensor node with the highest measurement. This plot is denoted as `LS max start`. In the second approach, the initial point is randomly selected. Specifically, we randomly picked 10 different starting positions and chose the one that minimized the squared 2-norm of the residual (`resnorm`) in the least square results. This plot is denoted as `LS random start`. In the third approach we took the better of the two approaches as the one that minimized the overall `resnorm` and we denoted this by `LS combo`. The



(a) $\sigma^2 = 1$ (b) $\sigma^2 = 10$



(c) $\sigma^2 = 100$

Figure 7.1: Error vs number of sensor nodes for different conditions of noise variance.

results are shown in Figures 7.1(a)-7.1(c) for noise variance 1, 10 and 100 respectively. There are several things to observe from these plots:

1. The `LS random start` performs quite well for small numbers of sensor nodes whereas for sensor numbers greater than 50 the `LS max start` performs better. This is expected because the success of the least squares algorithm depends greatly on the starting point of the algorithm and having a starting point close to the actual plume source position enhances the probability of correctly converging to the correct minimum. When there are only a few sensors in the field using the `LS max start` often results in starting points that are far away from the source and so it is a better approach to randomly search for the start position. When the number of sensor nodes increases above a certain threshold this guarantees good coverage of the whole sensor

field and starting at the sensor node with the highest measurement results in starting the optimization in the local neighborhood of the source. In situations like these, `LS random start` performs even worse than the CPA approach. This is due to the large number of sensor nodes which allows several starting positions that are far away from the neighborhood of the source and as a result, the optimization algorithm ends in some local minimum other than the global one. The `LS combo` approach combines the best of both worlds and performs very well for all numbers of sensor nodes so we decided to adopt it for all subsequent experiments and discussions and will simply refer to it as LS.

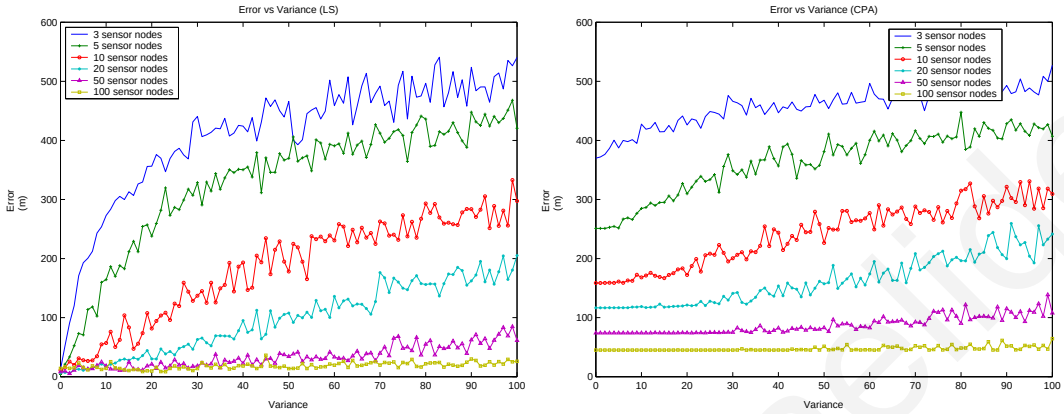
2. Increasing the number of sensor nodes does not always produce better results. For low variance conditions the error decreases until we reach the number of 10 sensor nodes to a value of less than 10m and from there on it follows steady state behavior with unpredictable fluctuations. These are very encouraging results because we want to use as few sensors as possible in the computations to save energy. This way the sensors that are not involved in the sensing or communication process can go to sleep to conserve energy. In higher variance conditions we need more sensor nodes to obtain similar results. For variance 10 we need 25 sensor nodes and for variance 100 we need all 100 sensors.

3. The least squares approach performs better than the CPA for all numbers of sensor nodes. The benefits are more obvious for a few sensor nodes where there is as much as 1500 percent more error in using the CPA. The difference between CPA and LS gradually decreases with increasing numbers of sensor nodes This is expected because increasing the number of sensors allows for a better coverage of the sensor field.

7.4.2 Effect of varying the noise variance

In the second set of experiments we investigated the effect of variance for six different sensor networks with 3, 5, 10, 20, 50 and 100 sensor nodes. The results are shown in Figure 7.2(a) for the least squares approach and Figure 7.2(b) for the closest point approach. From these results we make the following observations:

- The 100 sensor network is the only one that is robust to the noise variance. For the rest of the sensor networks the error increases with variance increase and for high variances it asymptotically approaches the CPA curves. The least squares approach



(a) LS (b) CPA

Figure 7.2: Error vs noise variance for different numbers of sensor nodes.

performs better than the CPA for all variance conditions. The benefits are more obvious for low noise variance conditions but seem to diminish for variance 100. At this point, it is worth pointing out that variance 100, implies that for a significant portion of the sensors, the measurements are dominated by noise, which explains the poor performance of the least squares approach.

- The rate of increase of error with variance increase is dependent on the number of sensor nodes. The 3 sensor node network exhibits the highest rate of increase with the error reaching 50 m for noise variance of just 1. The 5 sensor node network reaches the same error for variance 5, the 10 sensor network for variance 10, the 20 sensor network for variance 30 and the 50 sensor network for variance 75. The 100 sensor network has an error of about 10m for all variance conditions. Based on these results we conclude that for correctly determining the number of sensor nodes to use in the calculation to achieve certain accuracy in the results we need to take this decision based on the noise variance of the propagation model.

7.4.3 Effect of varying the number measurement samples

For all the previous experiments the number of measurements was fixed to $M = 10$. In this set of experiments we investigated the effect of the number of measurements M for six different sensor fields of fixed variance and number of sensor nodes. These were chosen from the previous sets of experiments as the threshold values for which the error was around

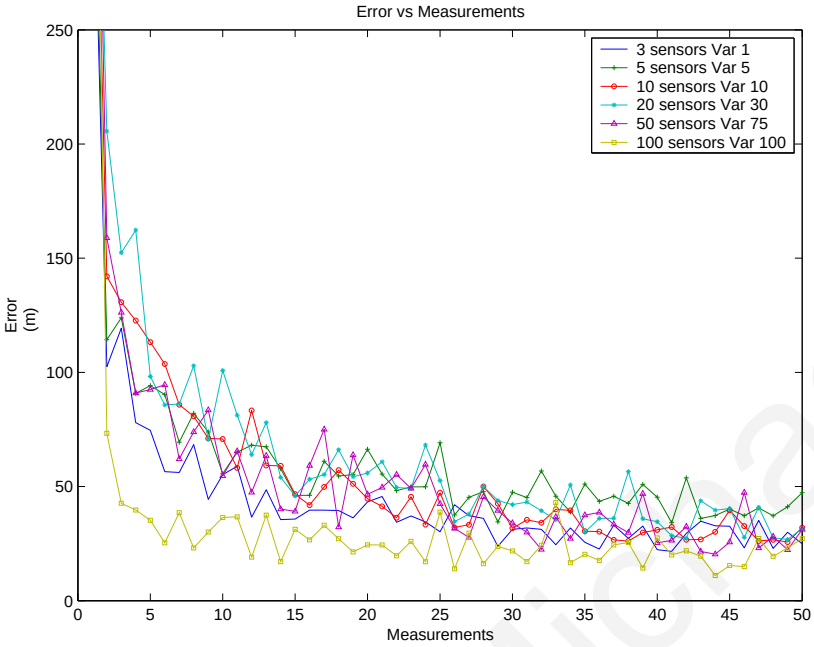


Figure 7.3: Error vs number of measurements.

50m. This time the number of measurements was varied from 1 to 100 and part of the results are displayed in Figure 7.3. From the plot we see that all six of the sensor networks follow a similar behavior. The error decreases exponentially as the number of measurements increases until a certain threshold value of measurements. After this threshold there is no real benefit for taking more measurements as the plots reach a steady state behavior. For the 100 sensor network this threshold is 5 measurements and for the rest of the sensor networks it is around 25 measurements. It is worthwhile to observe the sudden increase in error if we decide to take fewer than 10 measurements.

7.5 Directed Propagation Simulation Results

For this section we assume the simplified signal/substance propagation model which takes into consideration environmental conditions like wind or current flow described in Section 3.3 of Chapter 3.

The solution to the LS problem (7.1) greatly depends on the initial point as well as the search area constraints. In Section 7.4, to obtain the solution of the LS method, we used gradient

based techniques and solved the problem eleven times using eleven different initial points. Specifically, we obtained a solution starting from the location of the sensor with the highest measurement as the initial point as well as ten solutions from 10 random initial points in A. At the end, we simply picked the solution with the minimum residual cost. Furthermore, during any iteration, we constrained the solution in the sensor field area A. In this section, we show that further improvements of the estimation accuracy can be obtained with 2 different methods:

1. *Unconstrained Optimization* which means that we allow the solution to go outside the sensor field area A. This approach helps especially for sources that are located close to the boundaries because allowing the search to temporarily step outside the sensor field may lead to a more accurate estimate. Since the most difficult sources to estimate are the ones close to the boundaries (since there may not be enough sensors close by to record high enough measurements) the benefits can be significant.

2. *Constrained optimization based on wind direction* which means that we restrict the solution to be in the target area $\mathcal{T}$. Assuming that each node is also equipped with a wind direction sensor, we can further reduce the search for the event source in the upwind direction based on the location of the sensor with the highest concentration measurement. For this reason we define the *Target Area* $\mathcal{T}$ (see below). This improves our estimator performance since we are now performing minimization over a smaller area, (see Fig. 7.4).

Definition 7.5.1. *A sensor is said to be in the Target Area $\mathcal{T}$ if it is located in the area defined by the location of the sensor with the highest measurement, ϕ_w and ϕ_u (see Fig. 7.4).*

For all subsequent experiments we used a square sensor field of $1km \times 1km$ and assume that the sensor measurements were given by:

$$z_{i,t} = \min\left\{10^6, \frac{10^6}{r_i^2}\right\} + w_{i,t} \tag{7.6}$$

where $i = 1, \cdots, N$, $t = 1, \cdots, M$, $r_i^2 = (x_s - x_i)^2 + (y_s - y_i)^2$. Furthermore, we assume $w_{i,t}$ to be white Gaussian noise $N(0, \sigma_i^2)$. For all sensors in the active area $\mathcal{A}$ we assumed a constant signal to noise ratio (SNR), thus

$$\sigma_i^2 = \frac{c^2}{SNR \times r_i^4}$$

and for all other sensors $\sigma_i^2 = \frac{1}{SNR}$. The error reported is the average over K experiments where we assume that the plume source is randomly placed at points $(x_{s,k}, y_{s,k}) \in A$ and we

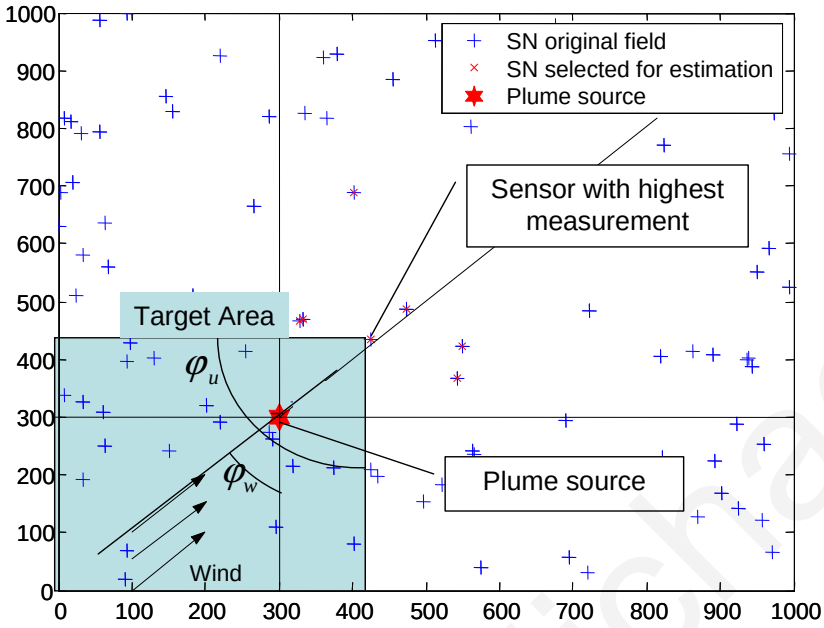


Figure 7.4: Target Area in a field with 100 randomly placed sensor nodes.

solve the problem (7.1) K-times to obtain $(\hat{x}_{s,k}, \hat{y}_{s,k})$, $k = 1, \cdots, K$. In other words, the error shown in our results is given by

$$Error = \frac{1}{K} \sum_{k=1}^{K} \sqrt{(x_{s,k} - \hat{x}_{s,k})^2 + (y_{s,k} - \hat{y}_{s,k})^2} \tag{7.7}$$

At the beginning of these K experiments we randomly initialize the sensor field but it remains fixed for all K experiments. A typical $1km \times 1km$ sensor field with 100 sensors is shown in Fig. 7.4. For the following experiments we use the Matlab package and assume $K = 1000$. We also use $M = 10$ (number of measurements before averaging). Finally we assume a constant north-eastern draft ($\phi_w = 45°$) and a spread angle $\phi_s = 90°$ (see Section 3.3).

For a sensor field with $N = 100$ sensors Fig. 7.5 shows the number of energized sensors (i.e., cardinality $|\mathcal{E}|$) for different values of the SNR. An important observation is that for this fairly dense field (100 nodes per square km) there are about 3.5% of event sources that remain undetected i.e., $|\mathcal{E}| = 0$. Furthermore, there are additional 7% where $|\mathcal{E}| < 3$ which means that the LS localization will not work (it needs at least 3 measurements). For these cases one needs to resort to the Closest Point Approach (CPA).

Fig. 7.6(a) shows the effect of varying the number of samples M that each sensor collects

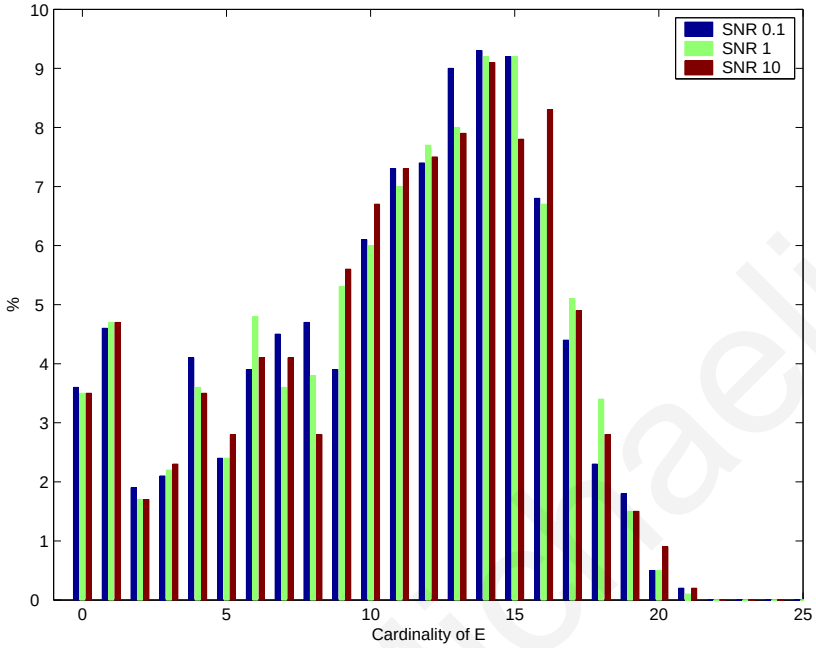


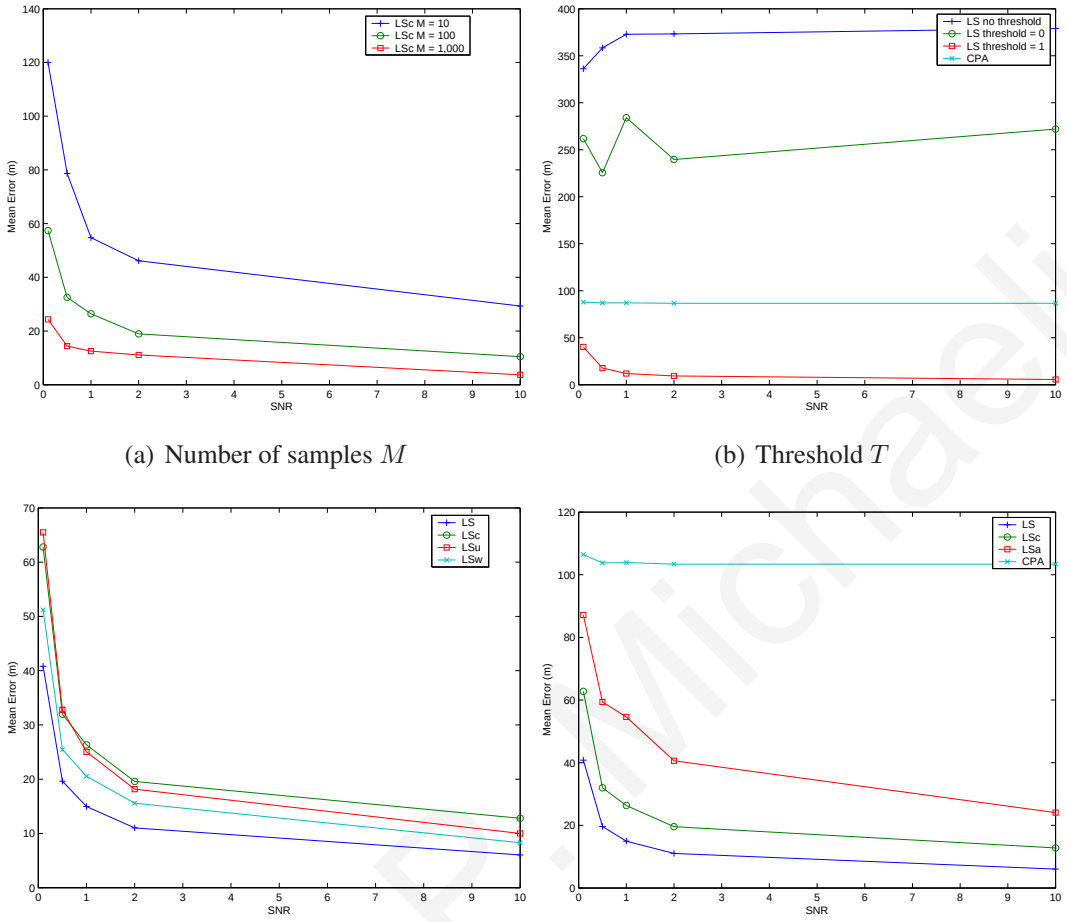
Figure 7.5: Percentage of times that $\mathcal{E}$ has a specific number of sensor nodes given a field with $N = 100$ nodes.

before evaluating the average measurement. As expected, more measurements imply better results particularly for low SNR. Of course, the trade off is that more energy will be needed to collect more data and more delay.

Fig. 7.6(b) shows that it is necessary to apply a threshold on the average measurements of each sensor before using the LS estimation approach. As indicated in the figure, if no threshold is applied (i.e., all sensor measurements are used), then the error is of the order of 370m. Using a threshold of zero improves the error to about 250m but it is still worse than the CPA. Applying a threshold of 1, reduces the error below 5m.

Fig. 7.6(c) compares the estimation accuracy when the event source position is constrained into different regions. LSc refers to a scenario where c is unknown and the source is constrained in the entire sensor field with area A. LSu refers to the same scenario but no constraints are used when the LS problem is solved. LSw refers to the same scenario but now the source position is constrained in the target area $\mathcal{T}$. Finally, the last curve LS refers to the case where c is known. The figure shows that knowledge of the wind direction can improve the estimator accuracy.

Finally, Fig. 7.6(d) compares the LS and CPA approaches when the propagation model is not



(a) Number of samples M

(b) Threshold T

(c) Search region

(d) Model uncertainty

Figure 7.6: Performance of the proposed Least Squares optimization techniques as we vary different parameters in the field.

known exactly, i.e., when the constant α is not known. For all LS approaches shown in the figure, the source in constrained in A. LS refers to the case where both c and α are known. LSc refers to the case where c is unknown but α is known. LSa refers to the case where both c and $1 \leq \alpha \leq 3$ are unknown. Note that in all cases, the LS gives much better results than the CPA.

7.6 Summary

In this chapter we propose sensor network that estimates the event source location in a constrained sensor field using nonlinear LS optimization techniques. First, we use a uniform propagation model in order to investigate the different parameters involved. Our results indicate that in situations with high noise variance it is necessary to increase the number of sensors or the number of measurements at each sensor node to achieve satisfactory results. Next, we use a directed propagation model that pushes the signal to a certain direction. The proposed LS algorithm performs better than the CPA (Closest Point Approach). Furthermore, we show that unlike the uniform propagation model, the performance of the LS now depends on the threshold T. On the other hand, T increases the probability that a source remains undetected. Determining an appropriate threshold is an optimization problem investigated in Chapter 4.

A Proof of Lemma 7.3.1

Let

$$g_i(x_s, y_s) = \frac{\hat{c}}{[x_i - {x_s}^2 + y_i - {y^s}^2]^{\frac{\alpha}{2}}}$$

for all $i = 1, \cdots, N$, where $\hat{c}$ is given by (7.2). Then, the objective function in the left hand side can be written as

$$J_1 = \sum_{i \in \mathcal{E}} \left(g_i(x, y) - \bar{z}_i\right)^2$$

Similarly the objective function in the right hand side can be written as

$$J_2 = \sum_{i \in \mathcal{E}} \sum_{t=1}^{M} \left(g_i(x, y) - z_{i,t}\right)^2$$

To find the extrema of the two functions we simply differentiate with respect to either x or y and set the result equal to zero. Differentiating J_1 with respect to x and setting the derivative equal to zero we get

$$\frac{\partial J_1}{\partial x} = \sum_{i \in \mathcal{E}} \frac{2 \partial g_i(x, y)}{\partial x} \left(g_i(x, y) - \bar{z}_i\right) = 0 \tag{7.8}$$

Differentiating J_2 with respect to x and setting the result to zero we get

$$\begin{aligned}
\frac{\partial J_2}{\partial x} &= \sum_{i\in\mathcal{E}}\sum_{t=1}^{M}\frac{2\partial g_i(x,y)}{\partial x}\left(g_i(x,y)-z_{i,t}\right)\\
&= \sum_{i\in\mathcal{E}}\frac{2\partial g_i(x,y)}{\partial x}\sum_{t=1}^{M}\left(g_i(x,y)-z_{i,t}\right)\\
&= \sum_{i\in\mathcal{E}}\frac{2\partial g_i(x,y)}{\partial x}\left(Mg_i(x,y)-M\bar{z}_i\right)\\
&= M\sum_{i\in\mathcal{E}}\frac{2\partial g_i(x,y)}{\partial x}\left(g_i(x,y)-\bar{z}_i\right)=0
\end{aligned}$$

Note that the last equation has precisely the same solution as (7.8). Similar equations are obtained when differentiating with respect to y, thus the lemma is proved. ∎

CHAPTER 8

THE SNAP ALGORITHM

8.1 Overview

This chapter continues to investigate the use of Wireless Sensor Networks (WSNs) for estimating the location of an event. In this context, we assume that the sensors make binary observations and report the event (positive observations) if the measured signal at their location is above a threshold; otherwise they remain silent (negative observations). Based on the sensor binary beliefs, a likelihood matrix is constructed whose maximum value points to the event location. The main contribution of this chapter is SNAP (Subtract on Negative Add on Positive), an estimation algorithm that provides an efficient way of constructing the likelihood matrix by simply adding ± 1 contributions from the sensor nodes depending on their observation state (positive or negative). Furthermore, we demonstrate how SNAP can be cast in a maximum likelihood estimator with respect to the binary data and can be implemented in a fully distributed fashion (dSNAP). In Chapter 9, we will be showing that SNAP achieves fault tolerant localization when a large percentage of the sensor nodes report erroneous observations.

8.2 Introduction

As already mentioned, a WSN consists of low-cost devices which have limited resources (processing capabilities, memory, and power) and may fail frequently. Note that in this context, it makes sense to use only binary observations; binary decisions are "easier" problems

for the sensors to solve and they also limit the communication cost (single bit transmissions). Moreover, binary decisions are less sensitive to calibration mismatches and varying sensor sensitivities[1]. With this in mind, the objective of this work is to develop a simple, fault tolerant algorithm that can quickly identify the event location using only binary data from the sensor nodes.

SNAP (Subtract on Negative Add on Positive), is one such algorithm and the main contribution of this chapter. The main idea behind SNAP is to use the observations of *all* sensors (positive or negative) to construct a likelihood matrix by summing contributions of ± 1. In other words, sensors with positive observations add their contributions to the cells of the likelihood matrix that correspond to their Region of Coverage (to be defined in the sequel), while sensors with negative observations subtract their contributions. By bounding the contribution of each sensor to ± 1, we do not allow any sensor measurement to dominate the overall estimation result which constitutes the basic reason for the algorithm's fault tolerant behavior (we will be demonstrating the fault tolerance of SNAP in Chapter 9). SNAP is designed to address some of the major challenges outlined in [5] in the new research area of Collaborative Signal Information Processing (CSIP): it promotes the collaborative efforts of the sensor nodes and provides a simple, fault-tolerant way to combine their binary data to estimate the event location.

Another contribution of this chapter is the implementation of SNAP in a distributed fashion (dSNAP). In this context, any alarmed node in the vicinity of the event can be elected as leader and perform the localization of the event by utilizing data from only the sensor nodes inside its neighborhood. The complexity of communications and computations grows rapidly with the number of sensors which makes centralized estimation algorithms impractical and distributed algorithms a more appropriate choice in the context of sensor networks. For an estimation algorithm to be *truly distributed* it has to have two important characteristics. First, an accurate enough estimate of the event characteristics should be obtained without contacting all the sensor nodes in the field. This is essential in terms of both energy efficiency and scalability and it allows sensor nodes irrelevant to the estimation to perform other tasks. Second, the algorithm needs to be simple enough so it can be performed by any sensor node. This way, there is no need to communicate measurements first to a central unit with enough power to carry on complex computations. In this chapter, we show that dSNAP is a truly distributed algorithm and explain how to set the neighborhood size in order to achieve the performance of the centralized algorithm.

[1] If a sensor has an offset $x > 0$, then *all* of its measurements include this error. On the other hand, if the sensor decision is binary, i.e., it has to decide whether its measurement z is less than a threshold Y, then its decision is correct for most measurements except only for the measurements in the range $Y - x < z < Y$.

The chapter is organized as follows. First, in Section 8.3, we present some definitions. Then, Section 8.4 describes the details of the proposed SNAP algorithm. Section 8.5 shows that SNAP can be cast in a maximum likelihood estimator with respect to the binary data. In Section 8.6, we introduce three variants of the SNAP algorithm. Section 8.7 explains the distributed implementation of SNAP. Section 8.8 presents several simulation results. Finally, this chapter concludes with a summary of the main results of this chapter in Section 8.9.

8.3 Definitions

For the sensor network that localizes an event we use the uniform propagation model described in Section 3.1 of Chapter 3. As previously stated, this propagation model may be accurate for sources that emit sound or electromagnetic waves, but it may not be very accurate for problems where an actual substance is released in the environment (for example in problems where the wind pushes the substance towards some direction). Nevertheless, we point out that extensions of SNAP to other signal propagation models are possible as we will be demonstrating in Chapter 10. Furthermore, we assume that the sensor nodes have been programmed with a common threshold T. Given T for any t, we define,

- *Alarmed Sensor*: Any sensor with $z_{n,t} \geq T$.
- *Non-Alarmed Sensor*: Any sensor with $z_{n,t} < T$.

The threshold T is chosen large enough such that the probability of the sensor being falsely alarmed is small, e.g., less than 0.01. This implies that with high probability the sensor nodes that become alarmed do so as a result of the event signal and not because of noise alone. Finally, we assume the fusion center (sink) has correctly detected the event. Regarding this assumption, readers are referred to Chapter 4, where the detection problem in the context of WSN is addressed.

Next, we make the following definition with respect to the footprint of the event.

Definition 8.3.1. Region of Influence *(ROI) of a source, is the area around the source location inside which a sensor node will be alarmed with high probability,* at least *0.5*[2].

[2]Using 0.5 is a convenient way to define the ROI when dealing with noise that has symmetric pdf (e.g. Gaussian) because the ROI size becomes independent of the noise variance.

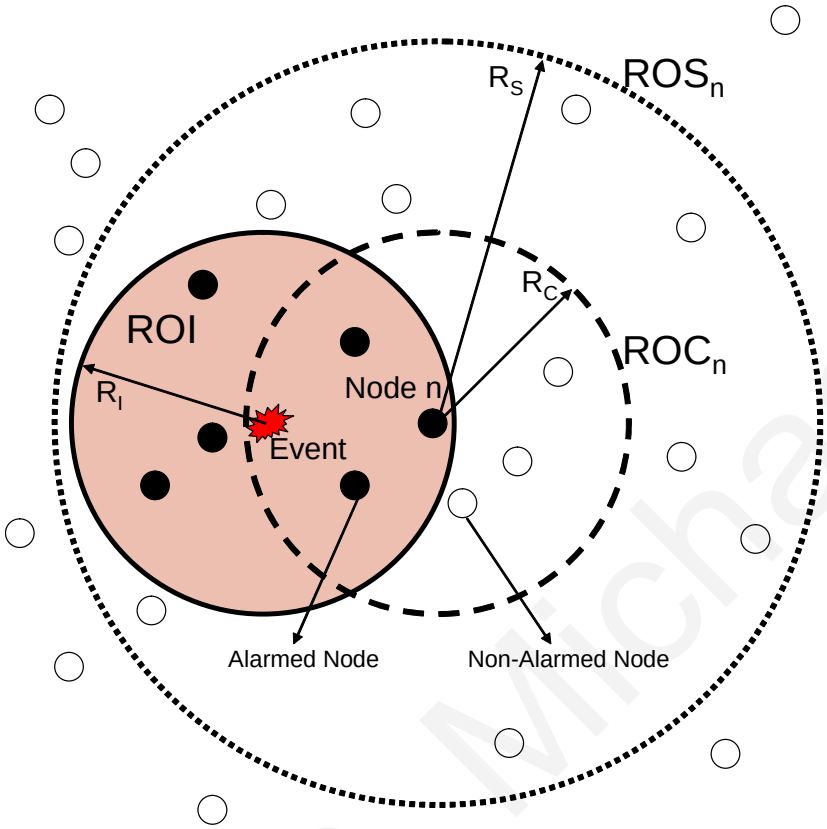


Figure 8.1: Different regions used for SNAP: Region of Influence (ROI), Region of Coverage (ROC_n) and Region of Subscription (ROS_n).

For the uniform propagation model used in this chapter, the ROI becomes a disc centered at the source location as shown in Fig. 8.1 with radius

$$R_I = \sqrt[\alpha]{\frac{c}{T}},$$

derived from (3.3). Since the threshold T is chosen to make the probability of falsely alarmed sensors very small, the ROI can also be interpreted as the area around the event location containing mostly "correctly alarmed" sensors with high probability.

From the sensor node perspective we define two more regions also depicted in Fig. 8.1.

Definition 8.3.2. Region of Coverage of sensor node n *(ROC_n), is the area around a sensor node location inside which if a source is present, then it will be detected with high probability (at least 0.5).*

For the uniform propagation model of Section 3.1 by symmetry, ROC_n becomes a disc centered at the alarmed sensor node location with area equal to that of the ROI.

Definition 8.3.3. Region of Subscription of sensor node n *(ROS_n), is the area around the sensor node n location inside which the sensor node needs to subscribe for information from all other sensor nodes which are relevant to the estimation problem. We also refer to this region as the* neighborhood *of sensor node n.*

This is the same region used in [46] to define the area where a node should notify possible competitors during the leader election process.

8.4 SNAP (Subtract on Negative Add on Positive) Algorithm

The SNAP algorithm consists of four main components:

1. *Grid Formation:* The entire area is divided into a grid $\mathcal{G}$.
2. *Region of Coverage (ROC):* This is the area covered (monitored) by each sensor as per Def. 8.3.2. Given $\mathcal{G}$, the ROC of a sensor is a neighborhood of grid cells around the sensor node location.
3. *Likelihood Matrix Construction:* Associated with $\mathcal{G}$ is a likelihood matrix $\mathcal{L}$. Inside the matrix cells that correspond to the sensors ROC, each sensor adds a value based on its binary observation (+1 on positive observation and -1 on negative). We calculate the likelihood of a source found in each grid cell, by summing the corresponding contributions from all sensor nodes.
4. *Maximization:* The maximum of this likelihood matrix points to the estimated event location.

Next, we describe the algorithm in more detail.

8.4.1 Grid Formation

The area is divided in a grid $\mathcal{G}$ with $G \times G$ cells and grid resolution g (e.g., a 100×100 field with $G = 20$ would have a grid resolution of $g = 5$). Let $C(i, j)$ for $i, j = 1, \cdots, G$,

denote the centers of these cells in a matrix form. The number of cells is a tradeoff between estimation accuracy and complexity that is further investigated in Section 8.8. Each sensor node is associated with a cell (i, j) based on its position (depending on the resolution a cell may contain multiple sensors or no sensors at all). The position index of each node is denoted by (X_n, Y_n), $n = 1, \cdots, N$, where $X_n, Y_n \in \{1, 2, ..., G\}$. Given $\mathcal{G}$ we also define a $G \times G$ likelihood matrix $\mathcal{L}$.

8.4.2 Region of Coverage (ROC)

For the uniform propagation model considered in this chapter we can determine the ROC of a sensor using the following result the proof of which is included in the appendix.

Lemma 8.4.1. *Under conditions of no noise and no faults, assuming (i) a uniform propagation model (as in Section 3.1), and, (ii) the presence of at least one sensor node inside the source ROI, then, the maximum of the likelihood matrix constructed by SNAP is $L_{max} > 0$ and the set of cells of the matrix that attain this value, denoted by A_{max}, always includes the true source location iff $ROC \equiv ROI$.*

Lemma assumption (ii) is rather technical and is only used to exclude cases of events that could not have been detected. The interpretation of Lemma 8.4.1 is that for the uniform propagation model used, SNAP can achieve the highest accuracy when one sets the ROC of the sensor equal to the ROI of the source. Note that for non-uniform propagation models, as in the case of environmental pollution where the wind may push the pollutant in some direction, this result does not hold. In fact, the ROC used in SNAP should be the "mirror image" of the source ROI. This is further investigated in Chapter 10.

8.4.3 Likelihood Matrix $\mathcal{L}$

Each alarmed sensor adds a positive one (+1) contribution to the elements of $\mathcal{L}$ that correspond to the cells inside its ROC. On the other hand, every non-alarmed sensor adds a negative one (-1) contribution to all elements of $\mathcal{L}$ that correspond to its ROC. Thus, the elements of the likelihood matrix are obtained by

$$L(i, j) = \sum_{n=1}^{N} \sum_{t=1}^{M} b_{n,t}(i, j), \text{ for } i, j = 1, \cdots, G, \tag{8.1}$$

where,

$$b_{n,t}(i,j) = \begin{cases} +1, & \text{if } z_{n,t} \geq T \text{ AND } (i,j) \in ROC_n \\ -1, & \text{if } z_{n,t} < T \text{ AND } (i,j) \in ROC_n \\ 0, & \text{otherwise} \end{cases} \tag{8.2}$$

and ROC_n represents the region of coverage of sensor node n.

8.4.4 Maximization

Let (i^*, j^*) be the element of $\mathcal{L}$ with the maximum value, i.e.,

$$L(i^*, j^*) \geq L(i,j) \quad \forall i,j = 1, \cdots, G$$

then the estimated event location is $C(i^*, j^*)$, i.e., the center of the corresponding cell in the grid $\mathcal{G}$. In cases where more than one elements of the $\mathcal{L}$ matrix have the same maximum value, the estimated event position is the centroid of the corresponding cell centers.

8.4.5 SNAP example

To illustrate the SNAP algorithm we provide a simple example using a square ROC. In this example, the ROC of sensor node n is the set of cells that fall in a square of 5×5 cells around cell (i,j) where sensor n is located as shown in Fig. 8.2. For the construction of the likelihood matrix $\mathcal{L}$, each alarmed sensor adds a positive one (+1) contribution to the elements of $\mathcal{L}$ that correspond to the cells inside its ROC as shown in Fig. 8.2. On the other hand, every non-alarmed sensor adds a negative one (-1) contribution to all elements of $\mathcal{L}$ that correspond to its ROC. The resulting likelihood matrix after adding and subtracting the contributions of 8 such sensors using SNAP is shown in Fig. 8.3. The event in this example is correctly localized in the grid cell with the maximum value +3.

8.5 SNAP: A Maximum Likelihood Estimator

In this section we show that SNAP can be cast in a maximum likelihood estimator with respect to the binary data. First, let us define the indicator function for $n = 1, \cdots, N$ and

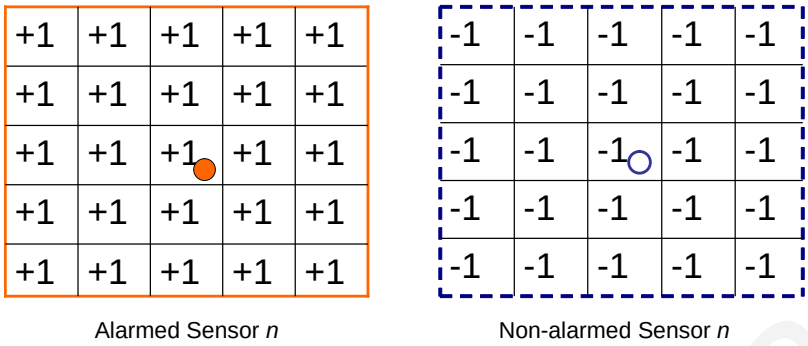


Figure 8.2: Region of Coverage (ROC).

$t = 1, \cdots, M$:

$$I_{n,t} = \begin{cases} 0, & \text{if } z_{n,t} < T \\ 1, & \text{if } z_{n,t} \geq T \end{cases} \tag{8.3}$$

thus, the sensor data can be represented as $\mathbf{I} = \{I_{n,t} : n = 1, \cdots, N, t = 1, \cdots, M\}$. The goal is to estimate the source location $\theta = [x_s, y_s]$ using the collected data $\mathbf{I}$.

8.5.1 Maximum Likelihood

The Maximum Likelihood Estimator has the form:

$$\hat{\theta}_{ML} = \max_{\theta} \log p(\mathbf{I} \mid \theta) \tag{8.4}$$

where the log-likelihood function is given by:

$$\begin{aligned} \log p(\mathbf{I} \mid \theta) = \sum_{n=1}^{N} \sum_{t=1}^{M} I_{n,t} \times \log \left[Q\left(\frac{T - s_n(\theta)}{\sigma_w} \right) \right] \\ + (1 - I_{n,t}) \times \log \left[1 - Q\left(\frac{T - s_n(\theta)}{\sigma_w} \right) \right] \end{aligned} \tag{8.5}$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{t^2}{2}} dt$ is the complementary distribution function of the standard Gaussian distribution. Also, $s_n(\theta)$ is the signal that would have been measured by sensor n if the source was at location θ and there was no noise (given in Section 3.1). The derivation details for the case where $\sigma_w^2 = 1, \forall n, \forall t$ can be found in [34].

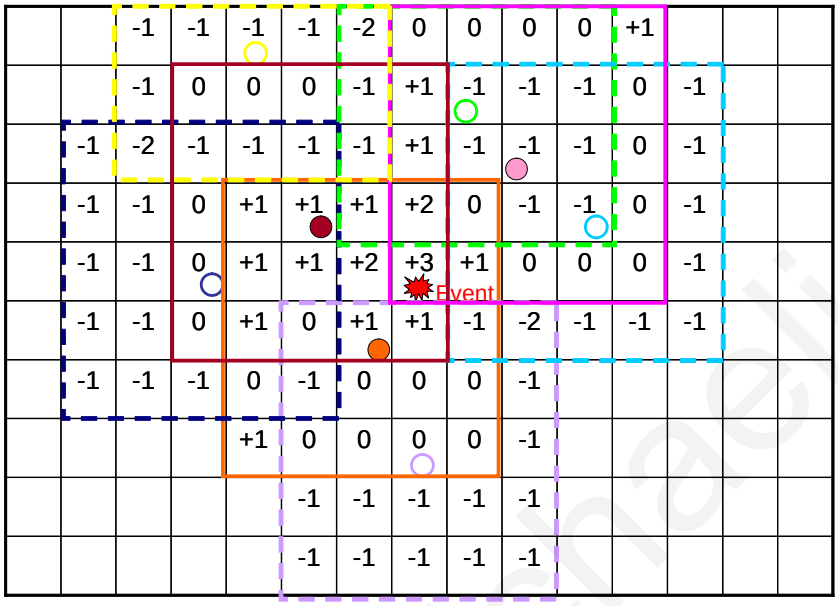


Figure 8.3: $\mathcal{L}$ resulting from SNAP with 8 sensor nodes, 3 of which are alarmed and are shown in solid color. The event is correctly localized in the grid cell with the maximum value +3.

8.5.2 q-ML

In this section we define a particular class of maximum likelihood estimators (q-ML). Given there is a source at a position θ, let us only consider the data from the sensors that fall inside the source neighborhood and let's assume that $0 < K \leq N$ such sensors exist. The data from these sensors can be represented as $\mathbf{I_K} \subseteq \mathbf{I} = \{I_{k,t} : r_k \leq R_c\}$ where r_k is the Euclidean distance between the sensor node $k = 1, \cdots, K$ and the source position θ. The joint likelihood function is given by:

$$p(\mathbf{I_K} \mid \theta) = \prod_{k=1}^{K} \prod_{t=1}^{M} \left[Q\left(\frac{T - s_k(\theta)}{\sigma_w} \right) \right]^{I_{k,t}} \times \left[1 - Q\left(\frac{T - s_k(\theta)}{\sigma_w} \right) \right]^{(1-I_{k,t})}$$

The only difference from (8.5) is that we only use the sensors inside the source neighborhood to construct this likelihood function. For the selected sensors, we propose the following

"arbitrary" probability assignment for their indicator function $I_{k,t}$:

$$\begin{aligned}\Pr\{I_{k,t}=1 \mid \theta\} &= Q\left(\frac{T-s_k(\theta)}{\sigma_w}\right)=q\\ \Pr\{I_{k,t}=0 \mid \theta\} &= 1-Q\left(\frac{T-s_k(\theta)}{\sigma_w}\right)=1-q.\end{aligned}$$

8.5.3 SNAP

Lemma 8.5.1. *SNAP belongs to the class q-ML of maximum likelihood estimators defined in Section 8.5.2 using* $q = 0.99$.

Proof: For SNAP, the source neighborhood becomes the source ROI (defined in Section 8.3). Given there is a source at a position θ, we only consider the data from the sensors that fall inside the ROI to construct this likelihood function. For the selected sensors, it is expected that most will be alarmed, thus we use $q = 0.99$. Next, consider the modified likelihood function $p'(\mathbf{I_K} \mid \theta) = 10^{2KM} p(\mathbf{I_K} \mid \theta)$. Taking the logarithm of the modified likelihood function we get:

$$\begin{aligned}\log p'(\mathbf{I_K} \mid \theta) &= \sum_{k=1}^{K}\sum_{t=1}^{M} I_{k,t}\times\log(9.9)\\ &\quad +(1-I_{k,t})\times\log(0.1)\\ &\approx \sum_{k=1}^{K}\sum_{t=1}^{M} I_{k,t}\times(+1)\\ &\quad +(1-I_{k,t})\times(-1)\end{aligned}$$

Therefore, in order to find the likelihood of each source location θ, we simply add one from all alarmed sensor nodes and subtract one from all non-alarmed sensor nodes that are sufficiently close to θ (i.e., inside the source ROI). This completes the proof. ∎

So the SNAP estimator can be cast in a maximum likelihood estimator with respect to the binary data. Compared to the traditional ML estimator presented in Section 8.5.1, it has two major differences. First, to construct the likelihood function at a location θ it uses only "local" information in the sense that it uses only the data from the sensors that are inside the ROI. Second, and most important, the way the likelihood function is created is very simple and it turns out to be extremely fault tolerant as it will be shown in Chapter 9.

8.6 SNAP Variants

In this section we introduce three variants of the SNAP algorithm. In the first, *SNAPm*, we attempt to conserve energy by first calculating the mean of M measurements *locally* at each sensor node before sending the information to the fusion center (sink). In the second, *SNAPe*, we provide a heuristic for estimating the ROC using *only* the fraction of alarmed sensor nodes in the field. Finally, *AP* is a variant of SNAP that does not consider the non-alarmed sensor nodes in estimating the source location.

8.6.1 SNAPm

Each sensor node n, first computes the mean of M measurements and compares the result to the threshold T to decide whether to become alarmed and communicate its observation to the sink i.e.

- *Alarmed Sensor*: Any sensor with $\overline{z}_n \geq T$.
- *Non-Alarmed Sensor*: Any sensor with $\overline{z}_n < T$.

where $\overline{z}_n = \frac{1}{M}\sum_{t=1}^{M} z_{n,t}$. At the sink, the elements of the likelihood matrix are obtained as before by using (8.1) and (8.2) and dropping the time index t:

$$L(i,j) = \sum_{n=1}^{N} b_n(i,j), \text{ for } i,j = 1,\cdots,G, \tag{8.6}$$

where,

$$b_n(i,j) = \begin{cases} +1, & \text{if } \overline{z}_n \geq T \text{ AND } (i,j) \in ROC_n \\ -1, & \text{if } \overline{z}_n < T \text{ AND } (i,j) \in ROC_n \\ 0, & \text{otherwise} \end{cases} \tag{8.7}$$

SNAPm results in further energy savings compared to SNAP since it requires a *single* message from each sensor node that is alarmed on the "average". In this way, it avoids congesting the network with repeated messages from sensor nodes that have correctly detected the target and eliminates occasional messages from sensor nodes that were having a false alarm. Since communication is the most expensive operation in terms of energy, SNAPm does not only improve the bandwidth utilization but it also prolongs the lifetime of the sensor network. As indicated in Section 8.8, SNAPm achieves similar estimation accuracy as SNAP. On the

other hand, depending on the sampling rate, SNAPm may increase the estimation delay since the sink will have to wait until all M samples are collected.

8.6.2 SNAPe

By Lemma 8.4.1, estimating the ROC is equivalent to estimating the ROI. To calculate the ROI we need to have information about the signal amplitude at the event location c, and the local detection threshold T, which is a function of the noise conditions in the field. In many real-time target tracking scenarios however, the target strength and/or the noise variance may be unknown. Under those circumstances, it is still important to estimate the ROI of a source in order to perform the event localization using SNAP. Thus, in this section we present a possible heuristic that one can use for this estimation, even though it is worth pointing out that other heuristics are also possible. Neglecting boundary conditions, for densely deployed sensor networks, the fraction of the ROI compared to the overall area of the field A should be approximately equal to the fraction of alarmed sensors, in other words we can write

$$\frac{\pi \hat{R}_c^{\,2}}{A} = \frac{N_{alarmed}}{N} \tag{8.8}$$

This result follows from the assumption that the source is uniformly distributed, making all points in area A equally probable [62]. From this equation R_c can be estimated from knowing only the fraction of alarmed sensor nodes in the field. The advantage of SNAPe is that the sink is not required to know a-priory the signal strength c. We point out that this is just a simple heuristic. Another possibility is to obtain an estimate of the diameter of the ROC from the maximum distance $R_{\max}$ between any two alarmed sensors, e.g., set $2 \cdot R_c = \lambda \cdot R_{\max}$, where λ is an appropriate scaling factor that will depend on the density of the network.

8.6.3 Add Positive (AP)

The Add Positive algorithm is similar to SNAP in the sense that it defines a "likelihood matrix" similar to SNAP but in this matrix only positive contributions $(+1)$ from the alarmed sensors inside the ROI of the source are added. The purpose of this algorithm is to demonstrate the value of using the negative information from non-alarmed sensors.

8.7 Distributed Subtract on Negative Add on Positive Algorithm (dSNAP)

In this section, we demonstrate how SNAP can be implemented in a fully distributed fashion (dSNAP). First, a single sensor node is elected leader among the alarmed sensor nodes using one of the leader election algorithms proposed in literature (e.g., [46]). For estimating the event location the elected leader (sensor node l) performs the following steps:

1. Collects information from all relevant sensor nodes inside its Region of Subscription (ROS_l).
2. Constructs a likelihood matrix over a grid $\mathcal{G}_l$ around its location, whose maximum points to the estimated event location.

Next, we provide more details for each of these steps.

8.7.1 Region of Subscription (ROS)

Before estimating the event location the leader node l needs to collect information from sensor nodes located inside its neighborhood (ROS_l). Since energy efficiency is the major consideration in sensor networks and communication is the most expensive operation in terms of energy, we would like to find the smallest ROS_l that achieves the desired objectives in terms of estimation accuracy. Since the leader node l is not aware of the exact location of the event, it needs to make its subscription region big enough to include the event ROI completely. Under conditions of low noise and faults for the uniform propagation model given in Section 3.1, *ROS_l becomes a disc centered at the location of sensor node l with radius twice that of its ROC_l*. This is illustrated in Fig. 8.1 for $l = n$. This way, any correctly alarmed sensor node inside the event ROI, will have in its neighborhood all other correctly alarmed sensor nodes relevant to the estimation problem. This result, is independent of the underlying leader election scheme. However, in scenarios with excessive noise or many sensor faults we run into situations where sensor nodes become alarmed outside the event ROI. In this case, it is better to increase the ROS so that more sensors are involved in the estimation in order to compensate for the effects of noise and faults. We investigate this further in Chapter 9.

Algorithm 1: Likelihood Matrix Construction

Input: $[X_n, Y_n, b_n]$ for sensor nodes $n = 1, ..., N_l \in ROS_l$
Output: Likelihood matrix L_l

1: $L_l \leftarrow 0$
2: **for all** cells $\mathcal{M}^{-1}(i,j) \in \mathcal{G}_l$ **do**
3: **for all** sensor nodes n that have cell $\mathcal{M}^{-1}(i,j) \in \overline{ROC}_n$ **do**
4: $L_l(i,j) \leftarrow L_l(i,j) + b_n$
5: **end for**
6: **end for**

Figure 8.4: Algorithm used by node l to calculate the values of L_l using binary beliefs b_n (± 1) from the nodes inside its ROS_l, with position indices (X_n, Y_n).

8.7.2 Likelihood Matrix L

The elected leader node l, $l \in \{1, \cdots, N\}$ is associated with $\mathcal{G}_l$, a sub-grid of $\mathcal{G}$ with $G_l \times G_l$ cells, centered around its location (X_l, Y_l). The size of the sub-grid $G_l = \left\lfloor \frac{R_s}{g} \right\rfloor + 1$ depends on the size of the ROS_l and the grid resolution g. Furthermore, node l defines a $G_l \times G_l$ Likelihood Matrix L_l where each element (i,j) of L_l corresponds to a cell (u,v) of $\mathcal{G}_l$. This relation is given by a mapping $\mathcal{M} : \mathcal{G}_l \rightarrow L_l$, thus

$$\mathcal{M}\left([u,v]^T\right) = \left[u - X_l + \left\lceil \frac{G_l}{2} \right\rceil, v - Y_l + \left\lceil \frac{G_l}{2} \right\rceil\right]^T$$

where $u, v \in \{1, \cdots, G\}$ and $i, j \in \{1, \cdots, G_l\}$. For every element of L_l, the leader adds the contribution of each sensor that has the corresponding cell in its ROC. The contributions can be ± 1 depending on the sensor's state: $+1$ on alarmed (positive observation) and -1 on non-alarmed (negative observation). In other words, the leader *subtracts on negative observations and adds on positive (SNAP)*. More specifically, the leader updates every element (i,j) of L_l using

$$L_l(i,j) = \sum_{m \in ROS_l} b_m(i,j), \;\; i,j \in \{1, \cdots, G_l\}$$

where

$$b_m(i,j) = \begin{cases} +1 & \text{if } m \text{ alarmed AND } \mathcal{M}^{-1}(i,j) \in \overline{ROC}_m \\ -1 & \text{if } m \text{ non-alarmed AND } \mathcal{M}^{-1}(i,j) \in \overline{ROC}_m \\ 0 & \text{otherwise} \end{cases}$$

and $\overline{ROC}_m$ is the set of all grid cells that are covered by the ROC of sensor m. The maximum of the likelihood matrix points to the estimated position of the source which is taken to be the center of the corresponding cell of $\mathcal{G}$. Let $L_l(i^*, j^*) = \max_{i,j} L_l(i,j)$, then the estimated source position is the center of $\mathcal{M}^{-1}(i^*, j^*)$. If two or more elements have the same maximum value, the estimated position is the centroid of the corresponding cell centers.

Fig. 8.4 demonstrates the algorithm used by the leader node for constructing the likelihood matrix. Fig. 8.5 presents the same test case scenario as in Section 8.4.5 for demonstrating the dSNAP algorithm. For the construction, six other nodes inside the ROS_l are contacted by the leader, two of which are alarmed (depicted with solid color). Nodes inside the gray-shaded area outside the ROS_l are irrelevant to the particular estimation. As in the centralized case, the event is once more correctly in the grid cell with the $L_{max} = +3$ after employing Algorithm 1 at the leader node as shown in Fig. 8.5.

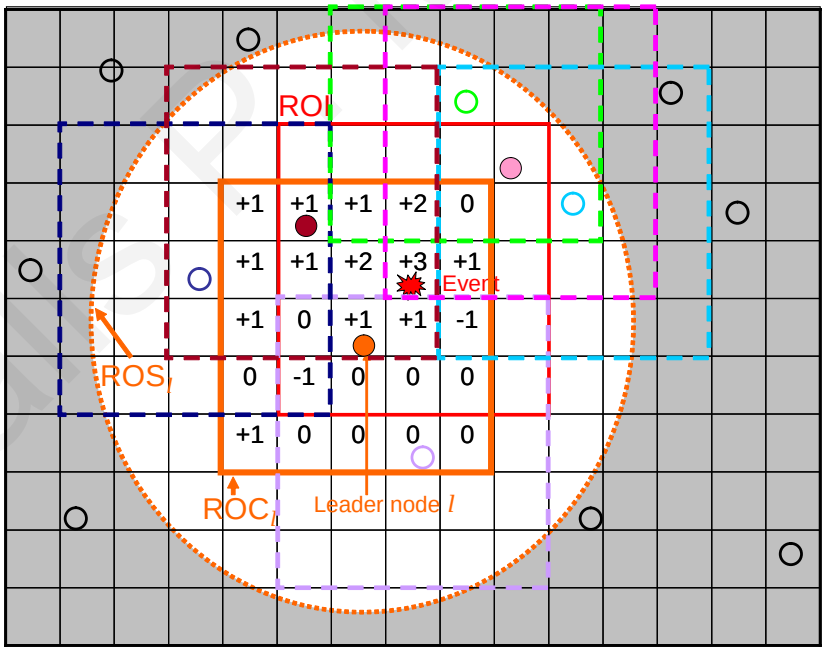


Figure 8.5: dSNAP test case scenario.

8.8 Simulation Results

For all subsequent experiments we use a square 100×100 sensor field where the sensor readings are given by:

$$z_{n,t} = \min\left\{V_{\max}, \frac{c}{r_n^2}\right\} + w_{n,t} \tag{8.9}$$

for $n = 1, \cdots, N$, $t = 1, \cdots, M$. Also for the parameters used in the experiments, we use the default values shown in Table 8.1, unless otherwise stated in the experiment.

Table 8.1: Default parameter values

Parameter	Symbol	Default Value
Number of sensor nodes	N	200
Number of measurements	M	1
Saturation voltage	$V_{\max}$	3000
Source amplitude	c	3000
Noise variance	σ_w^2	1
Threshold	T	5
Grid resolution	g	1

The RMS Error reported, is the average over B experiments where we assume that the source is randomly placed at points $(x_{s,b}, y_{s,b}) \in A$ and we solve the problem B times to obtain $(\hat{x}_{s,b}, \hat{y}_{s,b})$, $b = 1, \cdots, B$ for each estimator considered. In other words, the RMS Error shown in our results is given by,

$$RMS\ Error = \frac{1}{B}\sum_{k=1}^{B}\sqrt{(x_{s,k} - \hat{x}_{s,k})^2 + (y_{s,k} - \hat{y}_{s,k})^2}$$

At the beginning of these B experiments we randomly initialize the sensor field but it remains fixed for all B experiments. For the following experiments we used Matlab and assumed $B = 500$. For the performance evaluation of the SNAP algorithm, we also implement the following two algorithms: Centroid Estimator (CE) and Maximum Likelihood (ML), the details of which can be found in Section 2.6 of Chapter 2.

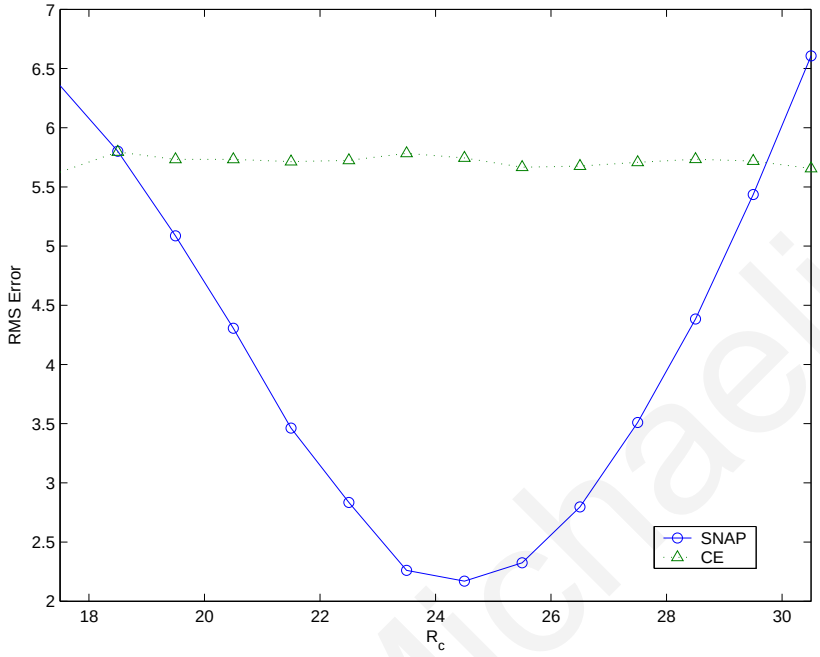


Figure 8.6: Experimental ROC calculation

8.8.1 ROC (Region of Coverage)

In the first set of experiments we investigate the Region of Coverage, which is a main ingredient of the SNAP algorithm. As mentioned earlier, given that $w_{n,t}$ is white noise and using (8.9), the ROC of a sensor is a circular disc with radius

$$R_c = \sqrt{\frac{c}{T}}. \tag{8.10}$$

Thus, we investigate the validity of equation (8.10) when calculating the optimal ROC. In Fig. 8.6 we plot the RMS error vs. R_c. CE performance is also indicated on the same plot for comparison purposes. For the scenario investigated, the optimal value for R_c is 24.5.

Table 8.2, shows the optimal R_c results for different percentages of alarmed sensor nodes. For obtaining the analytical results we used (8.10) and for obtaining the experimental results we varied R_c as in the first experiment and recorded the value that minimized the RMS Error. As seen from the table, for all cases tested, the experimental results for R_c are very much in agreement with the analytical ones.

Table 8.2: Optimal ROC calculation for different percentages of alarmed sensor nodes

c	%Nal	R_c	$\hat{R}_c$	$R_c(exp)$
1000	7	14.1	14.9	14.5
2000	14	20.0	21.1	20.5
3000	21	24.5	25.8	24.5
4000	28	28.3	29.8	28.5
5000	35	31.6	33.4	31.5

(%Nal denotes the percentage of alarmed sensors, R_c and $\hat{R}_c$ are calculated using (8.10) and (8.8) respectively while $R_c(exp)$ is obtained experimentally).

8.8.2 Region of Subscription for dSNAP

In the next set of experiments, we investigate the ROS size for dSNAP. Remember that the ROS defines the neighborhood that the leader node needs to obtain information relevant to the estimation problem. In Section 8.7, we made the claim that in the absence of noise and faults the ROS radius should be set at least twice the ROC radius to obtain accurate estimation results independent of the underlying leader election scheme involved. Fig. 8.7 demonstrates this result with a test case in which the radius of ROC is $R_c = 20$ and we vary the ROS radius R_s from 20 to 60. It is evident that for values $R_s \geq 40$ (i.e. $R_s \geq 2R_c$) dSNAP approximates the performance of the centralized algorithm. Of course with respect to energy efficiency we would like to keep the communication cost low, by keeping the ROS as small as possible while achieving the desired accuracy. Thus, for the case without faults it seems that $R_s = 2R_c$ is a good choice.

8.8.3 Performance Evaluation

In this section, we test the performance of SNAP against the other estimators in terms of accuracy and time complexity. First, Fig. 8.8(a) shows the results for various grid resolutions ($g = 1, 2, 5, 10, 20$). Then, Fig. 8.8(b) shows the results as a function of the threshold T - we varied threshold T from 1 (80% of sensors alarmed) to 20 (5% of the sensors alarmed). Finally, Fig. 8.8(c) shows the results as we decrease the number of alive sensor nodes in the field from 200 to 50 (e.g., due to node failures). In all plots, it is evident that SNAP and ML exhibit a similar performance with the ML being the most accurate. This is attributed partly

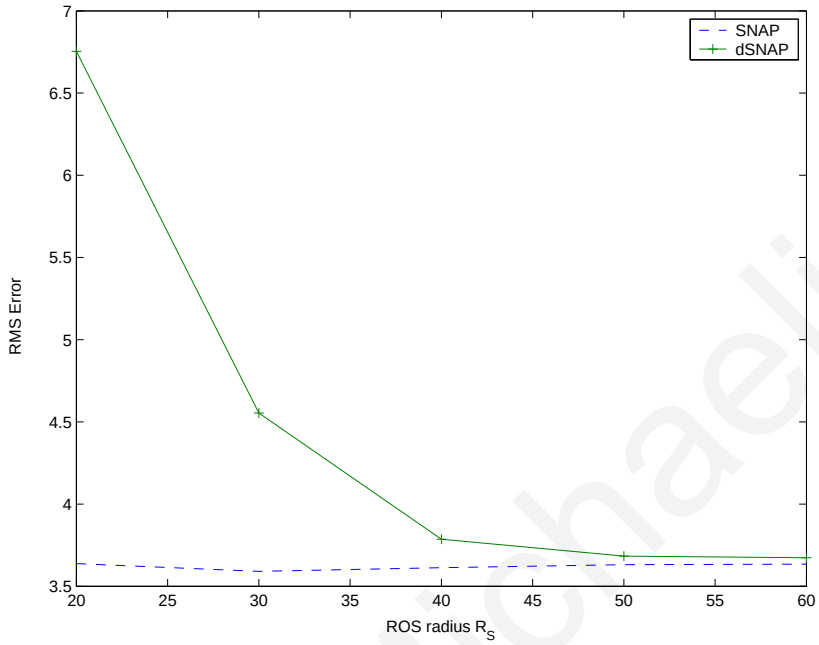


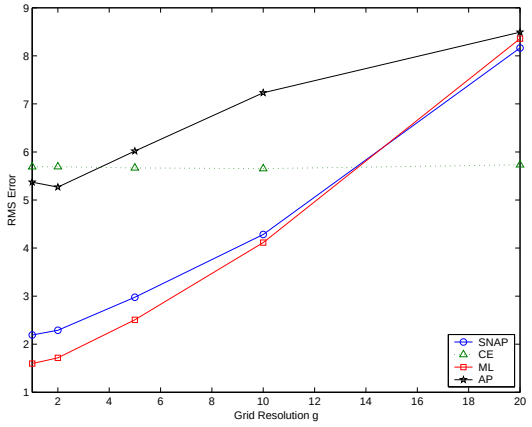
Figure 8.7: Effect of varying ROS radius for dSNAP.

to the fact that ML uses the exact sensor positions while for SNAP the sensor positions are quantized. Also, ML calculates the exact probabilities while SNAP has "arbitrarily" assigned some values. AP has worse performance than SNAP for all grid resolutions and thresholds tested; this is a direct result of not including the non-alarmed sensor nodes in the estimation procedure.

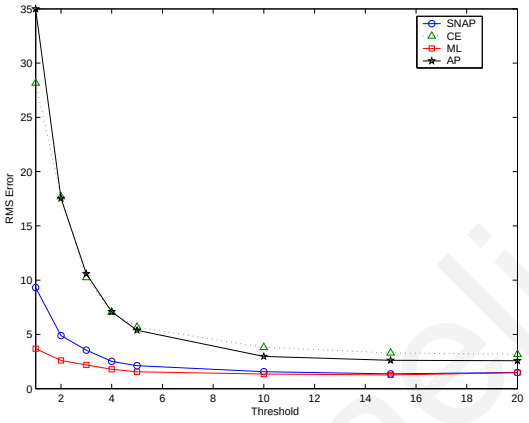
Furthermore, there is a tradeoff between accuracy and complexity of the different algorithms that we need to consider. Some typical results are shown in Table 8.3. The algorithms are run on a regular PC (with Intel Pentium 4 CPU, 3.6GHz, 2GB of RAM). CE completes essentially in zero time and is not shown on the table. SNAP (or AP) only require simple additions to estimate the event position and they complete relatively fast even for the smallest grid resolution tested ($g = 1$). ML on the other hand, requires numerical integration and the time for completion is considerably larger. From Fig. 8.8(a) and Table 8.3 it is evident that one should use the smallest g allowed by the available computational budget.

8.8.4 SNAPm

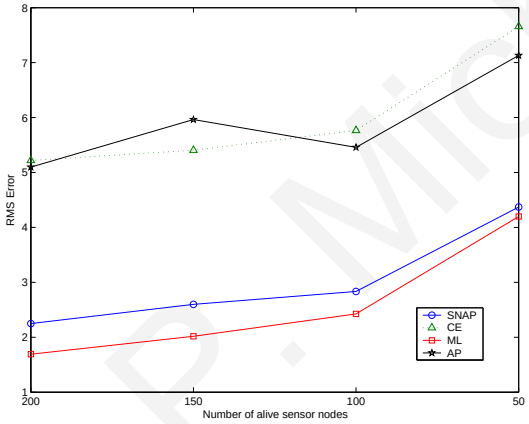
In this section, we investigate SNAPm, the version of SNAP where each sensor node first computes the mean of the M measurements to decide whether to become alarmed. Fig. 8.9



(a) Grid resolution g



(b) Threshold T



(c) Number of alive sensor nodes

Figure 8.8: Estimator performance evaluation for different varying parameters.

shows the results for both algorithms as we vary the noise variance for different values of M. Please note that in the period of getting the M observations, it is assumed that the event source is static and has a fixed amplitude. From the plot, it is evident that both algorithms exhibit a similar performance, with SNAP performing slightly better than SNAPm for smaller noise variance conditions. SNAPm however, is more energy efficient since it requires a single transmission from each sensor node and this benefit increases as we increase M. On the other hand, SNAPm is not expected to be as robust as SNAP when faults occur (e.g., dropped packets) since the sink will have less information to counteract such errors. Thus, when a sensor has a large number of measurements M, a possibility is to group them into h groups with M/h measurements in each group and send up to h packets that will correspond to the event that the sensor is alarmed or not "on the average".

Table 8.3: Complexity analysis in elapsed time (sec)

Algorithm	$g = 1$	$g = 2$	$g = 5$	$g = 10$
SNAP / AP	0.07	0.03	0.02	0.02
ML	2.55	0.66	0.13	0.05

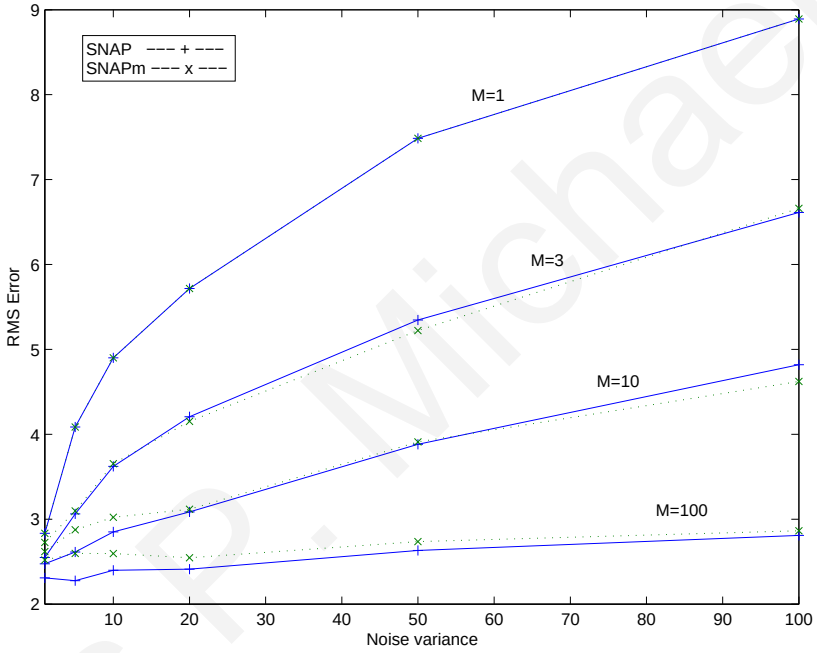


Figure 8.9: SNAP performance for various M, σ_w^2.

8.8.5 SNAPe

Next, we investigate SNAPe, the version of SNAP that does not require knowledge of the source amplitude c or the threshold T. Remember that for SNAPe, R_c is estimated using (8.8) based on the fraction of alarmed sensor nodes.

First, we investigate the validity of (8.8) for computing the optimal R_c for SNAPe. Table 8.2 shows the results for different percentages of alarmed sensor nodes. The percentage of alarmed sensors (%Nal) shown in the table, is the average obtained by varying the event amplitude c and using only non-boundary sources in the simulations. R_c shows an almost linear relationship to the % of alarmed sensor nodes in the field. For all cases tested, (8.8) is very good at estimating the analytical values of R_c obtained by (8.10). The same experiment

was also performed with $N = 100$ with almost identical results. This shows that R_c is not very sensitive to the total number of sensors deployed.

Next, in Fig. 8.10, we investigate the performance of SNAPe. There is an evident loss in performance compared to SNAP - the case when we use the analytical R_c calculated from (8.10). This is due mainly to sources on the boundaries, where the number of alarmed sensor nodes does not produce the correct ROC for estimating R_c. The performance of SNAPe is still better than CE for $g = 1, 2, 5$.

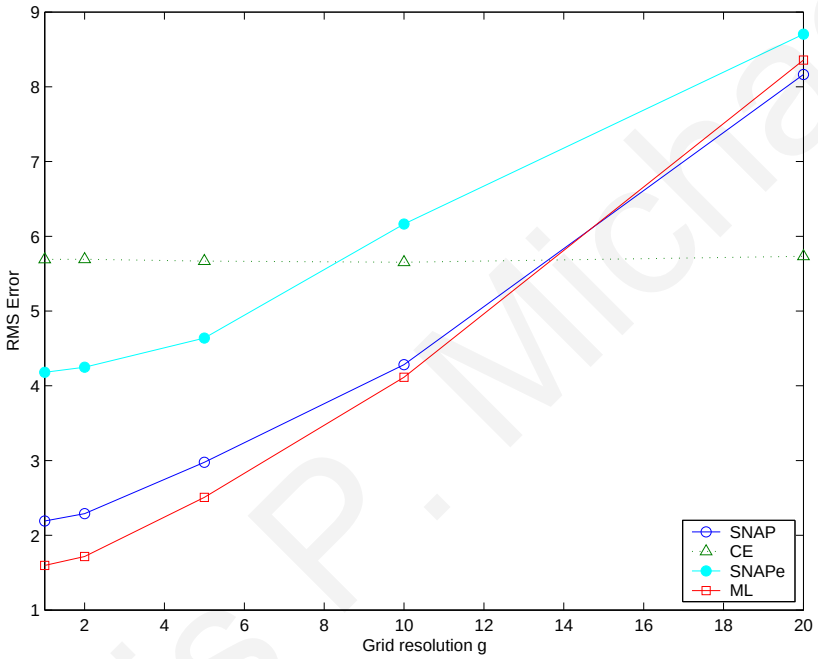


Figure 8.10: SNAPe performance for various grid resolutions g.

8.8.6 Uncertainty in the Position of the Sensor Nodes

In the final set of experiments, we relax Assumption 1 of Chapter 3 by assuming that the position of each sensor node is not known exactly but it is estimated imprecisely. We model the errors in the estimated sensor node positions with normal Gaussian noise $N(0, \sigma_p^2)$. Fig. 8.11 shows the results as we vary σ_p^2 for 4 different values of the source amplitude ($c = 1000, 2000, 3000, 4000$). From the plot, it is evident that SNAP looses very little in accuracy when we have errors in the estimated sensor node positions. This robust behavior to sensor position errors is extremely important for sensor networks, since it is usually impossible to obtain accurate positions for hundreds of nodes using localization algorithms.

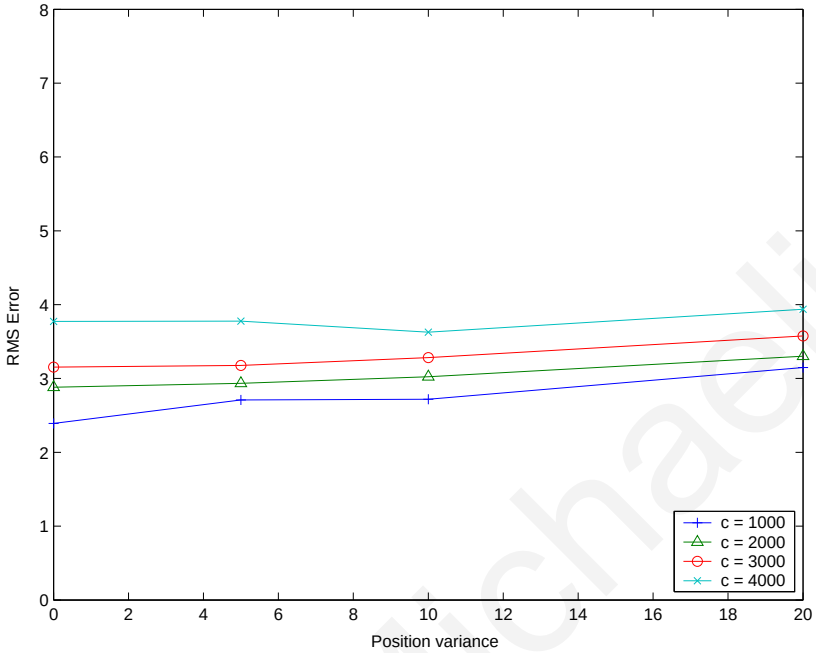


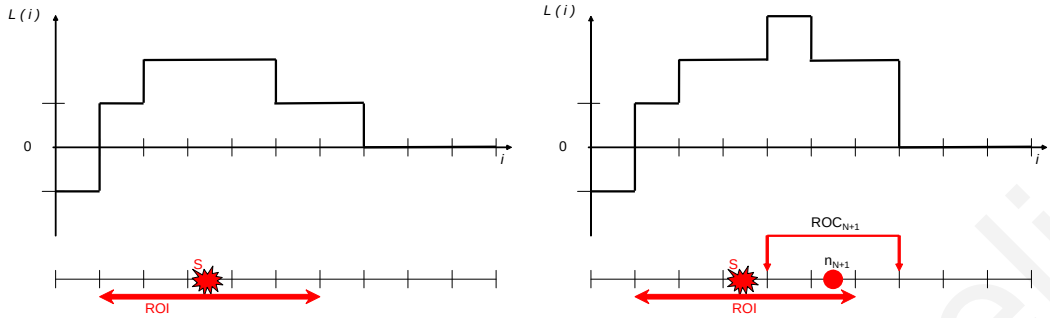
Figure 8.11: SNAP performance for various σ_p^2.

8.9 Summary

SNAP (Subtract on Negative Add on Positive), is a simple and efficient algorithm, that can be applied in time-critical applications for estimating the position of an event given only binary data from the sensor nodes. Compared to maximum likelihood estimation, SNAP is slightly less accurate but is computationally less demanding. Moreover, for the construction of the likelihood matrix we only need "local" information, which is something that was exploited in order to derive a distributed version of SNAP.

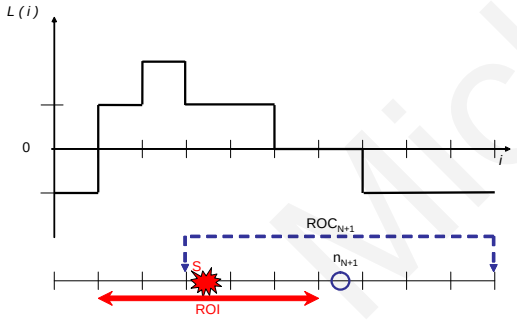
A Proof of Lemma 8.4.1

For the forward direction of the proof we use induction over the number of sensors n. Without loss of generality, assume that the source is placed at point $(0, 0)$ and its ROI is a circle with radius r_{ROI}. For $n = 1$, (single sensor case) by assumption (ii), this has to be placed inside the ROI of the source, and given the ideal conditions, it will be alarmed. Thus, $L_{max} = 1 > 0$ and A_{max} is the ROC of the sensor. Thus, a circular ROC with radius



(a) L function after contribution of n sensors

(b) $r_{ROC} < r_{ROI}$



(c) $r_{ROC} > r_{ROI}$

Figure 8.12: SNAP likelihood matrix along the line between the source and the $n+1$-th sensor.

$r_{ROC} \geq r_{ROI}$ guarantees that $(0,0) \in A_{max}$ and the forward statement of the lemma holds for $n = 1$. Next, let's assume that the lemma statement holds for any sensor field with n sensors. We need to show that is also holds for a field with $n+1$ sensors.

Fig. 8.12(a) shows a typical likelihood function due to n sensors along the line that connects the source with the $n+1$-th sensor. According to the lemma statement, the source position is included in $A_{max}(n)$ (the set of cells with maximum value when n sensors are used). If this sensor is placed inside the ROI of the source, it will be alarmed, and it will add $+1$ to all cells of the likelihood matrix that fall inside its ROC. If $r_{ROC} \geq r_{ROI}$, the ROC of the sensor will also include the true source position and it is guaranteed that the source will remain in the $A_{max}(n+1)$. If the sensor is placed outside the source ROI, then it will be non-alarmed and thus it will add a -1 contribution to all cells in its ROC. Thus, if $r_{ROC} \leq r_{ROI}$, the -1 will not affect the cell with the true source position, thus it is guaranteed that the source will remain in $A_{max}(n+1)$. Thus, to guarantee that the source will always remain in the

$A_{max}(n+1)$ we need $r_{ROC} = r_{ROI}$.

If $r_{ROC} \neq r_{ROI}$, one can easily construct examples where $A_{max}(n+1)$ does not include the true source position. If $r_{ROC} < r_{ROI}$, one can place the $n+1$-th sensor just inside the ROI so it will be alarmed but its ROC will not include $(0,0)$. Thus, it may add $+1$ to some of the cells in $A_{max}(n)$ which may constitute an $A_{max}(n+1)$ that does not include the true source position (see Fig. 8.12(b)). If $r_{ROC} > r_{ROI}$, one can place the $n+1$-th sensor just outside the ROI so it will be non-alarmed but its ROC will include $(0,0)$. By subtracting one from the cell with the true source position, it is possible that some other cells in $A_{max}(n)$ but further away from the $n+1$-th sensor are not affected. As a result, they may constitute an $A_{max}(n+1)$ that once more does not include $(0,0)$, the true source position (see Fig. 8.12(c)).

CHAPTER 9

FAULT TOLERANT MAXIMUM LIKELIHOOD EVENT LOCALIZATION

9.1 Overview

This chapter investigates Wireless Sensor Networks (WSNs) for achieving fault tolerant localization of an event using only binary information from the sensor nodes. In this context, faults occur due to various reasons and are manifested when a node outputs a wrong decision. The main contribution of this chapter is to show that the SNAP algorithm (developed in Chapter 8) retains its accuracy even when a large percentage of the sensor nodes report erroneous observations. SNAP is compared against the Centroid (CE) and the classical Maximum Likelihood (ML) estimators and is shown to be significantly more fault tolerant. Moreover, in this chapter we show how ML can be modified in order to accommodate erroneous observations coming from faulty sensor nodes. Specifically, by incorporating the fault probability when calculating the likelihood function we develop a fault tolerant maximum likelihood (FTML) estimator appropriate for sensor networks applications. Compared to the SNAP algorithm in the presence of faults, the two can achieve similar performance; FTML is slightly more accurate while SNAP is computationally less demanding and requires fewer parameters.

9.2 Introduction

This chapter investigates Wireless Sensor Networks (WSNs) for achieving fault tolerant localization of an event using only binary information from the sensor nodes. The simple nature of sensor nodes makes them extremely vulnerable to faults. These faults can occur for a variety of reasons: noise, energy depletion, environmental harsh conditions of operation, attacks, software problems. These have been reported in real experiments and can result in erroneous, unexpected behavior (Byzantine faults).

For event localization in WSNs using binary data, two different methods have been proposed. The Centroid Estimator (CE) [33] which simply takes the centroid of the positions of all alarmed sensor nodes as the estimated event location. Although sub-optimal this simple method works quite well under conditions of dense sensor deployment and when events are not located close to the field boundaries. The other approach is based on the classical maximum likelihood (ML) estimator which is shown to be optimal in [34] when enough sensor measurements are used. Both of these methods however, can yield significant estimation errors when the sensor nodes start failing; CE is sensitive to false positives, i.e., sensor nodes far away from the source becoming falsely alarmed. On the other hand, ML is extremely sensitive to false negatives, i.e., when a node located close to the event does not become alarmed.

The main contribution of this work is to show that the SNAP algorithm, a special form of the maximum likelihood estimator developed in Chapter 8, can achieve similar accuracy but is significantly more fault tolerant. Furthermore, we demonstrate how classical maximum likelihood estimation can be modified in order to accommodate erroneous observations coming from faulty sensor nodes. Specifically, by incorporating the fault probability when calculating the likelihood function we develop a fault tolerant maximum likelihood (FTML) estimator appropriate for sensor networks applications. SNAP can achieve similar accuracy and fault tolerance to FTML but it is computationally more efficient and makes fewer parameter assumptions.

The chapter is organized as follows. First, we present the fault model in Section 9.3. Then, in Section 9.4, we test the fault tolerance of SNAP, CE and ML binary estimators and propose the FTML estimator. Section 9.5 presents several simulation results. We conclude with a summary of the main results of this chapter in Section 9.6.

9.3 Fault Model

In this chapter, we use the uniform propagation model described in Section 3.1 of Chapter 3. For the fault tolerance analysis, we use a *Fault Model* where each sensor node can exhibit erroneous behavior with probability P_f and in this case its original belief is simply reversed as shown in the example of Fig. 9.1. When applying the fault model, some sensor nodes that fall outside the ROI (defined in Chapter 8) of the source become alarmed as a result of a fault and are shown as *false positives*. Similarly, sensors that fall inside the ROI become non-alarmed as a result of a fault and are shown as *false negatives*.

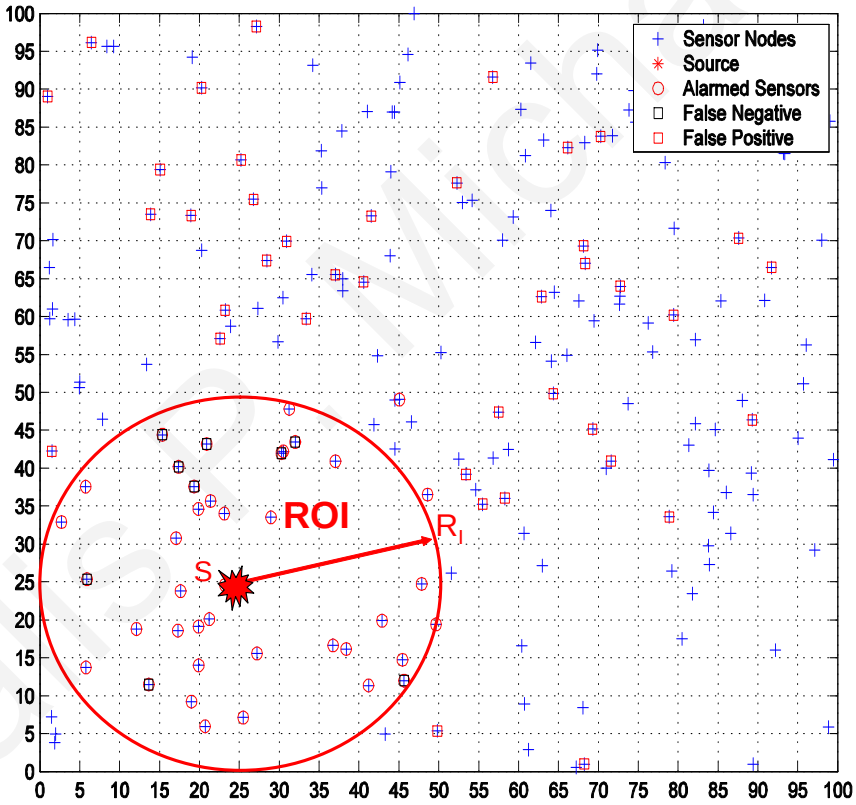


Figure 9.1: A field with 200 randomly deployed sensor nodes and a source placed at position (25,25). Alarmed sensors are indicated on the plot with red circles inside the disc around the source (ROI). 50 of the sensor nodes exhibit faulty behavior and are indicated on the plot as false positives (red squares outside the ROI) and false negatives (black squares inside the ROI).

9.4 Fault Tolerance of Binary Estimators

In this section we investigate three different estimators for localizing the event using only binary data: CE, ML and SNAP. First, we provide the details and analyze the fault tolerance of each of these estimators. Then, in order to gain some further insight into their behavior we present a simple example. Finally, we propose a Fault Tolerant Maximum Likelihood (FTML) estimator and compare its performance with SNAP.

9.4.1 Centroid Estimator (CE)

Recall from Section 2.6 in Chapter 2 that the event location estimated by the Centroid Estimator (CE) is the centroid of the positions of the alarmed sensor nodes given by

$$\hat{\theta}_{CE} = [\hat{x}_s, \hat{y}_s] = \left[\frac{1}{P} \sum_{n=1}^{P} x_n, \frac{1}{P} \sum_{n=1}^{P} y_n \right] \tag{9.1}$$

CE treats all alarmed sensor nodes with equal weight when calculating the centroid of the position of the alarmed sensor node (see (9.1)). This fact, makes the algorithm fairly sensitive to the presence of false positives, especially the ones that occur far away from the true event location. Such faults distort the estimated source position towards the location of the faulty sensor. On the other hand, CE is robust to a large percentage of false negatives. Since it does not consider information from non-alarmed sensor nodes, it can estimate the event location even with a single alarmed sensor node following a "closest point approach".

9.4.2 Maximum Likelihood (ML)

Recall from Section 2.6 in Chapter 2 that the Maximum Likelihood (ML) estimator has the form:

$$\hat{\theta}_{ML} = \max_{\theta} L(\theta)$$

where the log-likelihood function is given by

$$\begin{aligned} L(\theta) &= \log p(\mathbf{I} \mid \theta) \\ &= \sum_{n=1}^{N} \sum_{t=1}^{M} \{ I_{n,t} \log [P_{n,t}] + (1 - I_{n,t}) \log [1 - P_{n,t}] \} \end{aligned} \tag{9.2}$$

where

$$P_{n,t} = \Pr\{z_{n,t} \geq T\} = Q\left(\frac{T - s_n(\theta)}{\sigma_w}\right).$$

The weakness of ML is that it is extremely sensitive to false negatives. Even a single faulty sensor node inside the ROI of the source can completely throw off the estimation results. This is a direct result of the construction of the likelihood matrix of ML using (9.2). Note that for source positions θ close to the sensor, the term $P_{n,t} \to 1$ and as a result $\log[1 - P_{n,t}] \to -\infty$. If for some reason (e.g., due to a fault), a sensor fails to detect an event that is very close to it (false negative), then $(1 - I_{n,t}) = 1$ and as result the faulty sensor has a *very* large negative contribution to the likelihood function at all points near itself which (by assumption) is also the point where the source is located. On the other hand, a healthy sensor near the source contributes only $I_{n,t} \log[P_{n,t}] \to 0$ which is not sufficiently large to counteract the negative contribution of the faulty sensor. In fact, since the negative contribution of the faulty sensor is unbounded, even several well behaving sensors cannot correct the error.

9.4.3 SNAP (Subtract on Negative Add on Positive)

The SNAP algorithm (see Chapter 8 for more details) can be briefly described in 4 steps:

1. *Grid Formation*: The entire area is divided into a grid.
2. *Region of Coverage (ROC)*: For each sensor node we define a neighborhood of cells around the sensor node location that we call the Region of Coverage. Inside the cells of its ROC, each sensor node outputs a value based on its binary status: +1 on alarmed (positive observation) and -1 on non-alarmed (negative observation).
3. *Likelihood Matrix Construction*: For each cell of the area grid we calculate the likelihood of a source occurring in the particular cell by spatially superimposing and summing the ROC of the respective sensor nodes. In other words, we *add on positive observations and subtract on negative*.
4. *Maximization*: The maximum of the likelihood matrix points to the estimated event location. If more than one elements of the likelihood matrix have the same maximum value, the estimated event position is the centroid of the corresponding cell centers.

Definition 9.4.1. *Let L_{max} denote the maximum of the likelihood matrix constructed by SNAP and A_{max} the set of cells of the matrix that attain this value. Also, let A^k_{max} denote the cells of the matrix that attain the values $L(i,j) \geq L_{max} - k$ for $k = 1...L_{max}$.*

Note that $A_{max} \subseteq A^1_{max} \subseteq A^2_{max}$....

Definition 9.4.2. *Let the diameter of a set of cells denote the maximum pairwise distance between any two cells in the set.*

Lemma 9.4.1. *Assuming the maximum of the likelihood matrix constructed by SNAP is $L_{max} > 1$ and A_{max} includes the true source location, then the estimation error caused by a faulty sensor node, independent of its position, is upper bounded by $d_1/2$ where d_1 is the diameter of A^1_{max}.*

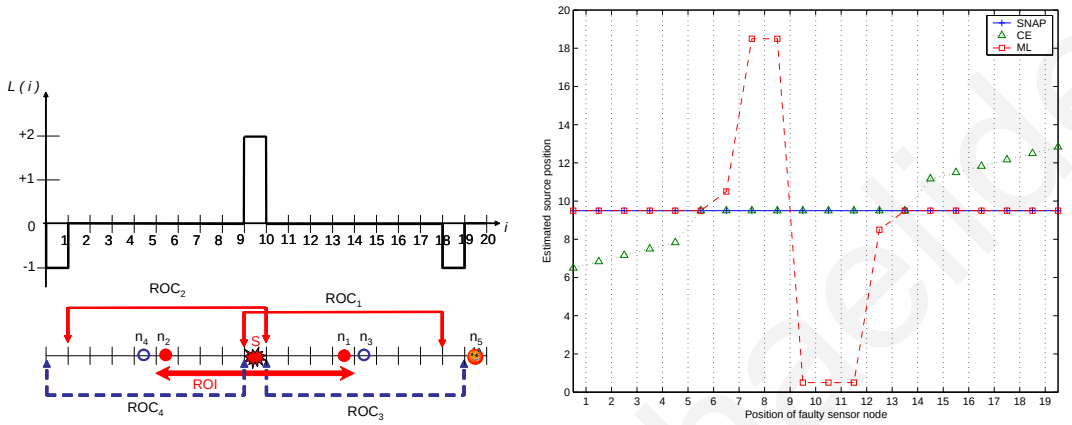
Proof: According to the SNAP algorithm, a faulty sensor node can only change the likelihood matrix by at most ± 1. Since the true event location is included in A_{max}, we only need to consider what happens to cells inside A^1_{max} since they are the only possible candidates for achieving the L_{max}. A faulty sensor can be a fault positive or a fault negative so have to investigate two separate cases: 1. A fault negative can only influence the SNAP estimation results if there are common cells between its ROC and A_{max}. Since it can subtract at most one from cells of A_{max}, the true source location is guaranteed to be included in A^1_{max} and the estimation error cannot be larger than half the diameter d_1. 2. A fault positive can only influence the SNAP estimation results by adding at most one and causing some cells of A^1_{max} to attain the value L_{max}. The true source location is still guaranteed to be included in A^1_{max} and the estimation error cannot be larger than half the diameter d_1. This completes the proof. ∎

Corollary 9.4.1. *Assuming the maximum of the likelihood matrix constructed by SNAP is $L_{max} > 1$, A_{max} is a single cell that includes the true source location and $A^1_{max} - A_{max} = \emptyset$ then the estimation error caused by a faulty sensor node, independent of its position, is 0.*

Proof: Since $A^1_{max} - A_{max} = \emptyset$, no cells attain the value $L_{max} - 1$. Therefore, a single faulty sensor with a ± 1 contribution cannot alter A_{max}. This completes the proof. ∎

The fault tolerant behavior of SNAP results from 2 main characteristics: First, to construct the likelihood function at a source location θ it uses only local information in the sense that it uses only the data from the sensors that are inside the ROI of θ. Therefore, false positives away from the source location have no influence on the estimation results. Second, and most important, it bounds the error that a faulty sensor can cause to the likelihood function by allowing a sensor to subtract *at most* one from the corresponding cells in the likelihood matrix. So a *single* healthy sensor close to the source can correct the error in the likelihood function caused by the faulty sensor.

9.4.4 Test Case



(a) Source and sensor positions and the resulting likelihood function.

(b) Estimated source location using the 3 different algorithms- SNAP, CE and ML-, as we vary the position of the faulty sensor node.

Figure 9.2: 1-D example with 5 sensor nodes, one of which is faulty (n_5).

Consider the 1-D scenario displayed in Fig. 9.2 where the line is divided into 20 equal cells. The event is located at position ($x_s = 9.5$) and we try to estimate its position using 4 sensor nodes which are located as shown in the figure. Two of the sensor nodes fall inside the source ROI and are alarmed ($x_1 = 13.5$, $x_2 = 5.5$), the other two fall outside and are non-alarmed ($x_3 = 14.5$, $x_4 = 4.5$). According to the SNAP algorithm described above, the alarmed sensor nodes provide +1 contribution in the cells inside their ROC, while the non-alarmed sensors provide -1 contribution. The sum of the contributions in each cell i gives the likelihood $L(i)$ of the source occurring in that cell and is plotted over the corresponding cells in Fig. 9.2(a) above the sensor locations. The maximum of this likelihood plot corresponds to the correctly estimated event location using SNAP. In fact, in the absence of faults all 3 estimation algorithms correctly estimate the event position.

Now consider a fifth sensor node which is faulty and is non-alarmed when positioned inside the source ROI and alarmed when positioned outside. Fig. 9.2(b) shows the estimated source location using the 3 different algorithms, as we vary the position of the faulty sensor node. From the plot, it is evident that only SNAP displays a fault tolerant behavior for all positions of the faulty sensor node; from Fig. 9.2(a) it is clear that the maximum of $L(i)$ cannot change no matter where we place the faulty sensor node. This result is expected from Corollary 9.4.1. ML fails to estimate the source location when the faulty sensor node is close to the event (false negative), while CE fails to estimate the event location when the faulty sensor node is

far away (false positive).

9.4.5 Fault Tolerant Maximum Likelihood (FTML)

The problem with ML is that it does not take into account the fact that any sensor may be faulty with some probability. This can be addressed by incorporating the probability of faults P_f when constructing the likelihood function. When we do this the log-likelihood function in (9.2) becomes:

$$\begin{aligned} L^{'}(\theta) &= \sum_{n=1}^{N}\sum_{t=1}^{M}\{I_{n,t}\log\left[P_{n,t}(1-P_f)+(1-P_{n,t})P_f\right] \\ &\quad +(1-I_{n,t})\log\left[(1-P_{n,t})(1-P_f)+P_{n,t}P_f\right]\}\,. \end{aligned} \tag{9.3}$$

In other words, when the indicator function takes the value 1, a sensor node can be correctly alarmed with probability $1-P_f$, or it can be falsely alarmed and produce an erroneous observation with probability P_f. Similar reasoning can be applied to the case where the indicator function takes the value 0. The modified likelihood function in (9.3) gives the desired fault tolerance to the maximum likelihood estimator because the functions in the $\log(\cdot)$ are always lower bounded by a small positive number. This way, the contribution of a faulty sensor never becomes unbounded. We refer to the estimator that uses this modified log-likelihood function as *Fault Tolerant Maximum Likelihood (FTML)* estimator.

FTML and SNAP can achieve similar accuracy but SNAP is computationally less demanding and makes fewer parameter assumptions. Recall that FTML requires numerical integration and utilizes the exact propagation model (see Section 3.1) together with information about the noise variance σ_w^2 and the fault probability P_f. SNAP on the other hand, only requires the estimation of the source ROI which can be obtained using various heuristics, e.g., from the percentage of alarmed sensor nodes in the neighborhood of the source. Finally, it is worth pointing that both FTML and SNAP can be implemented in a distributed fashion where any leader node can run the algorithms using *only* information from its neighbors (see dSNAP in Chapter 8).

9.5 Simulation Results

For all subsequent experiments (unless otherwise specified) we use a square 100×100 sensor field with 200 randomly deployed sensor nodes where the sensor readings are given by:

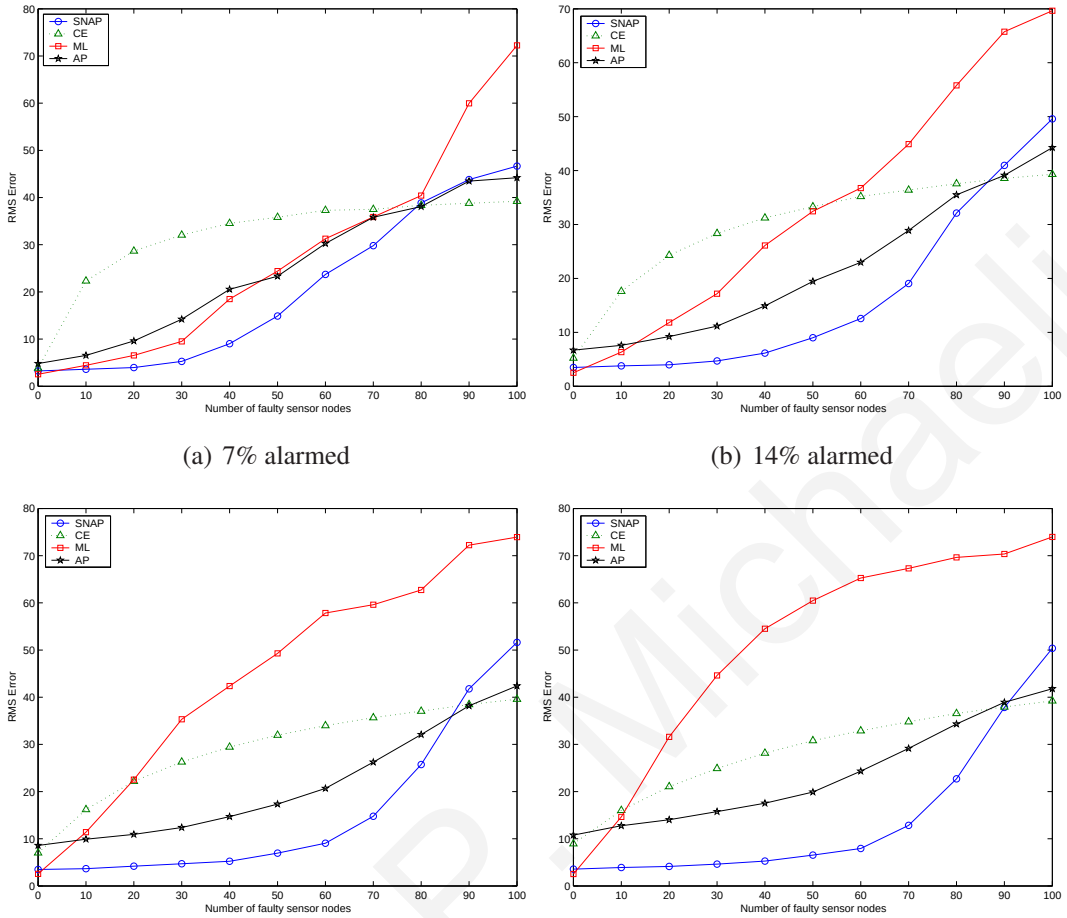
$$z_{n,t} = \min \left\{ 3000, \frac{3000}{r_n^2} + w_{n,t} \right\} \tag{9.4}$$

for $n = 1, \cdots, N$, $t = 1, \cdots, M$. Furthermore, we assume $w_{n,t}$ to be Gaussian $\mathcal{N}(0, 1)$, $M = 1$, and $T = 5$. For SNAP we use grid resolution $g = 1$ and ROC radius $R_c = 24.5$. For the fault tolerance analysis, we use the fault model of Section 9.3. Finally, the RMS error reported is the average over 500 Monte-Carlo simulations. For all experiments we use Matlab.

9.5.1 SNAP

In the first set of experiments, we investigate the fault tolerance of SNAP against CE, ML and AP. Remember from Chapter 8, that the AP (Add Positive) algorithm is a variant of SNAP that uses only the positive contributions from the alarmed sensors inside the ROI. For the fault tolerance analysis, we vary c to obtain different percentages of alarmed sensor nodes in the field.

Fig. 9.3 displays the results for the fault tolerance analysis of the different estimators. Although in the absence of faults the performance of ML and SNAP was very similar, the same cannot be stated here. In fact both CE and ML are very sensitive to sensor faults and loose accuracy continuously as the number of faults increase. This is especially evident for ML in situations with higher percentages of alarmed sensor nodes (see Fig. 9.3(c),9.3(d)). SNAP however, as it can be observed from these plots, displays a fault tolerant behavior and looses very little in accuracy even when 50 out of the 200 sensor nodes exhibit erroneous behavior. In fact, its fault tolerance improves when we increase the percentage of alarmed sensor nodes in the field. The intelligent construction of the likelihood function makes individual sensor faults unimportant in estimating the correct result. Finally, AP displays a similar fault tolerant behavior with SNAP though the estimation error is larger than SNAP. Note however, that for a range of faulty sensors from 10 to 80, AP exhibits better performance than both ML and CE.



(a) 7% alarmed (b) 14% alarmed

(c) 21% alarmed (d) 28% alarmed

Figure 9.3: Estimator performance for different percentages of alarmed sensor nodes as we vary the number of faulty sensor nodes in the field.

Dropped Packets

In this section, we investigate the performance of the four algorithms if packets are dropped by the network. Recall from Assumption 3 in Chapter 3 that for the network we investigate, alarmed sensors send a packet to the sink while non-alarmed sensors remain silent. Thus, if the sink does not receive a packet from a node, it assumes that the node is in the non-alarmed state. Therefore, to investigate the effect of dropped packets we change the fault model to allow *only* alarmed sensors to randomly flip their state (false negatives) with probability P_d

which corresponds to the probability of dropping a packet[1].

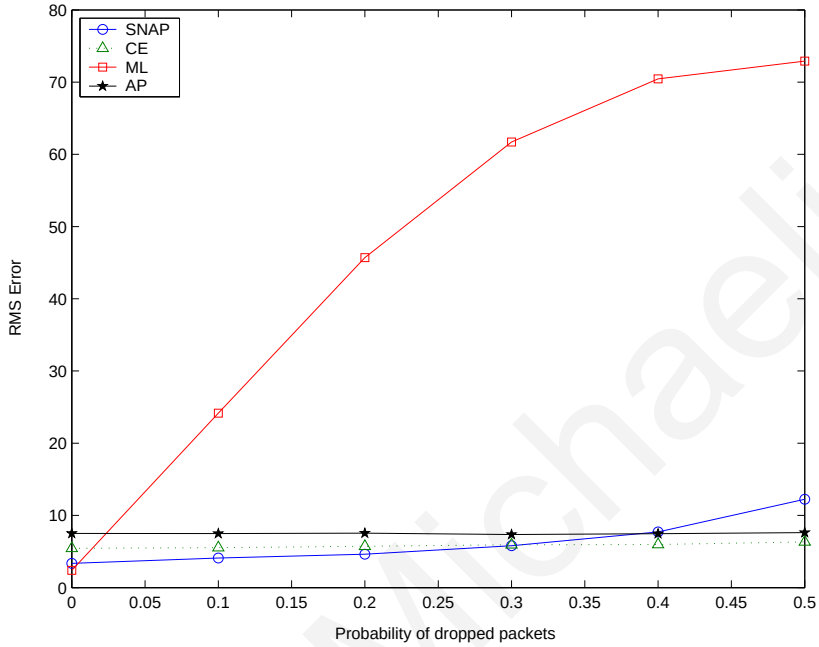


Figure 9.4: Estimator performance vs. probability of dropped packets P_d.

In Fig. 9.4 we investigate the performance of the estimators in the presence of false negatives. ML, as expected from the analysis in Section 9.4, looses accuracy immediately in the presence of false negatives. SNAP is the best estimator for values of $P_d \leq 0.3$. For higher percentages of dropped packets, CE becomes the better option. This is due to the increasing number of false negatives that counteract the small number of correctly alarmed sensor nodes left in the field when using SNAP. AP does not have this problem, so it displays a similar robust behavior to false negatives as CE. For the CE and AP, even one correctly alarmed sensor can localize the source at its own location since both algorithms completely neglect the non-alarmed sensor nodes in computing the event location.

Board Overheating

For symmetry, this section investigates the effect of the board overheating which, as reported in [38], caused the nodes to report false events. In Fig. 9.5 we investigate the performance of

[1]For simplicity we assume that each packet has equal probability of being dropped irrespective of the number of hops that it has to travel before it reaches the sink. This assumption is not expected to affect the fault tolerance analysis however.

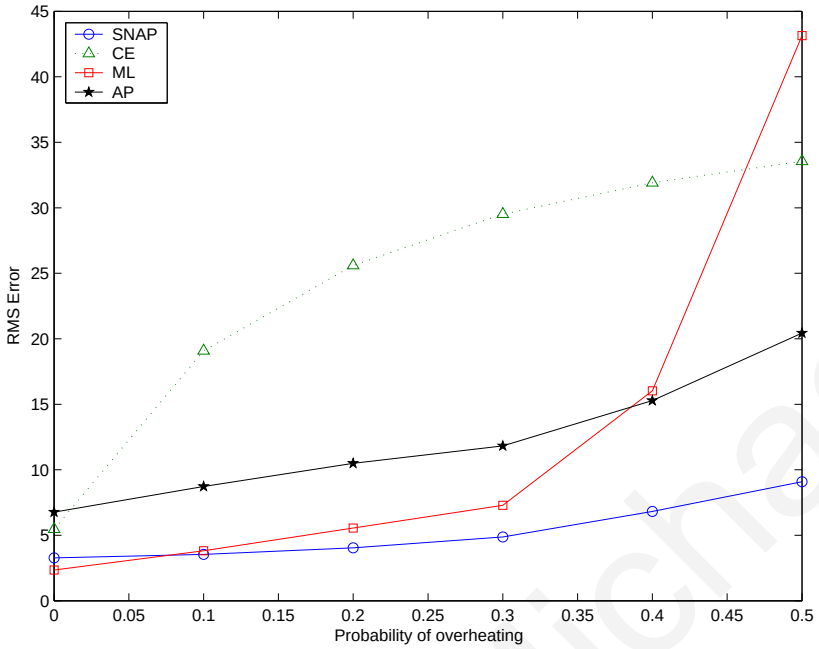


Figure 9.5: Estimator performance vs. probability of overheating P_o.

the estimators in the presence of false positives. We simulate the presence of false positives by varying the probability P_o that a non-alarmed sensor node produces a positive observation. CE displays the worst performance in the presence of false positives; this is expected from the analysis in Section 9.4. SNAP on the other hand, displays the most robust behavior against these overheating faults. The importance of the non-alarmed sensor nodes for SNAP in correcting false positives, is also evident by looking at the large difference in performance between SNAP and AP. ML steadily looses performance, and for values of $P_o \geq 0.4$ it becomes even worse than AP.

9.5.2 FTML

Next, we investigate the proposed Fault Tolerant Maximum Likelihood Estimator. Fig. 9.6 displays the estimation error as a function of the fault probability P_f. In the absence of faults ($P_f = 0$), ML and FTML display the same performance (as expected) which is slightly better than SNAP and significantly better than CE. As P_f increases, the performance of both CE and ML drops quickly (their RMS error increases). On the other hand, FTML and SNAP as it can be observed from the figure, display a similar fault tolerant behavior and loose very little in accuracy even when $P_f = 0.25$- this would be equivalent to 50 out of the 200 sensor

nodes exhibiting erroneous behavior!

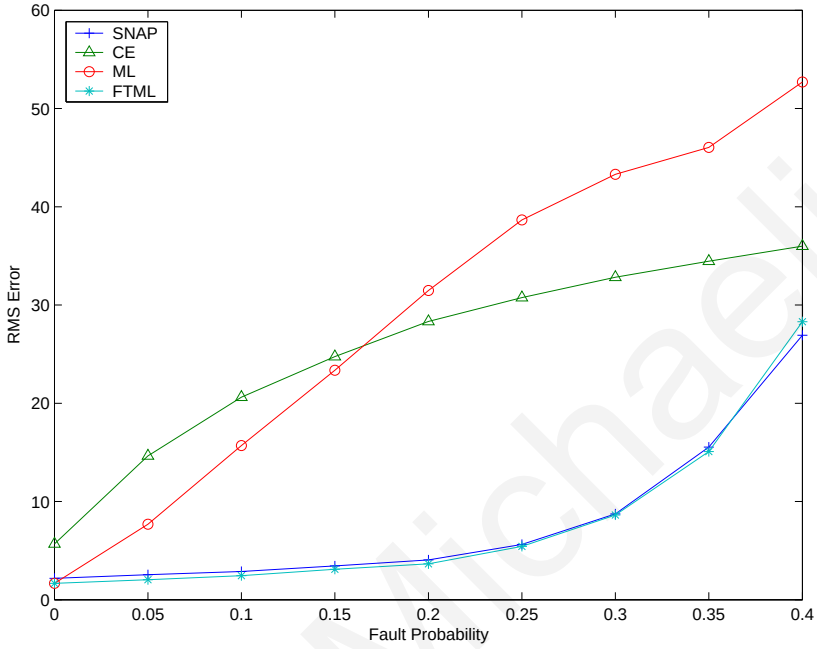
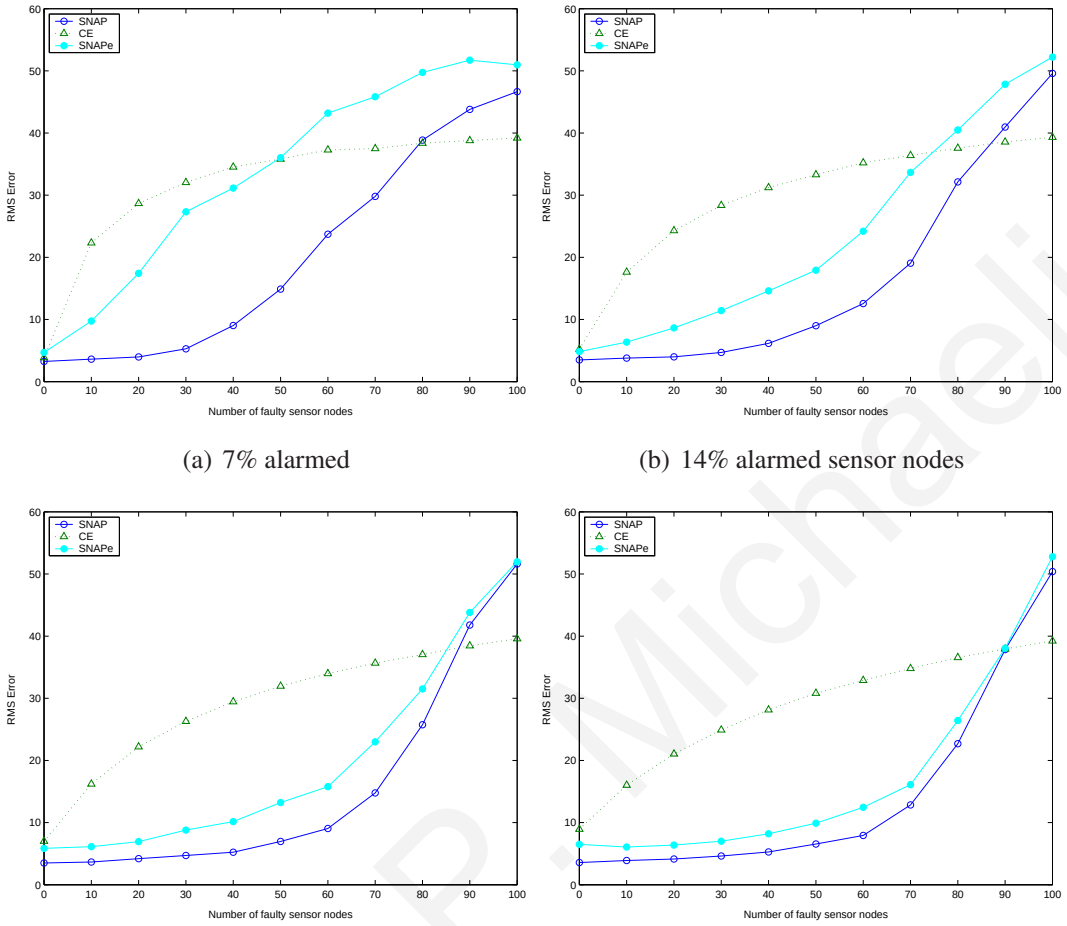


Figure 9.6: FTML performance vs. fault probability.

9.5.3 SNAPe

In this section, we investigate the fault tolerance of SNAPe (see Chapter 8), the version of SNAP that does not require knowledge of the source amplitude c or the threshold T. Remember that for SNAPe, R_c is estimated using the fraction of alarmed sensor nodes. Fig. 9.7 displays the fault tolerance results for SNAPe. CE and SNAP are also shown on the same plot for comparison purposes. For low percentages of alarmed sensor nodes (see Fig. 9.7(a)) SNAPe is very sensitive to sensor faults. As we increase the percentage of alarmed sensor nodes in the field however, SNAPe continuously improves in performance, and for higher percentages of alarmed sensor nodes (see Fig. 9.7(d)), it approximates the same fault tolerant behavior as SNAP! This result shows the robustness of SNAP for applications where we have a large number of alarmed sensor nodes. Using only information about the fraction of alarmed sensors in the field and binary information from the sensor nodes, it can localize the event position even when as many as 25% of the sensor nodes are faulty.



(a) 7% alarmed

(b) 14% alarmed sensor nodes

(c) 21% alarmed

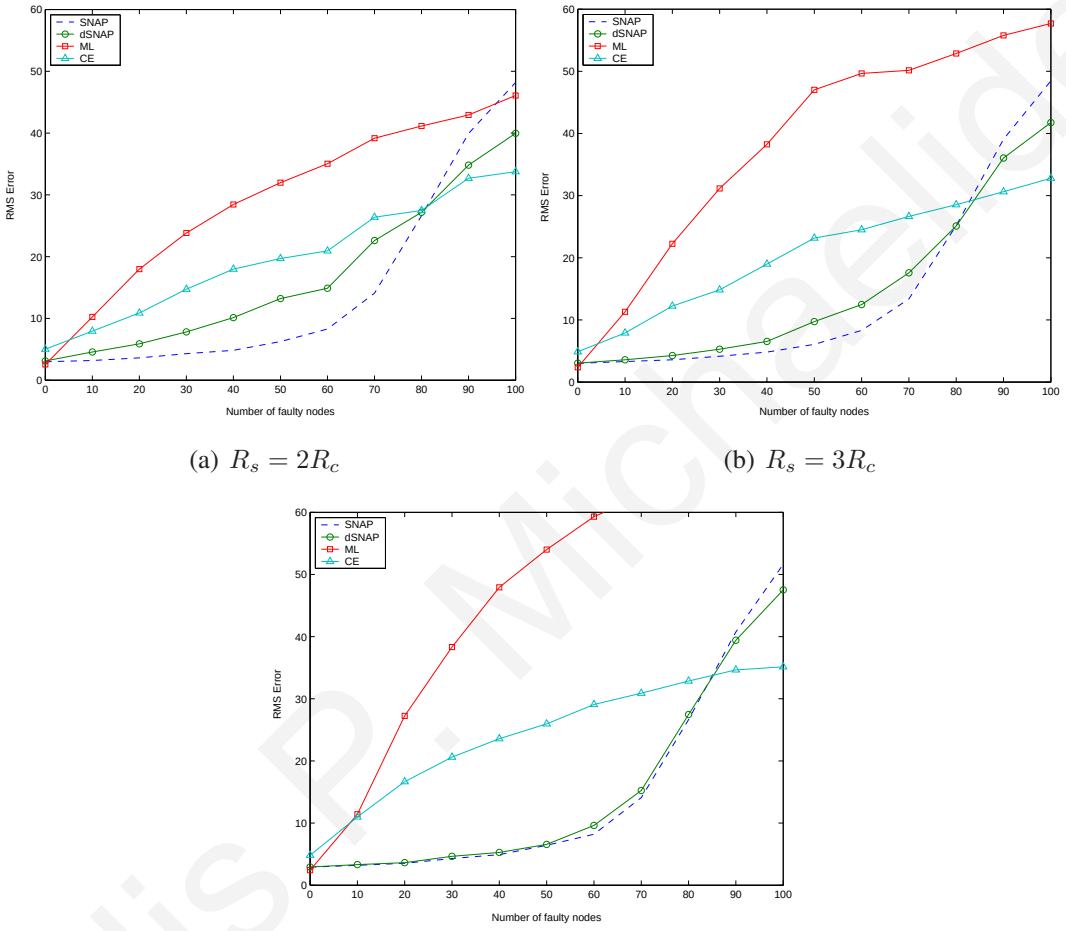
(d) 28% alarmed

Figure 9.7: SNAPe performance for different percentages of alarmed sensor nodes as we vary the number of faulty sensor nodes in the field.

9.5.4 dSNAP

The final set of experiments investigates the fault tolerance of dSNAP which is the distributed implementation of SNAP proposed in Chapter 8. For all subsequent experiments we use a sensor field with 100 randomly deployed nodes. We investigate the ROS size as we increase the number of faults in the field according to the fault model. Remember that the ROS defines the neighborhood that the leader node needs to obtain information relevant to the estimation problem. We test the performance of dSNAP against the distributed versions of the two other binary algorithms considered, CE and ML (see Section 2.6 of Chapter 2). For the CE, the event location is estimated as the centroid of the locations of all alarmed sensor nodes inside the leader's ROS_l. For the ML, the leader only contacts the sensor nodes inside its ROS_l for

evaluating the likelihood function. Finally, for reference, we also present the performance of the centralized algorithm, labeled as SNAP.



(a) $R_s = 2R_c$ (b) $R_s = 3R_c$

(c) $R_s = 4R_c$

Figure 9.8: Fault Tolerance of dSNAP as we vary the ROS size.

The results are displayed in Fig. 9.8. From the plots it is evident that dSNAP outperforms the other estimators (CE and ML). For dSNAP, we see that the choice of ROS is important in achieving the desired fault tolerance. In the absence of faults dSNAP achieves the performance of the centralized algorithm as expected, so it would make sense to use $R_s = 2R_c$ for this case. As we increase the faults however, we need to increase the ROS size to achieve the performance of the centralized algorithm. This can be attributed to the increased probability of sensor nodes becoming alarmed outside the event ROI. For CE and ML, increasing the ROS actually makes their performance worse; the reason is that CE is especially sensitive to false positives and ML to false negatives.

9.6 Summary

This chapter investigates the fault tolerance of different estimators that can be applied for localizing an event in a sensor network given only binary data from the sensor nodes: Centroid Estimator (CE), Maximum Likelihood (ML), and SNAP (developed in Chapter 8). We show that the SNAP algorithm retains its accuracy even when a large percentage of the sensor nodes report erroneous observations. Our results indicate that SNAP is significantly more fault tolerant than both CE and ML. Furthermore, we develop a fault tolerant version of the maximum likelihood estimator (FTML). Our results indicate that SNAP can achieve similar accuracy and fault tolerance to FTML by making simpler calculations and fewer assumptions.

CHAPTER 10

LOCALIZATION USING SNAP OF SOURCES WITH NON-CIRCULAR FOOTPRINT

10.1 Overview

This chapter further investigates the use of the SNAP algorithm for applications of event localization in Wireless Sensor Networks (WSNs). In our work so far and the overwhelming majority of the papers that deal with event detection and localization, it is assumed that the source has a circular footprint in the sense that the event generates a signal that propagates uniformly in all directions, thus it can be detected by any sensor that is located in a circular disc around the source. The contribution of this chapter is that it deals with events with non-circular footprint. Specifically, we consider a covariance detection algorithm and a plume propagation model where the event footprint is not the usual circular disc.

10.2 Introduction

In Chapter 8, we proposed SNAP (Subtract on Negative Add on Positive), a very simple and fairly accurate algorithm for localizing events in WSNs using only binary data from the sensor nodes. In this context, the sensors make binary observations and become alarmed (positive observations) if the calculated test statistic at their location is above a threshold; otherwise they remain silent (negative observations). Based on the binary beliefs of all sensor nodes, SNAP can estimate the location of the event by building a likelihood matrix whose

maximum points to the event location. The main ingredient of the algorithm for constructing the likelihood matrix is the Region of Coverage (ROC), an area in the vicinity of each sensor node where it can infer information relevant to the estimation problem. For events with circular footprint the ROC becomes a disc centered at the sensor node location; this case has been investigated in Chapter 8. Uniform propagation models and circular footprints may be accurate for sources that emit sound or electromagnetic waves, but they are not appropriate for several other problems. An important feature of SNAP, is that it can be applied for localizing event sources with *non-circular* footprint by appropriately adapting the shape, size and orientation of the ROC based on the underlying detection algorithm and the event propagation characteristics.

The main contribution of this chapter is to investigate event localization using SNAP for two specific cases where the source has a non-circular footprint. First, we investigate the case where pairs of sensors collaborate in order to decide their alarm status. In some applications, two sensors that are located close to each other are expected to record similar measurements. Moreover, the two sensors may be able to exploit this similarity to improve coverage using a Covariance Detection (CD) as in Chapter 5. The appropriate ROC to be used in this case, is not a circular disc but an ellipse. These ellipses between pairs of sensor nodes can be effectively used by the SNAP algorithm to estimate the event location. To the best of our knowledge, SNAP with an elliptic ROC is one of the first algorithms that can exploit the correlation between the measurements of pairs of sensors for localization purposes. The second problem we investigate involves a scenario where the signal or substance does not propagate uniformly in all directions. Rather, we assume that there exists a draft that pushes the signal in one direction. Such scenario may arise in environmental monitoring applications when trying to localize the position of a plume source. In this context, the source emits a signal (e.g., smoke) which is then pushed in one direction by the wind. For such scenarios the ROC used by SNAP is actually the "mirror" image of the source footprint.

The chapter is organized as follows. First, in Section 10.3, we show how the SNAP algorithm can exploit covariance information for localization. Then, Section 10.4 investigates plume source localization. Finally, we conclude with a summary of the main results of this chapter in Section 10.5.

10.3 SNAP using Covariance Detector

In this section, we investigate how the ROC needs to adapt based on the underlying detection algorithm used for determining the alarm state of each sensor. The circular shaped ROC investigated in Chapter 8, would be appropriate for detection algorithms like the Mean Detector (MD) proposed in Chapter 4, or the Energy Detector (ED) proposed in Chapter 5. In this section, we investigate the Covariance Detector (CD) proposed in Chapter 5, that results in an elliptic shaped ROC.

10.3.1 Elliptic ROC

For the Covariance Detector (CD) covariance information between two closest neighbors is the deciding factor for detecting the event. The test statistic used is the sample covariance of the measurements between two closest neighbors compared to a constant threshold T. For a pair of sensor nodes using the CD, as shown in Chapter 5, the *ROC becomes an ellipse* with the two sensor nodes located at the foci. Note that the size of the ROC depends on the separating distance d between the two sensor nodes. For $d = 0$, the two sensors are located at the same point, and the coverage area attains its maximum possible value (a circle). At this point it is worth pointing out that for SNAP only "unique pairs" of sensor nodes are used for the construction of the likelihood matrix $\mathcal{L}$. For example, if node i is closest neighbor of j and j is the closest neighbor of i, then only one of the two updates $\mathcal{L}$. Fig. 10.1 shows an example of the ROC ellipse contour plots created by SNAP for 25 randomly deployed sensor nodes with the source located in the middle of the field.

10.3.2 Elliptic ROC Simulation Results

In this section, we show how covariance information can be also used to accurately localize the event using SNAP. We compare the performance of SNAP against 2 other binary estimators that have been previously defined[1]: the Centroid Estimator (CE) and Add Positive (AP). For all subsequent experiments, we use a square sensor field of 100×100 with 200 randomly deployed sensors and assume that the sensor measurements were given by the stochastic propagation model described in Section 3.2 of Chapter 3. Also, it is assumed that $c = 1000$, $\lambda_v = \lambda_c = 50$, $\sigma_W^2 = 1$, $\sigma_S^2 = 10$ and $M = 1000$. The sensor nodes utilize

[1]For CE see Section 2.6 of Chapter 2 and for AP see Section 8.6.3 of Chapter 8.

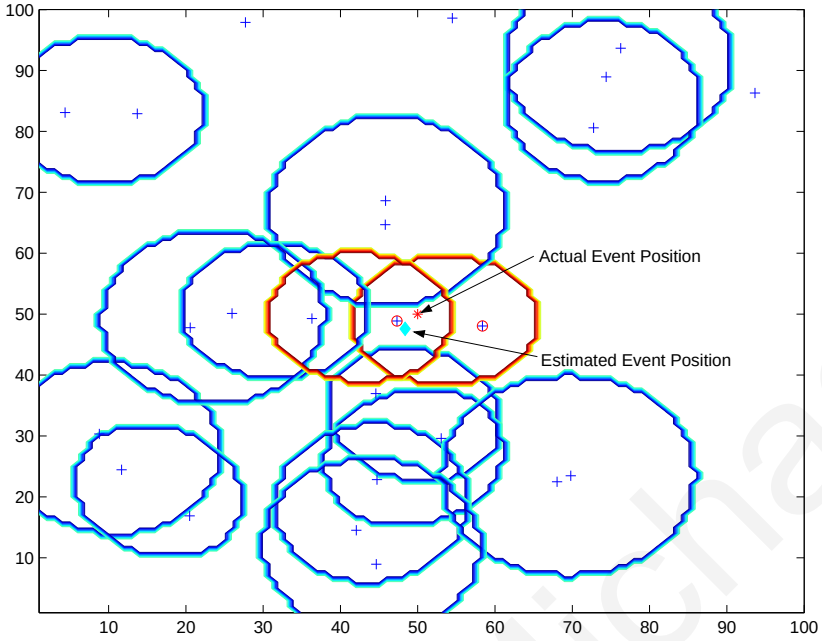
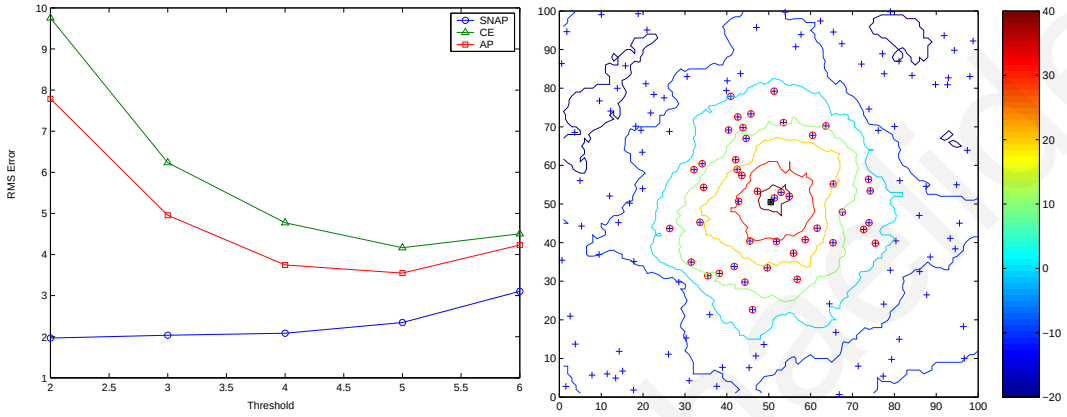


Figure 10.1: SNAP with elliptic ROC localization example using covariance information from 25 randomly deployed sensor nodes. Alarmed nodes create a "positive" ROC (filled with $+1$) with contour indicated with red color while non-alarmed nodes create a "negative" ROC (filled with -1) with contour indicated with blue color.

the covariance test statistic for the CD to determine their alarm status. For all experiments, the sensor field is initialized at the beginning and remains fixed throughout the experiments. Finally, the RMS error reported is the average over 500 randomly deployed sources. For all experiments we use Matlab.

The results are displayed in Fig. 10.2. Fig. 10.2(a) shows the SNAP performance using the elliptic shaped ROC as we vary the detection threshold T, while Fig. 10.2(b) demonstrates a snapshot of the contour plot of the likelihood matrix constructed using SNAP for $T = 3$. For all experiments, SNAP achieved a fairly robust behavior to a large range of thresholds. The robust behavior of SNAP is especially evident for smaller threshold values where we have a larger number of alarmed sensor nodes. This is especially important in estimation because smaller thresholds allow the detection of weaker signals and also increase the system fault tolerance. Note that the performance of all algorithms deteriorates for thresholds above 6 mainly because the ROC of each sensor becomes very small and we run into some situations with no alarmed sensor nodes that cause large estimation errors (in those situations, since we cannot employ the localization algorithms, we assume that the event is located at the center of the field). Also, it is worth pointing out that the comparison between SNAP and AP

demonstrates once more the value of utilizing the non-alarmed sensor nodes in localizing the event.



(a) (b) $T = 3$

Figure 10.2: SNAP using elliptic ROC performance evaluation: a) RMS error vs. threshold b) Likelihood matrix contour snapshot for $T = 3$.

10.4 Plume Source Localization using SNAP

In this section we will investigate how the ROC needs to adapt based on the event propagation characteristics. Specifically, we first show that for the case of a directed propagation model the ROC becomes the "mirror" shape of the source ROI. Next, we extend these results for localization of pollutant sources with Gaussian concentration distribution using SNAP.

10.4.1 Directed Propagation Model

For this section we assume the simplified signal/substance propagation model described in Section 3.3 of Chapter 3 which takes into account environmental conditions like wind or current flow. Specifically, we assume that there is a draft in a constant direction. As a result, the substance is spread in a limited angle ϕ_s as shown in Fig. 10.3. For this propagation model, the Region of Influence (ROI) is illustrated with a triangle in Fig. 10.3. In this example, we assume that the direction of propagation is along the x-axis and thus the signal spreading is constrained within two rays that pass from the point of the event with angles $-\phi_s/2$ to

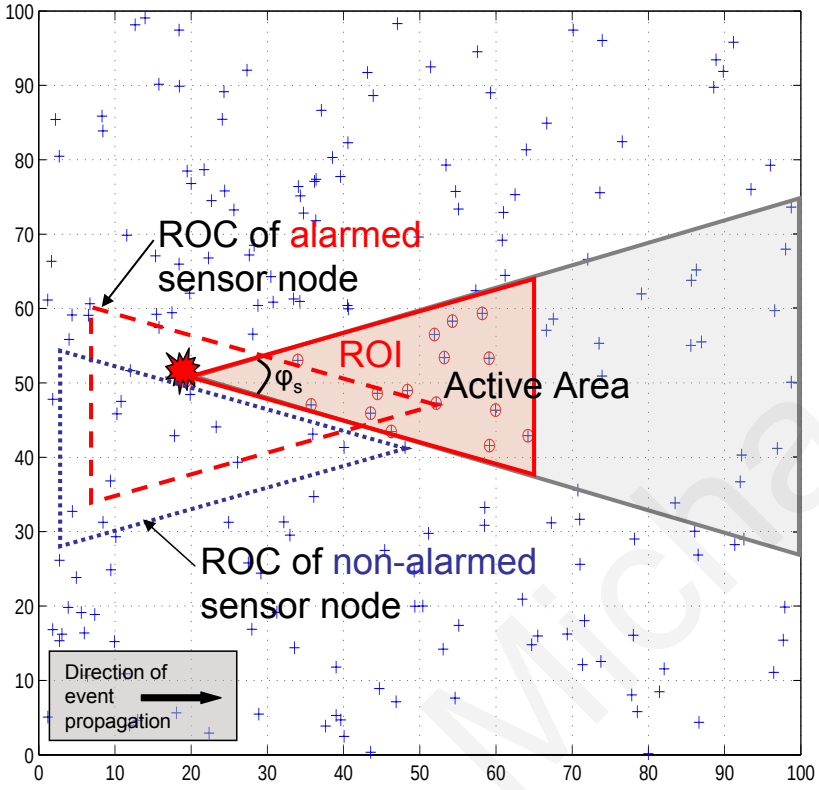


Figure 10.3: Directed event propagation model with Active Area $\mathcal{A}$ defined by direction of propagation and spread angle ϕ_s. For this case the ROC becomes the "mirror" shape of the source ROI as indicated on the figure.

$\phi_s/2$. Given this constrain, a sensor will receive some signal information only if the source is located within the two rays that pass from the sensor and with angles $\pi \pm \phi_s/2$. So for the directed propagation model the ROC becomes the "mirror" shape of the source ROI (see Fig. 10.3). In other words, the ROC of the alarmed sensor node contains all the possible source locations from where the event could have originated in order for the particular sensor node to become alarmed. On the other hand, the ROC of the non-alarmed sensor node contains all the "impossible" source locations where the event *could not* have originated.

10.4.2 Plume ROC Simulation Results

In this section we present results on SNAP localization for the Gaussian Plume Model [60] with one-dimensional spreading and a steady wind direction along the x-axis. This creates a "drop" shaped ROI that depends on the source parameters and the threshold. For constructing

the plume ROC we use the "mirror" shape of the source ROI as explained in the previous section (see also [59] where a similar ROC has been used). Fig. 10.4 shows an example of the plume ROC contour plots created by SNAP using 25 randomly deployed sensor nodes.

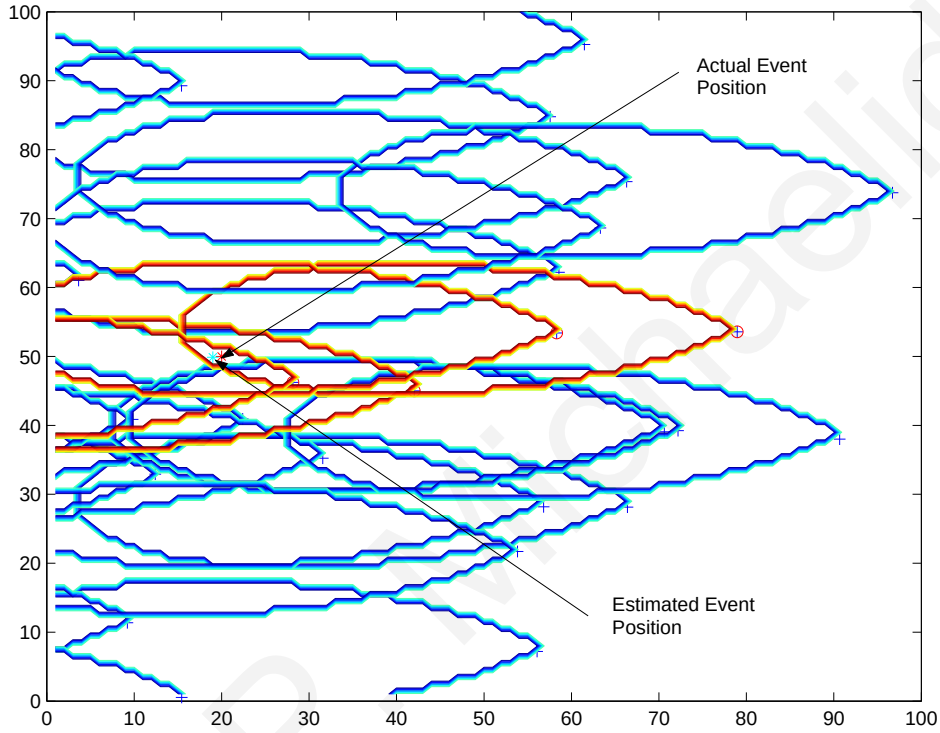


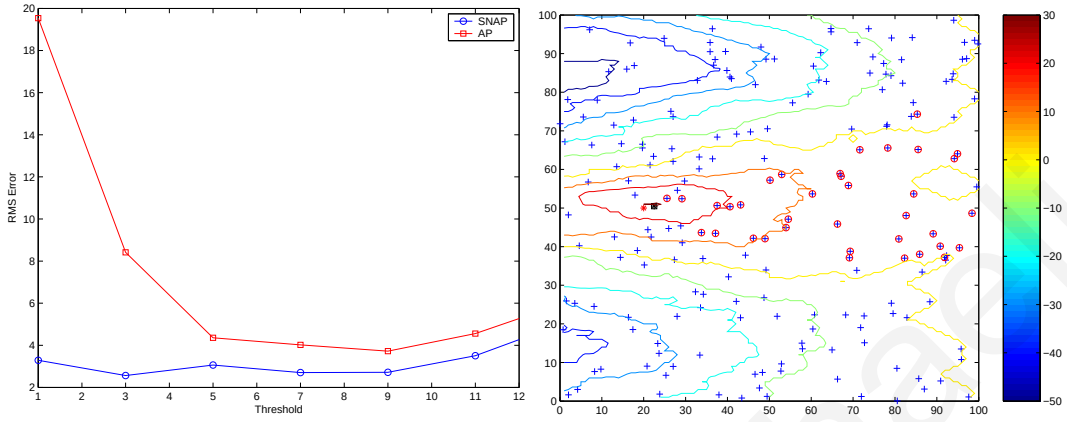
Figure 10.4: SNAP with plume ROC localization example using 25 randomly deployed sensor nodes. Alarmed nodes create a "positive" ROC (filled with $+1$) with contour indicated with red color while non-alarmed nodes create a "negative" ROC (filled with -1) with contour indicated with blue color.

For the simulation results we use the model in Section 3.3 with $W \sim \mathcal{N}(0,1)$, $\phi_s = \pi^2$ and $S_{n,t}$ given by (3.10). Furthermore, we use $c = 200$, $b = 0.25$ and a 100×100 field with 200 randomly deployed nodes. Also, we assume the event source is placed at position $(20, 50)$ and take the average over 500 randomly deployed fields to calculate the RMS error.

The results are displayed in Fig. 10.5. Fig. 10.5(a) shows the SNAP performance using the plume shaped ROC as we vary the detection threshold T, while Fig. 10.5(b) demonstrates a snapshot of the contour plot of the likelihood matrix constructed using SNAP for $T = 3$. From these results, we see that SNAP can be also effectively applied for the localization of

[2]The Gaussian plume model with one-dimensional spreading in Section 3.3 is only defined in the positive-x direction.

certain plume sources when its ROC is properly configured.



(a) (b) $T = 3$

Figure 10.5: SNAP using plume ROC performance evaluation: a) RMS error vs. threshold b) Likelihood matrix contour snapshot for $T = 3$.

10.5 Summary

This chapter investigates the SNAP algorithm, proposed in Chapter 8, for event sources with non-circular footprint. We show how the ROC can be adapted, in terms of size, shape and orientation in order to achieve accurate localization results when using SNAP for two different event conditions. First, our results indicate that covariance information between measurements of sensor nodes in proximity can be used by SNAP to achieve accurate event localization. Second, we show how SNAP can be applied for localizing events with directed propagation (for example in problems of environmental pollution).

CHAPTER 11

CONCLUSION

This dissertation focuses on distributed event detection and localization in Wireless Sensor Networks (WSNs). The proposed sensor network can deal with a number of environmental monitoring applications. The main objective of this dissertation is to detect and localize the event from the spatially distributed information provided by the sensor nodes in a simple, localized and fault tolerant manner. For the problem of distributed detection this dissertation proposes and analyzes two novel detection algorithms: the Covariance Detector (CD) and the Enhanced Covariance Detector (ECD) that exploit the spatial correlation between the measurements of sensor nodes in close proximity in order to improve the overall coverage of the sensor network. For the problem of distributed localization this dissertation proposes and analyzes two new algorithms: the SNAP (Subtract on Negative Add on Positive) algorithm and the Fault Tolerant Maximum Likelihood (FTML) algorithm. Both algorithms feature accuracy and robustness with respect to faults in the sensor network. In addition, SNAP has desirable properties such as low computational complexity and distributed implementation capability.

For our future work we plan to investigate other propagations models and noise distributions. For detection, we plan to investigate optimal fusion rules. For localization, we plan to apply the SNAP algorithm for sensor node localization, target tracking applications and situations where we have multiple sources. Below we provide some more details for each of these items:

Propagation Models: In the real world, events do not have uniform distributions because of varying environmental conditions. For our future work, we plan to investigate other propagations models, both static and dynamic. For static models, we will assume noise with different distribution characteristics.

Optimal Fusion Rules: At the local sensor level we will be investigating distributed detection

strategies for the case where we have three or more sensor nodes collaborating for improving their coverage. At the network level, from our work on event detection, we conjecture that the OR fusion rule at the base station (at least one sensor node reports detection) is actually the optimal fusion rule that achieves maximum coverage when energy constrains are taken into consideration (by constraining the number of sensors reporting to the base station to be relatively small) - this is based on extensive simulation results.

Sensor Node Localization: Most applications envisioned for WSNs, require the sensor nodes to know or estimate their locations. One of the common assumptions made for WSNs, is that a fraction of the sensor nodes uses GPS, while the rest estimate their location using distributed localization algorithms. An interesting research direction would be to use SNAP for localizing the remaining sensor nodes using received signal strength indication (RSSI) measurements.

Target Tracking: Mobile target tracking is one of the most fundamental collaborative information processing problems in WSNs. Tracking techniques, as opposed to the localization techniques discussed in this dissertation, usually incorporate a target mobility model and assume a probability distribution for the sensor measurement errors to achieve superior performance. They rely on Bayesian filtering variants, such as Kalman Filter or Particle Filter, to mitigate the effect of measurement noise and alleviate high positioning errors that do not reflect the target's mobility pattern. The SNAP algorithm provides high quality position estimates that reflect the active region of a moving target. However, under certain conditions, such as low node density and/or high probability of faulty nodes, higher positioning errors may be observed. Our approach will be to combine SNAP with a position filtering scheme (e.g., Kalman Filter) to smooth out the position estimates that do not correspond to the moving target's dynamics.

Multiple Sources: Having multiple sources makes the localization problem considerably more challenging. The way we plan to attack this problem is in two distinct phases. First, the number of sources need to be identified with a leader node elected for each distinct source. Then, each source can be localized by the respective leader node using SNAP.

Finally, we plan implement the proposed algorithms in a real-time application using WSNs.

BIBLIOGRAPHY

[1] M. Weiser, "The computer for the twenty-first century," *Scientific American*, pp. 94–104, Sep 1991.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, pp. 102–114, Aug. 2002.

[3] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. San Francisco, CA: Morgan Kaufmann, 2004.

[4] C. Chong and S. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, Aug. 2003.

[5] S. Kumar, F. Zhao, and D. Shepherd, "Collaborative signal and information processing in microsensor networks," *IEEE Signal Processing Magazine*, vol. 19, pp. 13–14, Mar. 2002.

[6] R. Viswanathan and P. Varshney, "Distributed detection with multiple sensors: Part I-fundamentals," *Proceedings of the IEEE*, vol. 85, Jan 1997.

[7] R. Blum, S. Kassam, and V. Poor, "Distributed detection with multiple sensors: Part II-advanced topics," *Proceedings of the IEEE*, vol. 85, Jan 1997.

[8] S. M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*. New Jersey: Prentice Hall PTR, 1993.

[9] H. Van Trees, *Detection, Estimation and Modulation Theory*. New York: John Wiley and Sons Inc., 2001.

[10] P. K. Varshney, *Distributed Detection and Data Fusion*. New York: Springer, 1997.

[11] Z. Chair and P. Varshney, "Optimal data fusion in multiple sensor detection systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 22, pp. 98–101, Jan. 1986.

[12] G. Lauer and N. Sandell, "Distributed detection of known signal in correlated noise," *ALPHATECH Report, Burlington, MA*, 1982.

[13] J. Tsitsiklis and M. Athans, "On the complexity of decentralized decision making and detection problems," *IEEE Transactions on Automatic Control*, May 1985.

[14] E. Drakopoulos and C. Lee, "Optimal multisensor fusion of correlated local decisions," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 27, pp. 593–606, July 1991.

[15] M. Kam, Q. Zhu, and S. Gray, "Optimal data fusion of correlated local decisions in multiple sensor detection systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 28, pp. 916–920, July 1992.

[16] P. Willett, P. Swaszek, and R. Blum, "The good, bad, and ugly: distributed detection of a known signal in dependent gaussian noise," *IEEE Transactions on Signal Processing*, vol. 48, pp. 3266–3279, Dec. 2000.

[17] Y. Sung, L. Tong, and A. Swami, "Asymptotic locally optimal detector for large-scale sensor networks under the poisson regime," *IEEE Transactions on Signal Processing*, vol. 53, March 2005.

[18] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Coverage problems in wireless ad-hoc sensor networks," in *INFOCOM*, pp. 1380–1387, 2001.

[19] M. Marengoni, B. Draper, A. Hanson, and R. Sitaraman, "A system to place observers on a polyhedral terrain in polynomial time," *Image and Vision Computing*, vol. 18, no. 10, pp. 773–780, 2000.

[20] S. Meguerdichian, G. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks," in *MOBICOM*, pp. 139–150, July 2001.

[21] C. Huang and Y. Tseng, "The coverage problem in a wireless sensor network," *MONET*, vol. 10, no. 4, pp. 519–528, 2005.

[22] D. Tian and N. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *First ACM Workshop on Wireless Sensor Networks and Applications*, 2002.

[23] N. Ahmed, S. Kanhere, and S. Jha, "Probabilistic coverage in wireless sensor networks," in *IEEE Local Computer Networks*, pp. 672–679, 2005.

[24] V. Phipatanasuphorn and P. Ramanathan, "Vulnerability of sensor networks to unauthorized traversal and monitoring," *IEEE Transactions on Computers*, vol. 53, pp. 364–369, March 2004.

[25] S. Adlakha and M. Srivastava, "Critical density thresholds for coverage in wireless sensor networks," in *IEEE Wireless Communications and Networking Conference (WCNC '03)*, Mar 2003.

[26] A. Jindal and K. Psounis, "Modeling spatially correlated data in sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 2, no. 4, pp. 466–499, 2006.

[27] M. Vuran, O. Akan, and I. Akyildiz, "Spatio-temporal correlation: theory and applications for wireless sensor networks," *Computer Networks*, vol. 45, pp. 245–259, March 2004.

[28] B. Krishnamachari and S. Iyengar, "Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks," *IEEE Transactions on Computers*, vol. 53, pp. 241–250, March 2004.

[29] J. Chen, R. Hudson, and K. Yao, "Maximum-likelihood source localization and unknown sensor location estimation for wideband signals in the near-field," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1843–1854, Aug. 2002.

[30] X. Sheng and Y. Hu, "Energy based acoustic source localization," in *Information Processing in Sensor Networks (IPSN), Second International Workshop, Palo Alto, CA, USA*, pp. 286–300, April 2003.

[31] D. Li, K. Wong, Y. Hu, and A. Sayeed, "Detection, classification, and tracking of targets," *IEEE Signal Processing Magazine*, vol. 19, pp. 17–29, Mar. 2002.

[32] G. Stuber and J. Caffery, *Radiolocation Techniques*. The Communications Handbook, CRC Press, 2002.

[33] M. Ding, F. Liu, A. Thaeler, D. Chen, and X. Cheng, "Fault-tolerant target localization in sensor networks," *EURASIP Journal on Wireless Communications and Networking*, 2007.

[34] R. Niu and P. Varshney, "Target location estimation in wireless sensor networks using binary data," in *38th Annual Conference on Information Sciences and Systems*, (Princeton, NJ), March 2004.

[35] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. New Jersey: Prentice Hall PTR, 1993.

[36] A. Czarlinska, W. Luh, and D. Kundur, "Attacks on sensing in hostile wireless sensor-actuator environments," in *IEEE Global Telecommunications Conference (GLOBECOM)*, pp. 1001–1005, 2007.

[37] A. Tanenbaum, C. Gamage, and B. Crispo, "Taking sensor networks from the lab to the jungle," *IEEE Computer*, vol. 39, no. 8, pp. 98–100, 2006.

[38] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, "A line in the sand: a wireless sensor network for target detection, classification, and tracking," *Computer Networks*, vol. 46, pp. 605–634, Dec. 2004.

[39] Q. Chen, K. Lam, and P. Fan, "Comments on "Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks"," *IEEE Transactions on Computers*, vol. 54, no. 9, pp. 1182–1183, 2005.

[40] X. Luo, M. Dong, and Y. Huang, "On distributed fault-tolerant detection in wireless sensor networks," *IEEE Transactions on Computers*, vol. 55, no. 1, pp. 58–70, 2006.

[41] E. Ould-Ahmed-Vall, G. Riley, and B. Heck, "A distributed fault-tolerant algorithm for event detection using heterogeneous wireless sensor networks," in *45th IEEE Conference on Decision and Control*, pp. 3634–3639, Dec. 2006.

[42] V. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello, "Bayesian filtering for location estimation," *IEEE Pervasive Computing*, vol. 2, no. 3, pp. 24–33, 2003.

[43] H. Yang and B. Sikdar, "A protocol for tracking mobile targets using sensor networks," in *First IEEE Workshop on Sensor Network Protocols and Applications*, pp. 71–81, 2003.

[44] N. Shrivastava, R. Madhow, U. Mudumbai, and S. Suri, "Target tracking with binary proximity sensors: Fundamental limits, minimal descriptions, and algorithms," in *4th International Conference on Embedded Networked Sensor Systems*, pp. 251–264, 2006.

[45] W.-P. Chen, J. Hou, and L. Sha, "Dynamic clustering for acoustic target tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 3, pp. 258–271, 2004.

[46] J. Liu, J. Reigh, P. Cheung, and F. Zhao, "Distributed group management in sensor networks: Algorithms and applications to localization and tracking," *Telecommunication Systems*, vol. 26, no. 2, 2004.

[47] Q. Fang, F. Zhao, and L. Guibas, "Counting targets: Building and managing aggregates in wireless sensor networks," tech. rep., Palo Alto Research Center (PARC), June 2002.

[48] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 61–72, 2002.

[49] M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," tech. rep., Xerox Palo Alto Research Center, 2001.

[50] D. McErlean and S. Narayanan, "Distributed detection and tracking in sensor networks," in *Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1174–1178, 2002.

[51] J. Kulesz, "Development of a common data highway for comprehensive incident management," tech. rep., Apr. 2003.

[52] S. Brennan, A. Mielke, D. Torney, and A. Maccabe, "Radiation detection with distributed sensor networks," *IEEE Computer Magazine*, pp. 57–59, Aug. 2004.

[53] W. Kates, "Robots to keep drinking water safe," May 2004. `www.cbsnews.com`.

[54] X. Yang, K. Ong, W. Dreschel, K. Zeng, C. Mungle, and C. Grimes, "Design of a wireless sensor network for long-term, in-situ monitoring of an aqueous environment," *Sensors*, vol. 2, pp. 455–472, 2002.

[55] A. Ailamaki, C. Faloutsos, P. Fischbeck, M. Small, and J. VanBriesen, "An environmental sensor network to determine drinking water quality and security," in *SIGMOD Record*, vol. 32, Dec. 2003.

[56] L. Sacks, M. Britton, I. Wokoma, A. Marbini, T. Adebutu, I. Marshall, C. Roadknight, J. Tateson, D. Robinson, and A. Gonzalez-Velazquez, "The development of a robust, autonomous sensor network platform for environmental monitoring," in *Proceedings of XII Conference on Sensors & their Applications*, 2003.

[57] `www.adastral.ucl.ac.uk/sensornets/secoas`.

[58] J. Farrell, S. Pang, and W. Li, "Plume mapping via hidden Markov methods," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 33, no. 6, Dec. 2003.

[59] G. Nofsinger, "Tracking based plume detection," *PhD Thesis, Dartmouth College, Hanover, New Hampshire*, 2006.

[60] R. B. Stull, *Meteorology for Scientists and Engineers, 2nd Ed.* Pacific Grove, CA, USA: Brooks Cole, 2000.

[61] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networks*, August 2000.

[62] S. Ross, *Introduction to Probability Models*. Academic Press, 8th ed., 2003.

[63] S. Nadarajah and S. Kotz, "Exact distribution of the max/min of two gaussian random variables," *IEEE Transactions on VLSI Systems*, vol. 16, pp. 210–212, Feb. 2008.

[64] J. Greunen and J. Rabaey, "Lightweight time synchronization for sensor networks," in *2nd ACM international conference on Wireless sensor networks and applications (WSNA'03)*, Sep. 2003.

[65] R. Niu, P. Varshney, M. Moore, and D. Klamer, "Decision fusion in a wireless sensor network with a large number of sensors," in *7th IEEE International Conference on Information Fusion (ICIF'04)*, (Stockholm, Sweden), June 2004.

[66] T. Coleman and Y. Li, "An interior, trust region approach for nonlinear minimization subject to bounds," *SIAM Journal on Optimization*, vol. 6, pp. 418–445, 1996.

CURRICULUM VITAE

Michalis P. Michaelides
E-mail: michalism@ucy.ac.cy, michalismp@cytanet.com.cy

Education

2004 - 2009 PhD in Electrical Engineering,
Electrical and Computer Engineering Department,
University of Cyprus.

1997 - 1998 Master of Science in Electrical Engineering,
Emphasis on systems and automatic controls,
Purdue University, West Lafayette, Indiana, USA. GPA: 3.8 / 4.0

1993 - 1997 Bachelor of Science in Electrical and Computer Engineering,
Purdue University, West Lafayette, Indiana, USA. GPA: 3.97 / 4.0

Publications

1. M. P. Michaelides and C. G. Panayiotou, "Plume source position estimation using sensor networks," in *13th Mediterranean Conference on Control and Automation*, pp. 731-736, Jun. 2005.

2. C. G. Panayiotou, D. Fatta, and M. P. Michaelides, "Environmental monitoring using sensor networks," in *Workshop on Modeling and Control of Complex Systems*, Ayia Napa, Cyprus, Jun. 2005.

3. M. P. Michaelides and C. G. Panayiotou, "Event source position estimation using sensor networks," in *14th Mediterranean Conference on Control and Automation*, Feb. 2006.

4. M. P. Michaelides and C. G. Panayiotou, "Event detection using sensor networks," in *45th IEEE Conference on Decision and Control*, pp. 6784-6789, Dec. 2006.

5. M. P. Michaelides and C. G. Panayiotou, "Fault tolerant event location estimation in sensor networks using binary data," in Tutorial and PhD Student Poster Workshop, *MedHocNet 2007*, Corfu, Greece, June 2007.

6. M. P. Michaelides and C. G. Panayiotou, "Exploiting spatial correlation for improving coverage in sensor networks," in *IEEE GLOBECOM 2007*, Nov. 2007.

7. M. P. Michaelides and C. G. Panayiotou, "Event detection using sensor networks: A case for a hybrid detector," in *46th IEEE Conference on Decision and Control*, Dec. 2007.

8. M. P. Michaelides and C. G. Panayiotou, "Subtract on Negative Add on Positive (SNAP) estimation algorithm for sensor networks," in *7th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT 07)*, Dec. 2007.

9. M. P. Michaelides and C. G. Panayiotou, "Fault tolerant event localization in sensor networks using binary data," in *American Control Conference 2008 (ACC 08)*, June 2008.

10. M. P. Michaelides and C. G. Panayiotou, "SNAP: Fault Tolerant Event Location Estimation in Sensor Networks using Binary Data," in *IEEE Transactions on Computers*, accepted Jan. 2009, to be published in 2009.

11. M. P. Michaelides and C. G. Panayiotou, "Fault Tolerant Maximum Likelihood Event Localization in Sensor Networks using Binary Data," in *IEEE Signal Processing Letters*, accepted Jan. 2009, to be published in 2009.

12. M. P. Michaelides and C. G. Panayiotou, "Improved Coverage in WSNs by Exploiting Spatial Correlation: The two sensor case," in *IEEE Transactions on Computers*, submitted Nov. 2008, under review.

13. M. P. Michaelides, C. Laoudias and C. G. Panayiotou, "Fault Tolerant Target Localization and Tracking in Wireless Sensor Networks using Binary Data," in *Wireless Communications and Mobile Computing*, submitted Dec. 2008, under review.

14. M. P. Michaelides, C. Laoudias and C. G. Panayiotou, “Fault Tolerant Detection and Tracking of Multiple Sources in WSNs using Binary Data,” in *48th IEEE Conference on Decision and Control*, submitted Feb. 2009, under review.

■