

STATISTICAL PROBLEMS IN WIRELESS SENSOR NETWORKS

by

Natallia V. Katenka

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2009

Doctoral Committee:

Professor George Michailidis, Co-Chair
Assistant Professor Elizaveta Levina, Co-Chair
Professor Alfred Hero III
Assistant Professor Stilian Stoev

© Natallia V. Katenka 2009
All Rights Reserved

To my Parents, Husband, and Sister

ACKNOWLEDGEMENTS

I owe a debt of gratitude to many people who contributed to my success in completing this dissertation. First of all, I have been privileged to have the direction and guidance of two excellent advisers, my co-chairs Professors Elizaveta Levina and George Michailidis. From the first day of my graduate studies in Michigan, Elizaveta has been very generous with her time and has provided very helpful advice on everything from the course schedule, the choice of my dissertation topic, the methodological design, the writing process to the choice of my future career. Her support and her ability to bring me back on track at difficult times have been a godsend. George likewise made invaluable contributions to the development of my ideas and the organization of my research. He has always motivated me to do my best work, provided timely feedback on each part of my dissertation and keen insight into the significance of my research. Both Liza and George introduced me to the joys of academic research not only by teaching me about the issues at hand, but also by encouraging my trips to a number of workshops and conferences and by sharing with me practical experiences they had accumulated through their professional careers and personal lives.

My other committee members, Professors Alfred Hero and Stilian Stoev, have also contributed insightful and helpful comments and suggestions. I am grateful to Alfred Hero for stimulating conversations that reassured me of the significance of my research during several workshops and seminars. I would also like to express my deep

gratitude to Professor Tailen Hsing for his help during one of my projects and my job application process. I would like to offer my sincere thanks to Brenda Gunderson, a person, who awakened and inspired my teaching abilities, and who always provided me with understanding and comforting talks. I am grateful to the Department of Statistics of the University of Michigan that supported me financially in various ways throughout the past five years. I am very thankful to our graduate program assistant Lu Ann Custer and department assistant Mary Ann King for their help with all my paperwork. Many thanks to all my friends who helped me to relieve the emotional storms and stress of graduate school, especially, to Bodhisattva Sen, Matthew Linn, Harsh Singhal, Herle McGowan, Amy Wagaman, Jason Goldstick, Sahar Zangeneh, and Ali Shojaie.

Finally, I owe much gratitude to my parents, Valjantsina and Uladzimir Katenka, for always believing in me and encouraging me to achieve my goals. I owe them for all they had sacrificed to give me a good education and a happy life. I will always feel in my heart their infinite love, joy, and faith in me. I am deeply grateful to my husband, Pavel Mikhno. His sincere love, understanding, and constant everyday support have been invaluable in helping me to focus on my academic pursuit. I would very much like to thank my only sister, Olga Marchenko, and her family. With her hard work, enthusiasm, compassion, and generosity, Olga has always been a prime example for me. She has always been there for me, and has never failed to do what she could to further my progress.

TABLE OF CONTENTS

DEDICATION	ii
ACKNOWLEDGEMENTS	iii
LIST OF FIGURES	vii
CHAPTER	
I. Overview	1
1.1 Wireless Sensor Network Design Issues	2
1.2 Wireless Sensor Network Data Fusion	5
1.2.1 Target Detection	6
1.2.2 Single Target Localization and Tracking	6
1.2.3 Multiple Target Localizations and Tracking	7
1.3 General Linear Data Fusion for Classification	8
II. Cost-Efficient Approach to Wireless Sensor Network Design	9
2.1 Introduction	9
2.2 Problem Formulation	12
2.3 Methods and Algorithms	14
2.3.1 The Coverage Constraint	14
2.3.2 The Connectivity Constraint	20
2.3.3 Joint Coverage and Connectivity Optimization	26
2.4 Extensions	31
III. Target Detection with Wireless Sensor Networks	33
3.1 Introduction	33
3.2 Methods and Algorithm	38
3.2.1 Decision Fusion for Target Detection	38
3.2.2 Local Vote Decision Fusion (LVDF): the algorithm	40
3.2.3 LVDF: threshold selection	42
3.2.4 LVDF: Central Limit Theory for threshold approximation	47
3.3 Performance Evaluation	50
3.4 Temporal Decision Fusion	57
3.5 Performance Evaluation for Temporal LVDF	59
IV. Localization and Tracking of a Single Target with Wireless Sensor Networks	62
4.1 Introduction	62
4.2 Methods and Algorithms	65
4.2.1 Localization from Original Decisions	66

4.2.2	Localization from LVDF Decisions	68
4.2.3	Hybrid Maximum Likelihood Estimates	70
4.2.4	Properties of Maximum Likelihood Estimates	71
4.3	Performance Evaluation	73
4.3.1	Detection Performance	74
4.3.2	Localization and Signal Estimation Accuracy	75
4.3.3	Starting Values	77
4.3.4	Robustness to Model Misspecification	81
4.3.5	Confidence Region Estimation	82
4.3.6	Computational Costs	83
4.4	Single Target Tracking	85
V.	Localization and Tracking of Multiple Targets with Wireless Sensor Networks	89
5.1	Introduction	89
5.2	Problem Formulation	93
5.3	Methods and Algorithms	94
5.3.1	Localization from Energy Readings	95
5.3.2	Localization from Binary Decisions	95
5.3.3	Localization from LVDF Decisions	96
5.3.4	Hybrid Maximum Likelihood Estimates	97
5.3.5	Estimating the Number of Targets	98
5.4	Multiple Target Tracking	98
5.5	Performance Evaluation	102
5.5.1	Choosing Starting Values for the Localization of Multiple Targets	103
5.5.2	Identifying the Location and Estimating the Number of Multiple Targets	107
5.5.3	Tracking of Multiple Targets	109
5.6	Applications	112
5.6.1	The NEST project	112
5.6.2	The ZebraNet Project	114
VI.	Local Data Fusion Framework for Classification in Wireless Sensor Networks and Beyond	118
6.1	Introduction	118
6.2	Methods and Algorithms	120
6.2.1	Problem Formulation	120
6.2.2	The Linear Discriminant Rule	121
6.2.3	A Local Data Fusion for Classification	122
6.3	Extensions	131
6.4	Performance Evaluation	133
6.4.1	Classification of binary textures	133
6.4.2	Handwritten Digit Recognition	138
VII.	Conclusions and Future Research	140
BIBLIOGRAPHY		142

LIST OF FIGURES

Figure

2.1	Average area not covered ϵ_S (solid line) with a 95% confidence band (dotted lines) obtained from simulations, together with the theoretical value (dash-dot line) as a function of the number of sensors n for $s = 0.1$ (left panel) and $s = 0.2$ (right panel), with $p = 1$. Results are averaged over 100 deployments on a unit square.	16
2.2	Number of sensors $n(s)$ as a function of sensing radius s for different values of the active state probability p with $\epsilon = 0.05$ (left panel), and for different values of ϵ with $p = 1$ (right panel).	16
2.3	Network cost function $n(s)C(s)$ (left panel) and its first derivative (right panel) with coefficients $c_0 = 1$ and $c_1 = 1000$ for different values of β_1 , with $p = 1$, $\epsilon = 0.05$, $S = [0.01, 0.035]$	18
2.4	Number of sensors $n(s)$ (left panel), sensor unit cost $C(s)$ (middle panel), and network cost function $n(s)C(s)$ (right panel) for collected and approximated data with $p = 1$, $\epsilon = 0.05$, $S = [0.0025, 0.3]$	19
2.5	Left panel: measure of fit (R^2) and estimated error variance (Error) for different regression models (“3” corresponds to $q(\delta)$ in (2.19)). Right panel: estimated values of $q(\delta)$ from simulations over 500 random deployments for different values of r with $p = 1$, together with the fitted curve.	23
2.6	Number of sensors n as a function of communication radius r for $\delta = 0.05$ (left panel) and $\delta = 0.1$ (right panel).	24
2.7	Approximated sensor unit cost $C(r)$ (left panel), corresponding number of sensors $n(r)$ (middle panel), and network cost $n(r)C(r)$ (right panel) as a function of connectivity range $r = [0.05, 0.35]$ with $p = 1$ and $\delta = 0.05$	26
2.8	The curve defined by $n_1(s) = n_2(r)$ with $\epsilon = \delta = 0.05$, $p = 1$	28
2.9	Number of sensors $n(s, r)$ (left panel), sensor cost function $C(s, r)$ (middle panel), and overall network cost $n(s, r)C(s, r)$ (right panel) with $c_0 = 93$, $c_1 = 104.329$, $c_2 = 12.2623$, $\beta_1 = 2.8765$, $\beta_2 = 1.6163$, $\epsilon = \delta = 0.05$, and $p = 1$. The optimal design is $s = 0.156$, $r = 0.348$, $n = 38$ and the optimal cost is $38 \times 346.4 = 13163.2$ dollars.	30
3.1	Left panel: Target signal generated by the model $S_i(v) = S_0 \exp(- s_i - v ^2/\eta^2)$ for a target at location $v = (0.5, 0.5)$ with $S_0 = 2$, $\eta = 0.1$; Right panel: Target energy contaminated by Gaussian noise of variance $\sigma^2 = 0.16$ (signal-to-noise ratio $S_0/\sigma = 5$).	34

3.2	Ordinary vs Local Vote decision fusion under a square grid design (top panels) and random deployment (bottom panels). The network is comprised of 100 sensors, with individual sensor false alarm probability $\gamma = 0.2$, system-wide false alarm probability $F = 0.1$ and a target located at the center of the monitored region R . The signal is generated by the model $S_i = S_0 \exp(-\ s_i - v\ ^2/\eta^2)$, with $S_0 = 2$, $\eta = 0.1$, and the measured energy is corrupted by Gaussian noise with $\sigma = 0.4$	41
3.3	Example of sensor neighborhoods with $M_i = 6$, $M_j = 5$ and $n_{ij} = 3$	42
3.4	Square (left panel), hexagonal (center panel) and diamond-shaped (right panel) neighborhoods on a regular grid.	45
3.5	Left panel: Root mean squared error (RMSE) of the normal approximation of the system's false alarm probability for grids with neighborhood sizes $M = 5, 9$, together with ODF, as a function of the grid size n (number of sensors $N = n^2$). The individual sensor's false alarm $\gamma = 0.2$, and the RMSE is computed over the range of $F = 0 \dots 0.5$. Right panel: A plot of the calculated false alarm probability quantiles against the theoretical ones of a standard normal distribution for a 25×25 grid, with $\gamma = 0.2$ and $M = 5$	52
3.6	Probability of target detection as a function of SNR (in dB) for models M1 (left panel) and M2 (right panel).	54
3.7	Probability of detection as a function of the system-wide false alarm probability F (ROC curve) for models M1 (left panel) and M2 (right panel).	55
3.8	Probability of detection as a function of SNR for signal decay parameter(left panel) and the system-wide false alarm probability F (ROC curve)(right panel) for $\eta = 0.08$	55
3.9	Number of positive decisions as a function of the sensor's false alarm α (left panel, with $\eta = 0.1$) and as a function of the signal decay parameter η (right panel, with $\alpha = 0.2$) for model M1 with $F = 0.1$ and $SNR = 7$	56
3.10	Target detection probability for various levels of signal decay η	57
3.11	Temporal LVDF probability of detection for different time slots and uncorrelated noise (left panel), and for a fixed time slot with different values of the correlation coefficient for correlated noise (right panel).	60
3.12	Temporal LVDF probability of detection for different time slots and uncorrelated noise (left panel), and for a fixed time slot with different values of the correlation coefficient for correlated noise (right panel).	60
3.13	Temporal LVDF probability of detection for slow (left panel) and fast moving targets (right panel) under a grid deployment.	61
4.1	Average distance from true target location v as a function of η for square grid deployment. (a) SNR = 2, model M1; (b) SNR = 2, model M2; (c) SNR = 5, model M1; (d) SNR = 5, model M2.	78
4.2	Root Mean Squared Error of estimating S_0 as a function of η for square grid deployment. (a) SNR = 2, model M1; (b) SNR = 2, model M2; (c) SNR = 5, model M1; (d) SNR = 5, model M2.	79

4.3	True model M2 misspecified as M1 with SNR=5. (a) The difference between average distances from true v for misspecified and true models; (b) The difference between RMSE of \hat{S}_0 for misspecified and true models.	82
4.4	True noise distribution t_3 misspecified as Gaussian, with SNR=5. (a) The difference between average distances from true v for misspecified and true models; (b) The difference between RMSE of \hat{S}_0 for misspecified and true models.	83
4.5	Distribution of iterations to convergence. (a) ML algorithms; (b) EM algorithms; (c) HEM algorithms.	85
4.6	Tracking a moving target: (a) Estimated target trajectory for a single realization; (b) Average distance from the true trajectory (over 100 replications).	87
4.7	Tracking a stationary target with evolving signal: (a) Estimated signal amplitude for a single realization; (b) Average RMSE of temporally fused S_0 (over 100 replications).	88
5.1	Left panel: The activation pattern of NEST sensors by a person traversing the monitored area. Right panel: The trajectory of a single zebra in the monitored area.	90
5.2	Target located at $x = [0.25, 0.25]$	104
5.3	Clustering results of initial (upper panels) and corrected (lower panel) decisions.	105
5.4	Four targets with $\eta = 0.1$ and $S_0 = 2$ located at $x_1 = (0.42, 0.26)$, $x_2 = (0.59, 0.69)$, $x_3 = (0.42, 0.75)$, $x_4 = (0.77, 0.95)$ in the monitored area R ; SNR=3. (a) Signal emitted by targets, (b) initial and (c) corrected decisions.	109
5.5	True trajectories (solid lines) and positions estimated by ML(Z) at each time point for three targets with SNR = 5. (a) The signal from the second target is briefly lost; (b) Two targets come close together and the third target briefly loses signal; (c) Another noise realization for (b).	110
5.6	(a) True trajectories; (b) Average estimated number of targets as a function of time.	111
5.7	Estimated and true trajectories for one, two, and three NEST targets.	113
5.8	Averaged distance from the true trajectory of a single moving target (Sc.1) as a function of λ	113
5.9	The recorded locations of the single zebra (a) and the four zebras (b) scaled and plotted on the unit square.	116
5.10	True and estimated by HEM(Z) coordinates $x(t)$ and $y(t)$ for the zebra.	116
5.11	True and estimated by HEM(Z) coordinates $x(t)$ and $y(t)$ for the four zebras.	117
6.1	(a) Weights and (b) the probability of correct classification as a function of p . ($d = 1$, $\pi_0/\pi_1 = 1$, $\sigma = 1$, $\mu_1 = 1$, $\mu_0 = 0$)	130
6.2	Parameters $\{\alpha, \beta, \gamma, \delta, \epsilon\}$ corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.	134

6.3	True labels Y_{ij} corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.	134
6.4	Simulated X_{ij} corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.	135
6.5	Labels predicted via LDA for binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.	135
6.6	Labels predicted via LDA-LF corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.	136
6.7	(a) Estimated average difference $(P_z - P_x)$ as a function of SNR .(b) Universal neighborhood 4×2	137

CHAPTER I

Overview

Wireless Sensor Networks (WSN) is a new technology which allows monitoring natural phenomena in space and time. Originally, wireless sensor networks were used for military applications, but currently they are employed in a variety of civil applications, including surveillance, chemical and biological agent monitoring, security in critical infrastructures, environmental and habitat monitoring, etc. Some of these application areas cover a broad range of objectives. For example, surveillance applications extend from intruders' detection and tracking to in-home monitoring of elderly patients [102]. Environmental applications include land monitoring [75], recognition of amphibian populations [46] and underwater tsunami and seaquake detection [5]. Other recent applications of WSNs include identification of chemical, biological, radiological, nuclear and explosive phenomena¹, and infrastructure monitoring [123]. Technological constraints imposed by the nature of WSN together with their increased importance in a number of applications have created a fertile ground for research. There is active involvement from diverse research communities, including electrical engineering and computer science, materials science and manufacturing, and, more recently, statistics.

¹Sensornet: Nationwide detection and assessment of chemical, biological, radiological, nuclear and explosive threats, "http://www.sensornet.gov"

Wireless sensor networks are built from a large number of autonomously powered devices (sensors) capable of sensing signals from their surrounding environment, with limited communication, computing and storage capabilities. Each sensor consists of four basic components: a power unit which supports all sensor operations; a sensing unit which collects environmental measurements and translates the analog signal of the observed phenomenon (energy) into a digital signal via an analog-to-digital converter (ADC); a processing unit, which stores and pre-processes the digital signal; and a transceiver unit, which is responsible for all sensor communications. In a WSN, sensors are linked by a wireless medium – radio, infrared or optical. The transmitted data need to be routed to a 'sink' node, also known as task manager, or data fusion center. The network layer controls the routing protocol between the sensor and the sink nodes. Power efficiency is of paramount importance, and specially designed protocols have been developed for WSNs (more details can be found in [4]). Processing and storage capabilities of WSNs range from devices capable of carrying out computational tasks to simple sensing devices; the former are able to obtain measurements, process them and even store some 'sufficient' statistics for a limited time period, while the latter are constrained to getting data and immediately transmitting them. Finally, in many cases the location of the sensors is unknown and needs to be estimated by the network [49]. When the knowledge of the sensor locations is crucial, an additional location finding unit (for instance, GPS) may be installed. However, it may significantly increase the cost of the network.

1.1 Wireless Sensor Network Design Issues

Technological constraints of sensors coupled with the application area under consideration determine to a large extent the deployment strategy for the WSN. For

example, in industrial applications [59] the sensors are deployed at specific locations of interest; the same holds true for structural monitoring [124]. In various environmental applications the sensors are often deployed in a one-dimensional array pattern (see Soil Pylon Development and Palmdale Test Bed Deployments²) or on a fairly regular grid.

However, in many situations deterministic deployment is neither feasible nor practical; e.g., when the region monitored by the WSN is not easily accessible. In such cases, deployment mechanisms are often equivalent to a random positioning of the sensors. For example, a non-accessible area, such as a contaminated site or a battlefield, requires aerial deployment, which leads to a random assignment of sensor locations [4]. Sensors in underwater networks are usually anchored to the sea floor and attached to a floating buoy that regulates their position [5]. The imprecision of this mechanism can be well approximated by a random deployment. Another example is embedding sensors in materials for measuring their properties, such as concrete, ablative materials (see the MSRS developed by NASA³) or polymer structures [66].

As a rule, a sensor network under any deployment should satisfy two fundamental constraints: coverage (all or most of the region of interest is within the sensing range of at least one sensor) and connectivity (each sensor can communicate with any other sensor either directly or by relaying information through its neighbors). These constraints are critical for the wireless sensor network to be able to accomplish its task. In regular deterministic deployments it is easier to provide the necessary coverage of the phenomenon under consideration and also to guarantee connectivity amongst the sensors; on the other hand, in a random deployment scenario it is more

²Soil Pylon Development and Palmdale Test Bed Deployments,
<http://research.cens.ucla.edu/areas/2007/Contaminant/projects.htm>.

³Embedded Sensors for Measuring Surface Regression, Stennis Space Center, Mississippi, NASA Tech. Briefs, Electronics and Computers, 2006.

difficult to satisfy these constraints.

In Chapter II we describe a general flexible approach to design of wireless sensor networks under the random deployment mechanism (see also [53]). Although sensors are relatively cheap, the overall cost may be large due to the size of the network; hence, the main objective is to minimize the overall network cost, while enforcing the coverage and connectivity constraints. The cost of sensing and communications is incorporated into the design of the network; we also allow for unreliable sensors. In the proposed approach, cost is treated generically and can correspond to either a fixed acquisition cost, or an operational cost or a combination of both. Our approach can be used as part of any feasibility study during the planning stages for the deployment of a wireless sensor network, when decisions about its capabilities and cost are considered. The technical contribution is the derivation of a new simple bound on the probability of a network being connected, which exhibits a very good performance in simulations shown to be better suited for network design studies than other existing bounds.

Additional sensor network design issues include the estimation of the sensor locations, lossless communication protocols, synchronized transmissions to other sensors and the center node, network size-scalability, network and sensors reliability, etc. Problems related to these issues are out of the scope of this work, but they have been of a particular interest of the engineering and computer science research community for the last decade (for comprehensive review see [6], [65], [113] and references therein).

In what follows next, we assume that all communication and networking issues have been settled in advance and we focus only on the collaborative signal processing task at hand.

1.2 Wireless Sensor Network Data Fusion

Detection, identification and tracking of spatial phenomena are important tasks in various environmental and infrastructure applications. Typically, the sensors measure the phenomenon under consideration at discrete points in space and time and the task of the network is to integrate (fuse) the available data in order to estimate and track the parameters of interest; for example, in surveillance monitoring [34], the task is to detect an intrusion and follow its path; in habitat monitoring [75], to identify and track herds; in environmental monitoring [87], to estimate soil moisture levels [13], dispersion of pollutants [56], etc.

In our setting, we assume that each sensor records a signal (temperature, vibration, etc) emitted from a target, makes a decision about the target's presence/absence, and then transmits either the signal or the decision to the fusion center, which makes a final situational assessment. Sensors themselves have limited capabilities, and data fusion from many sensors enhances the performance, especially if the individual sensors are only providing single bit (binary) results. Transmission of binary decisions instead of signals offers savings in communication costs, but binary decisions are unreliable in noisy environments. Our major contribution to this problem is an algorithm to improve the reliability of binary decisions, the Local Vote Decision Fusion (LVDF). In LVDF, sensors correct their initial decisions by first consulting the neighboring sensors. This results in correlated decisions, which presents technical challenges. Using the LVDF, we develop new data fusion algorithms for target detection (making a network-level decision about the presence of a target), target localization (estimating the target's position), and target tracking (estimating trajectories of multiple moving targets over time).

1.2.1 Target Detection

We investigate the problem of target detection by a wireless sensor network in Chapter III. We propose and describe an LVDF-based detection framework that guarantees a given false-alarm level for the network. We show that this framework performs significantly better than existing methods based on binary data, especially in a noisy environment. The critical step in this work is adapting a central limit theorem for correlated random fields and deriving an analytical approximation for the network-wide decision threshold. Our detection approach makes no assumptions about the target signal model or background noise distribution, and can be directly applied to multiple target detection. We also extend this framework to temporal fusion, where information becomes available over time.

1.2.2 Single Target Localization and Tracking

In Chapter IV, we discuss different approaches to the problem of a single target localization and signal diagnostics by a wireless sensor network. We develop a pseudo-likelihood based method for estimating the target's location and signal magnitude for the proposed LVDF mechanism. Two variants – direct optimization of the likelihood function and an expectation-maximization algorithm – are developed and compared both for the original and updated decisions. Uncertainty assessments of the parameters of interest are obtained via a bootstrap technique. Further, numerical results indicate that the LVDF-based algorithms outperform even the maximum likelihood estimate based on the actual energies in a low signal-to-noise environment, and outperform localization based on uncorrected decisions in every setting.

As an extension to the proposed methods above and assuming that energy readings from sensors that made positive decisions are available, we develop hybrid (pseudo-)

maximum likelihood and expectation-maximization algorithms. Simulation results show that the proposed framework significantly improves the accuracy in target location estimation and signal magnitude estimation. In the simulations two different signal models were used; robustness to model and noise distribution misspecification is also examined. Finally, extensions to tracking of a single moving target are considered.

1.2.3 Multiple Target Localizations and Tracking

The problem of localizing and tracking multiple targets introduces several new challenges, such as estimating their number, adapting to targets appearing and disappearing over time, and enforcing some smoothness in target trajectories over time. There has been a substantial amount of work on multiple target localization and tracking over the past decade. However, most of the proposed methods use a parametric approach that assumes either the number of targets known or the target trajectories or both. In order to address all the challenges above, we utilize model selection methods, clustering techniques, matching algorithms, and penalized likelihood optimization.

In Chapter V we develop the LVDF-based multiple tracking framework which also allows for sensor failures, targets appearing and disappearing over time, and complex target trajectories. We apply our framework to two case studies – an experiment involving tracking people and a project of tracking zebras, and in terms of estimation accuracy show that our tracking approach using binary decisions exhibits a competitive performance even compared to maximum likelihood estimation based on full energy measurements.

1.3 General Linear Data Fusion for Classification

Motivated by the success of the LVDF algorithm in WSNs, in Chapter VI we introduce a general data fusion framework for classification purposes. The set-up involves a collection of local classifiers that can consult their neighbors (in some suitably defined metric) and take their measurements or decisions into account. We assume that each classifier knows the probability that its neighbors are observing the same class label as they are, which can be modeled through some local covariance model (e.g., spatial), and derive optimal weights for fusing data from such correlated classifiers. The obvious application of this approach is classifying spatial data, but the framework can be equally well applied to the more general problem of fusing data from multiple sources with varying degrees of reliability, which has applications in security. Preliminary results on simulated Markov random fields and on handwritten digit data show significantly better classification performance compared to methods based on non-fused data.

In summary, Chapters III - V present a complete multi-target detection-localization-tracking framework for wireless sensor network based on corrected binary decisions; and Chapter VI presents a general data fusion algorithm for correlated classifiers. Detailed conclusions and future research problems are discussed in Chapter VII.

CHAPTER II

Cost-Efficient Approach to Wireless Sensor Network Design

2.1 Introduction

In this chapter we present a general flexible approach for the design of wireless sensor networks under the random deployment mechanism. The random deployment scenario has received a substantial amount of attention in the literature. The two fundamental constraints such a network has to satisfy are coverage (all or most of the region of interest is within the sensing range of at least one sensor) and connectivity (each sensor can communicate with any other sensor either directly or by relaying information through its neighbors). These constraints are critical for the wireless sensor network to be able to accomplish its task.

It turns out that coverage is easier to address, since for a spatial random marked point process the coverage of an area by sets centered at the random points is comprehensively studied in [42]. Additional issues specific to wireless sensor networks are discussed in [47], [61], and [68], and we elaborate on this point when we examine the coverage constraint in Section 2.3.1. The important point from the design point of view is that the coverage constraint has an explicit analytic form and is easy to solve as a function of the number of sensors.

On the other hand, the connectivity constraint turns out to be harder to analyze.

The general question of the number of connected components of a random graph is covered, for example, in [88], but all general connectivity results are asymptotic and provide little guidance for network design. In the context of wireless sensor networks this issue has been addressed in a number of papers (see [39], [98], [114], [11] and references therein). Typically, asymptotic bounds on the probability of the network being connected are obtained, but they either involve unknown functions [39] or are not tight enough [11], as we show below in Section 2.3.2. Here we will develop a new simple approximation to the probability of the network being connected for a given number of sensors n under random deployment, which, by utilizing an empirically determined constant, provides an explicit formula useful for network design purposes. Finally, in a number of studies both coverage and connectivity have been studied together ([118], [81], [121], [98]), again with the focus on deriving bounds. An important result for design purposes was proved in [118]: if the communications range of the sensors is at least twice their sensing range, then coverage implies connectivity, so only the coverage constraint needs to be enforced. Scheduling protocols designed to maintain both coverage and connectivity have been considered in [74], [69], [81], and [121].

Our goal here is to cast the network design problem as an optimization problem, where we are interested in minimizing the overall cost of the network which is affected by its sensing and communication capabilities, while maintaining the desired coverage and connectivity. Most currently available commercial products (e.g. Crossbow Technology wireless motes¹, MicroStrain's wireless measurement systems², and HOBOnode wireless data loggers³) allow for specification of communication and sensing ranges, albeit with a limited range of selections. We assume that we can

¹Crossbow Technology, <http://www.xbow.com>.

²MicroStrain, Inc., <http://www.microstrain.com/>.

³ONSET, <http://www.onsetcomp.com/>.

choose both the sensing radius and the communications one (either from a set of commercially available options, or from a larger set of available sensing and transmission options if the sensors are yet to be constructed), but there is a cost associated with each option. We treat the cost generically, so it can represent either the fixed acquisition cost, or the operational (energy) cost, or a combination of both. Further, we incorporate the possibility of unreliable sensors: we assume that each sensor is active and performs its task with probability $p \in (0, 1)$, *independent* of its other characteristics, and fails to do so with probability $1 - p$. In the design context, the concept of active sensors is also rather generic: a sensor may not be active because of hardware failure in which case p captures a priori reliability, or because the network protocol has put it to sleep to save on energy, in which case p is determined by the scheduling protocol. The issue of active sensors has briefly been addressed in [74] in the context of minimizing the power consumption of each individual sensor in the network, while ensuring the necessary coverage and connectivity, but not in terms of overall network cost.

Different aspects of network design through optimization of different cost functions have been considered in various studies. A design based on the minimal covering problem was studied by [8] and [121], but that precludes random deployment of sensors. In [80], a network design that minimizes an energy-based cost was developed under a lifetime constraint. Other cost functions include network search cost [3] and target localization error bound [117]. However, these cost functions do not incorporate the sensor characteristics themselves, such as the sensing and communications capabilities. Our approach provides a general flexible framework for network design, where different cost functions determined by the application are easy to incorporate. At the technical level, explicit expressions for the coverage and connectivity con-

straints are required. Hence, we provide a new simple connectivity bound which is shown to perform very well in simulations. The posited optimization problem can be part of any feasibility study during the planning stages for the design of a WSN.

The remainder of the Chapter is organized as follows. Section 5.2 formulates the problem as a constrained optimization problem. Section 2.3.1 focuses on the coverage constraint alone, while Section 2.3.2 focuses on the connectivity constraint and presents the new bound. Section 2.3.3 studies network design when enforcing both coverage and connectivity constraints. Section 2.4 discusses potential extensions of proposed framework and some concluding remarks are given in Section ??.

2.2 Problem Formulation

Suppose n sensors will be deployed at random within a two-dimensional monitoring region X which without loss of generality corresponds to the unit square. Further, suppose we can choose the sensing range s and communications range r from a set of feasible values $D \subset (0, \infty) \times (0, \infty)$, and they are associated with a cost function $C(s, r) \geq 0$, which denotes the cost per sensor. Different applications will have different forms of the set D . If the sensors are constructed by combining ready-made sensing and transmission components, there may be I fixed choices for the sensing range, with $S = \{s_i, i = 1, \dots, I\}$, and J choices for the communications range with $R = \{r_j, j = 1, \dots, J\}$, and $D = S \times R$. If the sensors are chosen from a fixed number of commercially available models, some combinations of s_i and r_j in this Cartesian product may not be possible. If the sensors can be constructed with any sensing or communications radii s and r in a certain range, which will be assumed later on, the sets S and R can be intervals of feasible values, with $D = S \times R$. We further assume that each sensor performs its task with probability $p \in (0, 1)$ and

fails to do so with probability $1 - p$. Our objective is to determine the number n and the type (in terms of the values of s and r) of sensors to be *randomly* deployed over X to minimize the total network cost,

$$(2.1) \quad \min_{n, (s, r) \in D} nC(s, r)$$

subject to:

- (a) **Coverage constraint:** the expected fraction of the region area covered exceeds a prespecified threshold $1 - \epsilon$, $\epsilon > 0$; and
- (b) **Connectivity constraint:** the probability that there is a communication path between any two sensors exceeds another prespecified threshold $1 - \delta$, $\delta > 0$.

We further assume that the individual sensor's cost function $C(s, r)$ is a *positive, continuous, non-decreasing function* in both arguments; i.e.

$$C(s, r) \geq 0, \quad \partial C(s, r)/\partial s \geq 0, \quad \partial C(s, r)/\partial r \geq 0.$$

This assumption is supported by empirical evidence, since sensors with more capabilities are more expensive.

Notice that the lower bound on the number of sensors n will be determined by the constraints, and the objective function is strictly increasing in n . Thus, if for each combination of values of s and r , we define

$$n(s, r) = \min\{n : \text{constraints (a) and (b) are satisfied}\},$$

then the optimization problem reduces to

$$(2.2) \quad \min_{(s, r) \in D} n(s, r)C(s, r).$$

Note that the random deployment mechanism of the sensors determines the exact expressions for the coverage and connectivity constraints. Further, the nature of the

optimization problem is determined by the nature of the set D . If there is a finite discrete number of options for the sensors' characteristics, the above optimization problem is a combinatorial one. Obviously, for a small set D an exhaustive search strategy is adequate, but for larger sets it becomes impractical. We consider the continuous version of the problem, where the sensing and communications radii can take any value in an interval; i.e., $S = [S_L, S_U]$, $R = [R_L, R_U]$, where $S_L > 0$ and $R_L > 0$, and the sensor design space D is given by the Cartesian product $[S_L, S_U] \times [R_L, R_U]$. This can be viewed as a *relaxation* of the discrete optimization problem.

Notice that if the objective function is convex in both s and r , a global minimum exists and can be determined by solving for the point $(\tilde{s}, \tilde{r}) \in D$. Obviously, if such a point (\tilde{s}, \tilde{r}) is not feasible, then the solution is given by a point on the boundary of the design space D . Ultimately, the solution is determined by the form of functions $n(s, r)$ and $C(s, r)$. Next, we investigate the shape of $n(s, r)$ comprehensively by first studying separately the coverage (Section 2.3.1) and connectivity (Section 2.3.2) constraints, and then apply our findings to the joint problem (Section 2.3.3).

2.3 Methods and Algorithms

2.3.1 The Coverage Constraint

Focusing solely on the coverage constraint first, the problem becomes

$$\min_{n, s \in S} nC(s), \quad \text{s.t.} \quad \mathbb{E}(\text{area not covered}) \leq \epsilon,$$

where $C(s)$ is a positive, continuous, non-decreasing function of s . The monitored area X is considered covered by the WSN if all points in it are within the sensing radius of at least one sensor.

The problem of coverage in WSN has received a lot of attention in the literature

from various perspectives. In [68] and [79] it is used as a 'quality-of-service' measure for the WSN to detect and track targets within its monitored region under a random deployment mechanism. An enhanced measure of coverage, called k -coverage, is discussed in [47], [61], and [45]. This introduction of redundancy by requiring each point within X to be covered by at least k sensors leads to improved detection, target localization and tracking. Further, an extension of k -coverage in the presence of active/inactive sensors is examined in [61]. In [109], the problem of selecting and scheduling when the sensors should become active/inactive while preserving the k -coverage property is addressed. A different scheduling protocol for the latter problem is discussed in [100]. Finally, results addressing detection and sensor scheduling issues under *partial* coverage of the monitored region were provided in [70] and in [97].

Recall that we formulate the design problem under the assumption that the sensors can be in an active or inactive state, induced either by hardware failure or as a result of a scheduling protocol. Let p denote the probability that a sensor is in the active state, and assume it is the same for all sensors. Then, the probability that a target/event at location $x \in X$ is detected by a single randomly deployed sensor is $p\pi s^2$. Let \tilde{X} denote the subset of X not covered by the network, and write $|\tilde{X}|$ for the area of \tilde{X} . A result from the theory of coverage processes (page 128 in [42]) gives that, for a random deployment of n sensors,

$$(2.3) \quad \mathbb{E}(|\tilde{X}|) = (1 - p\pi s^2/|X|)^n |X| .$$

Since it is known that $\text{Var}(|\tilde{X}|)$ converges to 0 as a function of n sufficiently fast [42], we can assume that the *actual* area not covered by the network deviates little from the above expression for expected area not covered. Figure 2.1 shows that the expected area *not covered* $E(|\tilde{X}|)$ calculated using (2.3), falls within the 95% confidence bounds based on simulations for different network sizes and values of the

sensing range s .

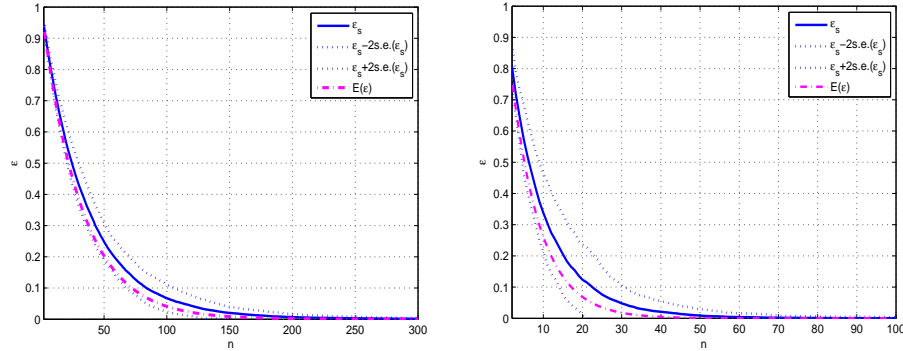


Figure 2.1: Average area not covered ϵ_s (solid line) with a 95% confidence band (dotted lines) obtained from simulations, together with the theoretical value (dash-dot line) as a function of the number of sensors n for $s = 0.1$ (left panel) and $s = 0.2$ (right panel), with $p = 1$. Results are averaged over 100 deployments on a unit square.

Assuming $|X| = 1$, the smallest n that satisfies the constraint $\mathbb{E}(|\tilde{X}|) \leq \epsilon$ is

$$(2.4) \quad n(s) = \frac{\log(\epsilon)}{\log(1 - p\pi s^2)}.$$

For illustration purposes, the function $n(s)$ (the number of sensors for achieving the required coverage) is plotted in Figure 2.2 for several values of p and ϵ . It can be seen that for small values of s the effect of both p and ϵ is rather small.

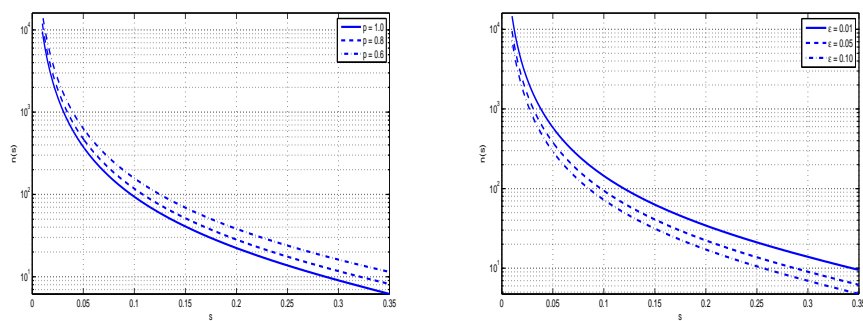


Figure 2.2: Number of sensors $n(s)$ as a function of sensing radius s for different values of the active state probability p with $\epsilon = 0.05$ (left panel), and for different values of ϵ with $p = 1$ (right panel).

As discussed in Section 5.2, the corresponding optimization problem takes the

form

$$\min_{s \in S} n(s)C(s).$$

The conditions for an interior point minimum to exist at point $\tilde{s} \in (S_L, S_U)$ are given by

$$(2.5) \quad \begin{cases} n'(\tilde{s})C(\tilde{s}) + n(\tilde{s})C'(\tilde{s}) = 0 \\ n''(\tilde{s})C(\tilde{s}) + 2n'(\tilde{s})C'(\tilde{s}) + C''(\tilde{s})n(\tilde{s}) > 0, \end{cases}$$

where

$$(2.6) \quad n'(s) = 2 \frac{\log(\epsilon) \pi s p}{(\log(1 - \pi s^2 p_s))^2 (1 - \pi s^2 p)}$$

$$(2.7) \quad n''(s) = 2 \frac{\log(\epsilon) \pi s p [4\pi p s^2 + (1 + \pi s^2 p) \log(1 - \pi s^2 p)]}{(\log(1 - \pi s^2 p))^3 (1 - \pi s^2 p)^2}$$

If such a point is not feasible ($\tilde{s} \notin S$) then the solution would occur either at S_L or at S_U .

To illustrate, consider the following cost function [110]:

$$(2.8) \quad C(s) = c_0(1 + c_1 s^{\beta_1}), \quad c_0, c_1, \beta_1 > 0.$$

This is an example of an additive cost function, where c_0 represents the *fixed* cost of a sensor independent of its sensing capabilities, while the second term captures the *variable* cost which is an increasing function of the sensing radius. For the optimization problem, we can take $c_0 = 1$ without loss of generality. Then, the network-wide cost and its first derivative are given by

$$\begin{aligned} n(s)C(s) &= \frac{\log(\epsilon)}{\log(1 - \pi s^2 p)} (1 + c_1 s^{\beta_1}), \\ ((n(s)C(s)))' &= \frac{\log(\epsilon)}{s \log(1 - \pi s^2 p)} (-b(s) + c_1(\beta_1 - b(s))s^{\beta_1}), \end{aligned}$$

where

$$(2.9) \quad b(s) = \frac{-2s^2 \pi p}{(1 - \pi s^2 p) \log(1 - \pi s^2 p)}.$$

The existence of an interior solution $\tilde{s} \in (S_L, S_U)$ depends on the specific choices of the parameters c_1 and β_1 . For $0 \leq \beta_1 < 2$, the overall network cost $n(s)C(s)$ is a decreasing function of s , so the optimal design is achieved for $s = S_U$. For $\beta_1 \geq 2$, the convexity and interior vs. boundary solution depend on c_1 . The network cost function is plotted in Figure 2.3 for several values of β_1 . It can be seen that sometimes there is more than one interior extremum point, and sometimes the solutions are on the boundary.

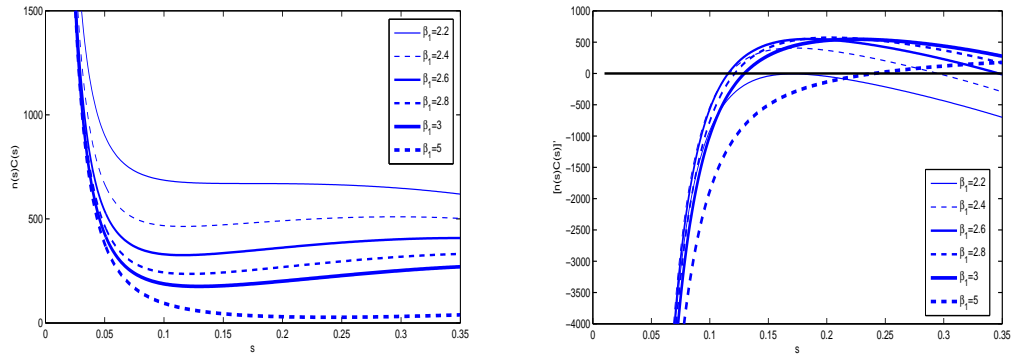


Figure 2.3: Network cost function $n(s)C(s)$ (left panel) and its first derivative (right panel) with coefficients $c_0 = 1$ and $c_1 = 1000$ for different values of β_1 , with $p = 1$, $\epsilon = 0.05$, $S = [0.01, 0.035]$.

The optimal solutions (the size of the WSN n and the corresponding sensing radius s) for $S = [0.01, 0.035]$ and various choices of c_1 and β_1 are given in Table 2.1.

Table 2.1: The optimal sensing radius s (rounded to 3 significant digits) and the number of sensors n for different values of c_1 and β_1 with $p = 1$ and $\epsilon = 0.05$.

β_1	2.2		2.4		2.6		2.8		3		5	
	s	n	s	n	s	n	s	n	s	n	s	n
100	.350	7	.350	7	.350	7	.350	7	.350	7	.350	7
500	.350	7	.350	7	.156	38	.159	37	.167	33	.280	11
1e3	.350	7	.116	69	.115	70	.121	64	.129	56	.240	16
5e3	.061	254	.057	293	.061	258	.067	213	.074	172	.171	32
1e4	.044	492	.042	529	.046	445	.052	352	.059	258	.148	43
1e5	.015	4112	.016	3652	.019	2644	.023	1843	.027	1290	.093	110

Example: This illustrative example is based on real data obtained for a sample of 35 low-power acoustic sensors (microphones) obtained from Digi-Key Corporation⁴

⁴Digi-Key Corporation, <http://www.digikey.com/>.

– an industry leader in the distribution of electronic components. Their sensing radii range from 2.5 meters to 300 meters ($s = [0.0025, 0.3]$), with a total of nine types. For each type, the unit cost was computed as the average cost of the sensors with the same sensing radius (Figure 2.4, middle panel). In addition, the number of sensors required for monitoring an area of size $1\text{km} \times 1\text{km}$ is shown in the left panel of Figure 2.4, together with the corresponding overall network cost in the right panel of Figure 2.4. It can be seen that the network cost is not convex in the sensing range. The global minimum is achieved at a boundary point and corresponds to \$675.5 (10 sensors at \$67.55 each with a sensing range of 300m), while an interior local minimum exists at \$1200.68 (52 sensors at \$23.09 each with a sensing range of 134m). Further, we can approximate the cost function $C(s)$ for the sensing range between 37.8 and 157.4m by the cost function of the form (2.8) with coefficients $c_0 = 3$, $c_1 = 3.2342 \times 10^3$, and $\beta_1 = 2.876$. The number of sensors required to satisfy the coverage constraint and the network cost function based on this approximation are also shown in Figure 2.4. In this case, an interior optimum network cost of \$1442.82 is achieved with the deployment of 139 sensors (at \$10.38 each) with a sensing range of 82.4m.

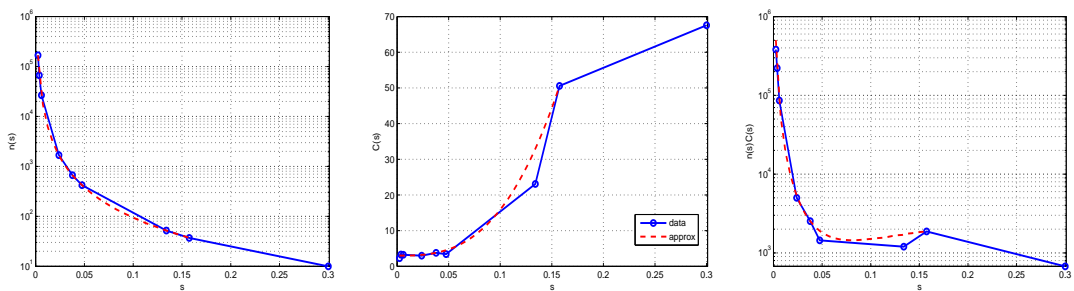


Figure 2.4: Number of sensors $n(s)$ (left panel), sensor unit cost $C(s)$ (middle panel), and network cost function $n(s)C(s)$ (right panel) for collected and approximated data with $p = 1$, $\epsilon = 0.05$, $S = [0.0025, 0.3]$.

2.3.2 The Connectivity Constraint

Analogously to the optimization problem subject to the coverage constraint alone, when the connectivity constraint is considered alone, the problem becomes

$$\min_{n,r \in \mathcal{R}} nC(r), \quad \text{s.t. } \mathbb{P}(\text{network is not connected}) \leq \delta,$$

where $C(r)$ is a positive, continuous, non-decreasing function of r . The WSN is considered to be fully connected if all the sensors can communicate and exchange information with each other through some path of intermediary nodes. Formally, let $G(n, r)$ denote the graph with node set corresponding to the sensors and edges between all pairs of nodes at the distance less than or equal to r . Then the network is considered connected if graph G is connected.

The fundamental problem of deriving conditions under which the WSN is essentially (with high probability) connected has been extensively studied in the literature. Next, we review some of the main results that could help us formulate the connectivity constraint. [39] studied the WSN randomly deployed on the unit circle and examined the probability of full connectivity as the size of the network n goes to infinity. The problem is formulated as finding the *minimum* communications radius $r(n)$ that ensures connectivity for a given number of sensors n , but since we assume that the cost function is non-decreasing in r , this formulation is obviously equivalent to ours. Specifically, it was shown that if

$$(2.10) \quad \pi r^2(n) = \frac{\log n + v(n)}{n},$$

then the probability that the underlying WSN is connected tends to 1 if and only if $v(n) \rightarrow \infty$ as $n \rightarrow \infty$. Any function $v(n)$ with this property would work, but from a practical standpoint this limits the applicability of the result. However, the

following bound derived in [39] as an intermediate step proves more useful. Let $P_d(n, r(n))$ denote the probability that the network is disconnected. Assume that $\lim_{n \rightarrow \infty} v(n) \equiv v$ and (2.10) holds. Then, the following holds (Theorem 3.1 in [39]):

$$(2.11) \quad \limsup_{n \rightarrow \infty} P_d(n, r(n)) \leq 4 \exp(-v).$$

Our goal is to derive an explicit relationship between n and r that would imply

$$(2.12) \quad P_d(n, r(n)) \leq \delta,$$

i.e. the network is disconnected with probability at most δ . Clearly, there exists the minimum $n(r)$ such that (2.12) holds for all $n \geq n(r)$, and it satisfies $P_d(n(r), r) = \delta$. Using (2.10), the bound (2.11), and setting $\delta = 4 \exp(-v)$ we get that the minimum sufficient number of sensors $n_{GK}(r)$ (GK stands for Gupta and Kumar) is the solution of

$$(2.13) \quad \pi r^2 n_{GK}(r) - \log(n_{GK}(r)) = -\log(\delta/4).$$

Equivalently, for a network of size n the connectivity radius should be at least

$$(2.14) \quad r_{GK}(n) = \left(\frac{\log n - \log(\delta/4)}{\pi n} \right)^{1/2}.$$

Another related result was obtained in [11], where the goal was to calculate the probability that none of the nodes in a randomly deployed WSN are *isolated*. Let A denote the event that there exists at least one sensor with no neighbors within the connectivity radius r . Then,

$$(2.15) \quad P(A) = (1 - \exp(-n \pi r^2))^n,$$

However, the fact no isolated nodes exist does not imply the whole network is connected (there could be two separate connected components). Therefore, the probability of the network being disconnected P_d is greater than the probability $P(A)$

that the network has isolated nodes. The original proposal in [11] was to use the probability $P(A)$ as an approximation to P_d , even though it is a lower bound and will yield an insufficient number of sensors to ensure full connectivity. We include this approximation here for comparison purposes. For given δ and r , the minimum number of nodes $n_B(r)$ (Bettstetter's bound in [11]) required is the solution of

$$(2.16) \quad n_B(r) \log(1 - \exp(-n_B(r) \pi r^2)) = \log \delta .$$

Similarly, if the network size n is fixed, then for a given $\delta > 0$, the communication radius r can be obtained explicitly and should be at least

$$(2.17) \quad r_B(n) = \left(-\frac{\log(1 - (1 - \delta)^{1/n})}{\pi n} \right)^{1/2} .$$

Simulation results below show that both solutions, (2.13) and (2.16), are lower bounds on the true $n(r)$ required for connectivity. Thus using them for design cannot guarantee desired network properties. Here, we propose an empirical and more precise way of identifying $n(r)$ as a function of r and δ . Using (2.13) as a starting point, we omit the slowly varying term $\log n(r)$, and instead absorb it into an unknown function on the right hand side that depends only on δ , $q(\delta)$. This results in a simple estimate for $n(r)$,

$$(2.18) \quad n(r) = \frac{q(\delta)}{\pi r^2} .$$

Further, we approximate the function q by its first order power expansion as

$$(2.19) \quad q(\delta) = \gamma_0 \delta^{-1} + \gamma_1 + \gamma_2 \delta^1 ,$$

The coefficients γ_0 , γ_1 , γ_2 were obtained from the least squares fit to simulation data. A large number (500) of random deployments were generated and the minimum number of sensors calculated for the network to be disconnected with probability δ for

a given r . From these data, the coefficient values were estimated as $\gamma_0 = 0.0397$, $\gamma_1 = 14.1208$, $\gamma_2 = -9.4352$, with a very satisfactory measure of fit ($R^2 = .975$). The fitted curve is shown in Figure 2.5 as a function of δ , along with simulated results for selected values of r , with fixed $p = 1$. One can see that the variation across r is small relative to the variability as a function of δ , which justifies modeling $q(\delta)$ as a function of δ only. Further, compared to simpler models ("1": $\gamma_1 + \gamma_2\delta^1$, "2": $\gamma_0\delta^{-1} + \gamma_1$), the chosen one ("3") shows a significant improvement in terms of fit and estimated error variance (see Figure 2.5, left panel). On the other hand, more complicated models that include second ("4" - "6") and third order terms ("7" - "9"), do not exhibit any noticeable gains in terms of fit or estimated error variance.

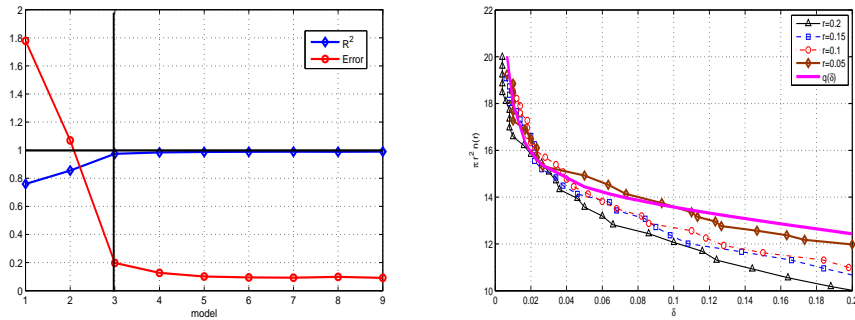


Figure 2.5: Left panel: measure of fit (R^2) and estimated error variance (Error) for different regression models ("3" corresponds to $q(\delta)$ in (2.19)). Right panel: estimated values of $q(\delta)$ from simulations over 500 random deployments for different values of r with $p = 1$, together with the fitted curve.

Figure 2.6 shows the proposed estimates of n as a function of r for two values of δ (0.05 and 0.1) and fixed $p = 1$. The options compared are (2.13) (labeled GK), (2.16) (labeled B), our empirical formula (2.18) (labeled Q) and the simulation result (Sim), which can be taken as the "ground truth". The results of the simulations strongly indicate that the values of n calculated from (2.13) and (2.16) are only lower bounds, and thus should not be used for design purposes, since they would lead to an under-provisioned network. Our empirical value, on the other hand, either agrees very

well with simulation or is slightly more conservative (estimating a larger n than necessary). Because a connected network is required for the successful operation of the WSN, a conservative estimate is a natural choice over an overly optimistic one.

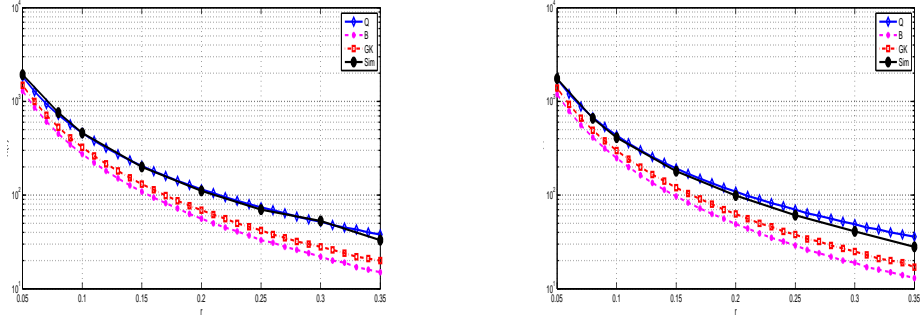


Figure 2.6: Number of sensors n as a function of communication radius r for $\delta = 0.05$ (left panel) and $\delta = 0.1$ (right panel).

Next we generalize our formula to the case where sensors can be active or inactive with probability p . This generalization was briefly discussed in [39], where it is stated that the main result still holds if $\pi r^2(n)$ is replaced by $\pi r^2(n)p(n)$ in (2.13); however, a more stringent condition is required. Further, it was shown in [74] that a sufficient condition is given by $\pi r^2(n)np \rightarrow \infty$. In fact, it was established that a network graph G is connected if for any $\epsilon \in (0, 1)$, $n_s(\epsilon) = (1 - \epsilon)np$,

1. $\pi r^2(n)np \rightarrow \infty$, as $n \rightarrow \infty$,
2. $\pi r^2(n)n_s(\epsilon) = \log(n_s(\epsilon)) + v(n_s(\epsilon))$, where $v(n) \rightarrow \infty$ as $n \rightarrow \infty$.

This generalization implies that our estimate $n(r)$ can easily incorporate p as follows:

$$(2.20) \quad n(r) = \frac{q(\delta)}{\pi r^2 p}.$$

Using formula (2.20) allows us to simplify the optimization problem for the connectivity constraint alone to

$$(2.21) \quad \min_{r \in R} r^{-2} C(r),$$

with the corresponding optimal value of n given by $n(r) = \frac{q(\delta)}{\pi r^2 p}$.

Again, we illustrate with an additive sensor cost function of the form

$$(2.22) \quad C(r) = c_0(1 + c_2 r^{\beta_2}),$$

where the variable component depends only on the communication radius r . Further, we assume without loss of generality $c_0 \equiv 1$, and $c_2 \geq 0$, $\beta_2 \geq 0$.

The corresponding network-wide cost function and its first derivative (omitting constants) are given by:

$$\begin{aligned} n(r) C(r) &= r^{-2} + c_2 r^{\beta_2-2}, \\ (n(r) C(r))' &= r^{-3} (-2 + c_2(\beta_2 - 2)r^{\beta_2}). \end{aligned}$$

For $0 < \beta_2 \leq 2$, the network cost is a decreasing function of r and the solution occurs at $r = R_U$. For $\beta_2 > 2$, there are values of c_2 for which an interior extremum point \tilde{r} of equation $(n(r) C(r))' = 0$ exists and can be expressed as:

$$(2.23) \quad \tilde{r} = (0.5 (\beta_2 - 2)c_2)^{-1/\beta_2} .$$

If $\beta_2 > 3$, the function is convex, the extremum point is unique, and (2.23) provides the solution to the optimization problem, as long as c_2 is such that $\tilde{r} \in R$, that is, the optimal radius r_o is given by

$$(2.24) \quad r_o = \begin{cases} R_L, & \tilde{r} \leq R_L \\ R_U, & \tilde{r} \geq R_U \\ \tilde{r}, & \tilde{r} \in (R_L, R_U) \end{cases}$$

For $2 < \beta_2 \leq 3$, the function is not convex, and the minimum may be achieved either at the extremum point or at the boundary.

For illustration, the solutions to the optimization problem (optimal radius r and the corresponding number of sensors n) for different values of β_2 and c_2 are given in Table 2.2.

Table 2.2: The optimal communication radius r (rounded to 3 significant digits) and the number of sensors n for different values of c_2 and β_2 with $p = 1$, $\delta = 0.05$, $R = [0.01, 0.35]$.

β_2	2.2		2.4		2.6		2.8		3		5	
c_2	r	n	r	n	r	n	rc_2	n	r	n	r	n
100	.350	39	.290	58	.271	65	.268	66	.272	64	.350	39
500	.169	165	.147	219	.146	222	.151	208	.159	187	.266	67
1e3	.124	310	.110	390	.112	379	.118	340	.126	297	.232	88
5e3	.059	1337	.056	1488	.060	1305	.067	1073	.074	867	.168	167
1e4	.044	2511	.042	2651	.046	2225	.052	1760	.059	1376	.146	221

Example: In this example, we illustrate the proposed approximation using data collected for a number of sensor models from different vendors. The possible communications radii range from 50 to 350 meters, while the unit cost ranges from \$99 to \$299. Then, the estimated unit cost of communication is approximated by

$$(2.25) \quad C(r) = 90 \left(1 + 12.671 r^{1.6163} \right) .$$

The approximated unit cost, the corresponding number and the network total cost as a function of the connectivity radius ($r = [0.05, 0.35]$) with the other parameters fixed to $p = 1$ and $\delta = 0.05$ are shown in Figure 2.7. The global minimum of \$11,362 for the network is achieved with 38 sensors with the largest sensing radius (300m).

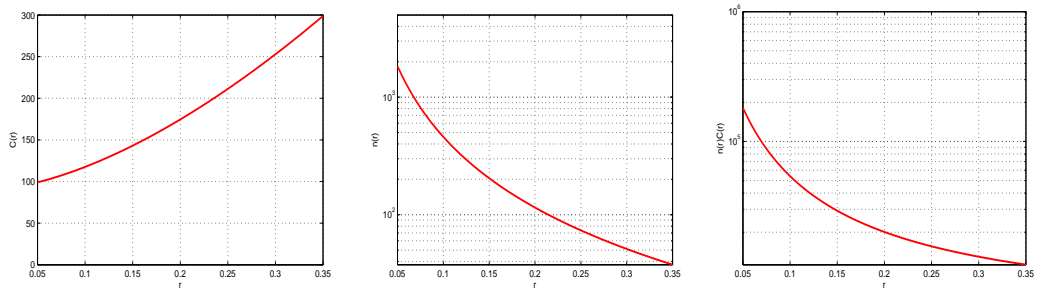


Figure 2.7: Approximated sensor unit cost $C(r)$ (left panel), corresponding number of sensors $n(r)$ (middle panel), and network cost $n(r)C(r)$ (right panel) as a function of connectivity range $r = [0.05, 0.35]$ with $p = 1$ and $\delta = 0.05$.

2.3.3 Joint Coverage and Connectivity Optimization

Recall that the optimization problem (2.2) formulated in Section 5.2 reduces to minimizing $n(s, r)C(s, r)$, where $n(s, r)$ is the smallest n that satisfies the connec-

tivity and coverage constraints for given r and s . Using the explicit expressions for $n(s) \equiv n_1(s)$ from (2.4) and $n(r) \equiv n_2(r)$ from (2.20), we can investigate the behavior of $n(s, r)$. Ultimately it depends on the feasible region D and the values of ϵ , δ , and p . The following three situations can occur:

1. $n_1(s) > n_2(r)$ for all $(s, r) \in D$,
2. $n_1(s) < n_2(r)$ for all $(s, r) \in D$,
3. $D = D_1 \cup D_2$, with $n_1(s) \geq n_2(r)$ for all $(s, r) \in D_1$, and $n_1(s) \leq n_2(r)$ for all $(s, r) \in D_2$.

Case 1: In this case, the coverage constraint dominates and $n(s, r) = n_1(s)$. Since we assume that $C(s, r)$ is a non-decreasing function of r for all s , the optimal value of r is clearly $\tilde{r} = R_L$. The optimal sensing radius \tilde{s} minimizes $n_1(s) C(s, R_L)$, which reduces to the problem of Section 2.3.1.

Case 2: Similarly, the connectivity constraint dominates in this case, and $n(s, r) = n_2(r)$. The cost function $C(s, r)$ is a non-decreasing function of s , and therefore the optimal value of s is $\tilde{s} = S_L$. The optimal communication radius \tilde{r} minimizes $n_2(r) C(S_L, r)$, which reduces to the problem of Section 2.3.2.

Case 3: Again because $C(s, r)$ is a non-decreasing function of s and r , the optimal values s and r will lie either on the boundary or on the curve $n_1(s) = n_2(r)$, which we know in this case passes through the feasible region. This curve is shown in Figure 2.8. It shows that the condition $r = 2s$, where it has been established that coverage implies connectivity [118], is sufficient but not necessary to have both coverage and connectivity constraints satisfied, although it provides a reasonable bound. Solving $n_1(s) = n_2(r)$ for r gives

$$(2.26) \quad r(s) = \left(\frac{q(\delta)}{\pi p \log(\epsilon)} \log(1 - \pi s^2 p) \right)^{1/2}.$$

The optimization problem then reduces to one-variable optimization of the function $n_1(s)C(s, r(s))$ over the set $(s, r(s)) \in D$ (plus boundaries of D). Depending on the functional form of $C(s, r)$, it may be more convenient to solve $n_1(s) = n_2(r)$ for s instead, which gives

$$(2.27) \quad s(r) = \left(\frac{1 - \exp(\pi r^2 p \log(\epsilon)/q(\delta))}{\pi p} \right)^{1/2},$$

and solve the optimization problem as a function of r over the set $(s(r), r) \in D$.

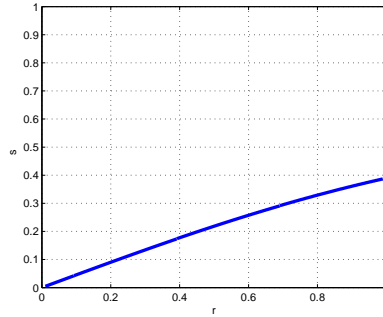


Figure 2.8: The curve defined by $n_1(s) = n_2(r)$ with $\epsilon = \delta = 0.05$, $p = 1$.

Extending the additive cost function idea to include costs associated with both sensing and transmission capabilities, we consider a sensor cost function of the form

$$(2.28) \quad C(s, r) = c_0(1 + c_1 s^{\beta_1} + c_2 r^{\beta_2}),$$

with $c_0 = 1$ without loss of generality and non-negative parameters $c_1, c_2, \beta_1, \beta_2$.

In general, all three cases can occur. Specifically,

Case 1: For the rectangular region D , this case occurs when $n_1(S_U) > n_2(R_L)$. The optimal value of r is $\tilde{r} = R_L$. The extremum points of the function $n_1(s)C(s, R_L)$ are solutions to the equation:

$$(2.29) \quad b(s)(1 + c_1 s^{\beta_1} + c_2 R_L^{\beta_2}) - c_1 \beta_1 s^{\beta_1} = 0,$$

where $b(s)$ is defined as where $b(s) = \frac{-2s^2\pi p}{(1-\pi s^2 p)\log(1-\pi s^2 p)}$. This equation can be solved numerically. The optimal solution \tilde{s} can be either an extremum point or a boundary point, depending on the values of β_1 , β_2 , c_1 , and c_2 (see examples in Table 2.3).

Case 2: This case occurs when $n_2(R_U) > n_1(S_L)$, and the optimal value for s is $\tilde{s} = S_L$. The extremum point of $n_2(r)C(S_L, r)$ is defined by

$$(2.30) \quad r^{\beta_2} = \frac{2(1 + c_1 S_L^{\beta_1})}{c_2(\beta_2 - 2)}$$

Clearly, extremum points only exist if $\beta_2 > 2$. The optimal value \tilde{r} may be an extremum point or one of the boundaries R_U, R_L .

Case 3: Plugging in the expression (2.27) for $s(r)$ into $n_2(r)C(r, s(r))$ and taking the derivative with respect to r gives an extremum point equation

$$(2.31) \quad -2 + c_2(\beta_2 - 2)r^{\beta_2} + s^{\beta_1}(r)c_1 \left(-2 + r\beta_1 \frac{s'(r)}{s(r)} \right) = 0 ,$$

where

$$s'(r) = -\frac{r \log(\epsilon)}{s(r) q(\delta)} \exp(\pi r^2 p \log(\epsilon)/q(\delta)) .$$

This can be solved numerically, and along with boundary points constitutes the candidate solution points.

As an illustration, in Table 2.3 the solutions to the joint optimization problem are presented for different values of β_1 and β_2 with $c_1 = c_2 = 10^3$. It is assumed that the feasible interval for both r and s is $[0.01, 0.35]$, all sensors are in active mode ($p = 1$), the maximum proportion of area not covered is $\epsilon = 0.05$, and the highest probability of a disconnected network is $\delta = 0.05$.

Example: We revisit the examples discussed in Sections 2.3.1 and Section 2.3.2. Recall that the network cost as a function of the sensing s and the communications r range was estimated as

$$(2.32) \quad C(s) = 3 \left(1 + 3234.2 s^{2.876} \right) ,$$

Table 2.3: Optimum values of the sensing radius s and communication radius r , and the corresponding number of sensors n , for different values of β_1 and β_2 with $c_1 = c_2 = 10^3$, $\epsilon = 0.05$, $\delta = 0.05$, and $p = 1$.

β_1	2.0			2.4			3.0			5.0		
β_2	s	r	n	s	r	n	s	r	n	s	r	n
2.0	.155	.347	39	.118	.262	67	.129	.287	56	.157	.350	38
2.4	.050	.110	380	.046	.101	451	.050	.110	380	.050	.110	380
2.8	.055	.121	315	.050	.110	380	.052	.114	354	.054	.118	331
3.0	.059	.130	273	.055	.121	315	.055	.121	315	.058	.126	290
5.0	.110	.244	78	.090	.199	117	.095	.210	105	.104	.230	87

and

$$(2.33) \quad C(r) = 90 \left(1 + 12.671 r^{1.6163} \right),$$

respectively. We consider next a joint additive model next

$$(2.34) \quad \begin{aligned} C(s, r) = C(s) + C(r) &= 93 \left(1 + \frac{3 \cdot 3234.2}{90 + 3} s^{2.876} + \frac{90 \cdot 12.671}{90 + 3} r^{1.6163} \right) = \\ &= c_0(1 + c_1 s^{\beta_1} + c_2 r^{\beta_2}), \end{aligned}$$

shown in Figure 2.9. The global minimum of the network cost function is achieved at $s = 0.156$ and $r = 0.348$ with $n = 38$ and the optimal cost is $38 \times 346.4 = 13163.2$ dollars.

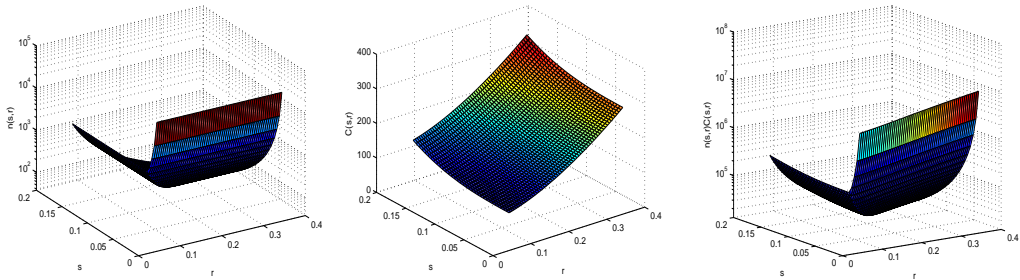


Figure 2.9: Number of sensors $n(s, r)$ (left panel), sensor cost function $C(s, r)$ (middle panel), and overall network cost $n(s, r)C(s, r)$ (right panel) with $c_0 = 93$, $c_1 = 104.329$, $c_2 = 12.2623$, $\beta_1 = 2.8765$, $\beta_2 = 1.6163$, $\epsilon = \delta = 0.05$, and $p = 1$. The optimal design is $s = 0.156$, $r = 0.348$, $n = 38$ and the optimal cost is $38 \times 346.4 = 13163.2$ dollars.

2.4 Extensions

The proposed framework can easily accommodate additional constraints; for example, separate costs and coverage requirements for cluster heads of hierarchical networks. The most straightforward extension is for coverage by a WSN comprised of heterogeneous types of sensors. Suppose there are K types of sensors (e.g. acoustic, thermal, etc.), with sensing radii s_k , $k = 1, \dots, K$, and n_k sensors of each type k are deployed at random over the region X . Further, suppose that all type k sensors have probability p_k of being in active mode, and a set of feasible sensing radii S_k . Then, it is fairly straightforward to extend expression (2.3) to show that the expected area not covered is

$$(2.35) \quad \mathbb{E}(|\tilde{X}|) = \prod_{k=1}^K \left(1 - \frac{p_k \pi s_k^2}{|X|}\right)^{n_k} |X|.$$

If the coverage problem is considered alone, and each sensor type has a cost function C_k , the problem becomes

$$\min_{n_k, s_k \in S_k} \sum_{k=1}^K n_k C_k(s_k) \quad \text{subject to} \quad \sum_{k=1}^K n_k \log(1 - p_k \pi s_k^2) < \log(\epsilon).$$

Example: We extend the example discussed in Section 2.3.2. It is assumed that the number of sensors with an extended sensing range of $s = 300$ is limited to 5. Provided that these 5 sensors have been uniformly deployed over a monitoring region of size 1km^2 , it is then of interest to estimate the number of sensors needed to cover the area with minimum cost, subject to a coverage constraint of $\epsilon = 0.05$. From the function fitted to the collected data, we obtain a solution corresponding to 23 sensors with $s = 134$ m, resulting in an overall cost of \$531. On the other hand, using the quadratic approximation we require 62 sensors of sensing radius 84.2 m for a cost of \$643.56. Adding to the two solutions the fixed cost of \$337.75 of the

5 high-capability sensors, it can be seen that savings of \$331.86 (\$461.51) can be achieved compared to a solution utilizing only low capability sensors.

The connectivity problem for a network with varying communications radii is an open problem in the general case. If one assumes that all types of sensors have the same communications capabilities, or that each type of sensors only communicates with other sensors of the same type, the connectivity constraint is easy to incorporate using the developments from Section 2.3.2.

For ease of presentation, here we assumed that the probability of a sensor actively sensing (p_1) is the same as the probability of being available for communications (p_2), with $p_1 = p_2 = p$. However, it is trivial to extend the calculations to the case $p_1 \neq p_2$. Another interesting extension is to allow the reliability parameter(s) to be linked to the sensor characteristics s and r and its cost. The general framework remains the same, but the optimization problem becomes significantly more involved.

CHAPTER III

Target Detection with Wireless Sensor Networks

3.1 Introduction

We explore in this chapter different data fusion algorithms for target detection by a WSN. In order to measure the accuracy of the detection the false alarm probability and the detection probability are used. The *false alarm probability* is the conditional probability that the sensors detect the target given that there is no target in the monitored region. The *detection probability* is the conditional probability that the sensors correctly report the presence of the target. So, when the signal-to-noise ratio is low for a given system false alarm, the probability of detection is also low.

Suppose that N sensors have been deployed at locations s_i , $i = 1, \dots, N$, over a two-dimensional monitoring region R , which without loss of generality corresponds to the unit square. A target at location $v \in R$ emits a signal captured by the sensors. Let $E_i = S_i + \epsilon_i$ denote the energy measured by the i -th sensor, where $S_i \equiv S_i(v)$ is the signal from the target measured at location i , and ϵ_i , $i = 1 \dots N$ are i.i.d. random noise. It is natural to assume that the signal strength decays monotonically as the distance from the sensor to the target increases. For example, the left panel of Figure 3.1 shows the signal strength of a target located in the center of R exhibiting exponential decay, while the right panel shows the same signal corrupted by Gaussian

noise.

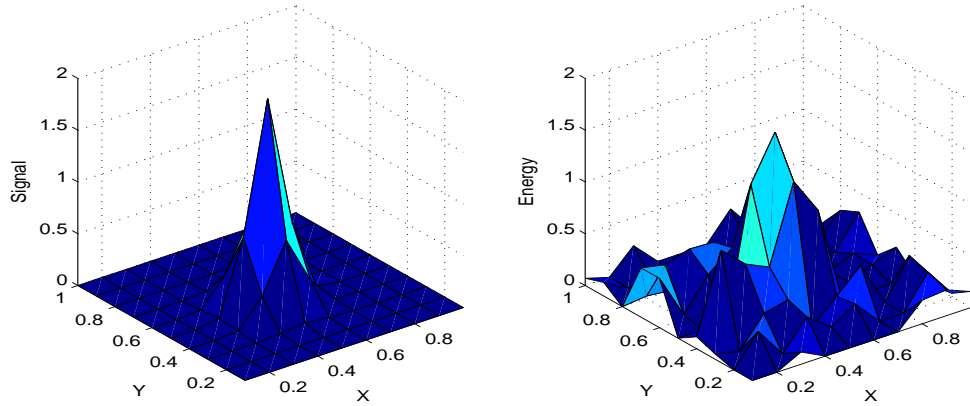


Figure 3.1: Left panel: Target signal generated by the model $S_i(v) = S_0 \exp(-\|s_i - v\|^2/\eta^2)$ for a target at location $v = (0.5, 0.5)$ with $S_0 = 2$, $\eta = 0.1$; Right panel: Target energy contaminated by Gaussian noise of variance $\sigma^2 = 0.16$ (signal-to-noise ratio $S_0/\sigma = 5$).

Based on the observed energy levels E_i , each sensor reaches a decision $Y_i \in \{0, 1\}$ regarding the presence of the target in the monitoring area. The decision depends on whether the energy level exceeds a pre-specified threshold τ_i , which determines the individual sensor's false alarm probability; i.e., $Y_i = I(E_i \geq \tau_i)$, where $I(\cdot)$ is the indicator function.

The two main options for reaching a joint decision are *value fusion*, where sensors transmit the energy readings back to the fusion center, and *decision fusion*, where each sensor decides first whether a target is present and only the binary decisions are transmitted. In [20] value and decision fusion were compared and it was found that the former performs better in terms of detection probability for low noise levels; however, for noisy energy measurements decision fusion proves more robust. Further, it offers significant savings in communications costs, since only *positive* one-bit decisions need to be transmitted. For these reasons, in our research we focus on decision fusion.

The classical approach to this problem (see [113] for a comprehensive review) is to assume a specific model for the signal and frame it as a hypothesis test of the null hypothesis H_0 : no target is present vs. the alternative H_1 : there is a target in the field. This approach was used by fusion algorithms developed in the 1980's with application to surveillance systems (e.g. radars). Optimal decision rules based on classical Bayesian decision theory have been worked out when signal and noise distributions are known, for independent [108, 14] and correlated [50] decisions, though in the latter case the computations are cumbersome. In [108], it is assumed that the false alarm and detection probabilities are the same for all sensors, a fairly reasonable assumption for a remote target in, say, a missile defense system, but not for a wireless sensor network with a relatively small target in the middle of a large region. In [14], optimal weights (in the sense of classical decision theory) were derived for combining decisions Y_i if the detection and false alarm probabilities are known for all sensors. The corresponding Chair-Varshney fusion rule is optimal in such a setting. Another way to derive an optimal rule is to apply the local asymptotic normality framework [104]. However, for a wireless sensor network, the assumption of known detection probabilities is unrealistic [15]: even if a specific model for the signal is assumed and all model parameters are known, the detection probability for each sensor depends on its distance from the target, and assuming a known location for the target would defeat the whole purpose of target detection. Further, the assumption of a known signal model can also be restrictive, since there may be different types of targets present, whose signals follow different models.

Here we propose a different approach to decision fusion: we make no assumptions about the signal model whatsoever (other than that the signal is positive). The null hypothesis of no target present is H_0 : $S_i = 0$ for all i , while the alternative is simply

H_1 : $S_i > 0$ for some i . This approach is unlikely to be optimal if a specific signal model is known in advance, since a better performance can be achieved under additional assumptions; however, it has the advantage that it is applicable even when no prior knowledge about the target's signal characteristics is available, as empirically shown in Section III. Note that this formulation follows the classical (frequentist) approach of treating the S_i 's as unknown non-random parameters. Then, the energy readings E_i and the corresponding decisions Y_i are independent, since the only randomness comes from the i.i.d. noise ϵ_i . Further, we assume that all sensors are identical, and that they all use the same threshold $\tau_i = \tau$. Then under H_0 , all E_i 's are i.i.d. and all sensors have the same false alarm probability $\gamma = P_{H_0}(Y_i = 1)$. In contrast, a large body of literature adopts the so-called *conditional independence* assumption: $P(E_1, \dots, E_N | H_m) = \prod_{i=1}^N P(E_i | H_m)$ for $m = 0, 1$. This assumption is inherently Bayesian, since conditioning on the hypothesis implies that the signal means S_i 's are treated as random variables.

A consequence of not assuming any particular model for the signal is that the optimal Chair-Varshney type weighted rules can not be computed. Nevertheless, the following simple model-independent decision fusion algorithm studied in the literature (e.g. [20]) can be used as a benchmark comparison. We will refer to it as *ordinary decision fusion* (ODF):

Ordinary Decision Fusion:

1. Each sensor i makes its own decision $Y_i \in \{0, 1\}$ w.r.t. the sensor threshold τ ,

$$Y_i = I(E_i \geq \tau);$$
2. Sensors transmit positive decisions to the fusion center;
3. The fusion center obtains final decision based on a pre-specified threshold T ,

$$I(\sum_i Y_i \geq T).$$

Given a target in R , the objective of the sensor network is to maximize its probability of detection D , while controlling the corresponding system-wide false alarm probability F . The more recent work on decision fusion for target detection by wireless sensor networks has primarily focused on threshold selection, both for individual sensors and the global decision. We note that, for any given sensor false alarm probability γ , the global threshold T can always be selected to achieve the desired overall system false alarm F , so in our problem formulation it is not possible to optimize both thresholds simultaneously. Under the assumption of independent sensor decisions, expressions for false alarm and detection have been calculated exactly using binomial probabilities [83], approximated using the central limit theorem [84] and the saddle-point approximation [7], and bounded using Chebyshev's inequality [127]. The overall detection calculations typically assume known sensor detection rates, which is, again, unrealistic for a target in an unknown location. The expressions for false alarm can be used to set the threshold, but none of these techniques would work for our proposed algorithm, since they rely on independence of sensor decisions.

On the algorithmic side, alternatives to ordinary decision fusion have been proposed. Distance Weighted Voting [29] weighs individual sensor decisions by the inverse of their distance to the target, which applies only to detection at a pre-specified location. Confidence Weighted Voting [103], perhaps closest in spirit to what we propose, weighs sensor decisions by a measure of confidence based on the neighborhood agreement. Finally, in [57] the decision for a pre-specified location is made by majority vote in its neighborhood, but this was only derived for a 3-sensor system. No analytical performance guarantees exist for these methods, and it is not clear how to choose thresholds to achieve a desired false alarm rate.

In this Chapter we make two main contributions: first, we propose a new decision fusion algorithm based on first locally adjusting individual sensors' decisions and subsequently integrating them at the network level. Second, we provide a rigorously derived analytical approximation for the system-wide decision threshold level T as a function of the system-wide false alarm probability F , obtained using limit theorems for random fields. This ensures one can design a network with a guaranteed false alarm rate using our algorithm. We show that the approximation performs very satisfactorily for both random and fixed grid deployments, even for networks comprised of a small number of sensors. Further, the proposed decision fusion algorithm substantially outperforms ordinary decision fusion in terms of target detection probability, and achieves good results at a significantly lower signal-to-noise ratio. Finally, we show how to extend this algorithm to temporal decision fusion, which allows detection of moving targets over time.

3.2 Methods and Algorithm

3.2.1 Decision Fusion for Target Detection

In order to guarantee the overall system performance of decision fusion for target detection, one must be able to obtain the threshold T for the whole network, given an individual sensor's and the system's false alarm probabilities γ and F , respectively. It is assumed that γ is determined either by hardware specifications or from information about background noise levels, whereas F is selected by the network's operator.

We start our exposition from the ordinary decision fusion algorithm, where the sensors' decisions are simply added at the fusion center, and give an expression for the false alarm rate and the corresponding target detection probability (similar analysis can be found e.g. in [83, 105]). Let G denote the distribution function of the noise levels, i.e. $\epsilon_i \sim G$. In the absence of a target, the probability of a positive decision

(false alarm probability) is given by the right tail of the binomial distribution,

$$(3.1) \quad F = \sum_{i=T}^N \binom{N}{i} \gamma^i (1-\gamma)^{N-i},$$

since sensors make individual decisions independently, with $\gamma = G(\tau)$.

In the presence of a large number of sensors, the above tail probability can be fairly accurately determined by the normal approximation given by

$$(3.2) \quad F \approx 1 - \Phi\left(\frac{T - N\gamma}{\sqrt{N\gamma(1-\gamma)}}\right),$$

where $\Phi(\cdot)$ denotes the standard normal cumulative distribution function. Therefore, for specific sensor and system false alarm probabilities γ and F , one can compute the corresponding threshold T . Note that knowledge of the background noise distribution G is not required, as long as γ is known. Then, a target is detected by the network if at least T sensors measure energy levels that exceed their individual threshold τ . Hence, the probability of detection is given by

$$(3.3) \quad D = \sum_{i=T}^N \sum_{\pi \in \Gamma} \prod_{j=1}^i (1 - G(\tau - E_{\pi(j)})) \prod_{j=i+1}^N G(\tau - E_{\pi(j)}),$$

where Γ denotes the set of all permutations of $\{1, \dots, N\}$. The first product term corresponds to the probability that sensors $\pi(1), \dots, \pi(i)$ make positive decisions, while the second product term corresponds to the probability that sensors $\pi(i+1), \dots, \pi(N)$ make negative decisions. The detection probability depends on the target's and the sensors' locations, signal parameters, and the noise distribution. Even though in principle (3.3) gives a closed-form analytical expression for detection probability, computing it this way is not feasible numerically; instead, we compute it through simulation (see Section 3.3).

3.2.2 Local Vote Decision Fusion (LVDF): the algorithm

We propose next a modification of the ODF mechanism that first adjusts each sensor's decision locally by taking a majority vote in its neighborhood. For each sensor i , the neighborhood $U(i)$ can be defined as either all sensors within a fixed distance r (e.g., communications range), or as a fixed number of its nearest neighbors. By definition, $i \in U(i)$, so the sensor's own decision is always taken into account.

Local Vote Decision Fusion (LVDF):

1. Sensor i makes an *initial* decision Y_i independent of all other sensors and communicates it to all other sensors j in its neighborhood $U(i)$,
2. Subsequently, given a set of decisions $\{Y_j : j \in U(i)\}$, sensor i adjusts its initial decision according to a majority vote; i.e., $Z_i = I(\sum_{j \in U(i)} Y_j > M_i/2)$, where $M_i = |U(i)|$ denotes the size of the neighborhood.
3. The positive updated decisions Z_i are communicated to the fusion center, which makes the final decision $I(\sum_{i=1}^N Z_i \geq T_\ell)$.

In practice, sensors only need to communicate positive decisions in step 1; an absence of communication according to some pre-specified protocol implies that $Y_i = 0$. In Figure 3.2, the advantage of local vote decision fusion over ordinary decision fusion is illustrated for both random and fixed grid deployments. Under ordinary decision fusion, the threshold T is higher since more wrong decisions from sensors located far away from the target are expected; these false positives have a significant adverse effect on the sensor network's final decision. On the other hand, the proposed local vote mechanism fixes those isolated decisions and helps the network reach the correct conclusion. If communications to neighbors are cheaper than those to the fusion center, which is typical, LVDF also reduces the overall communication costs since by

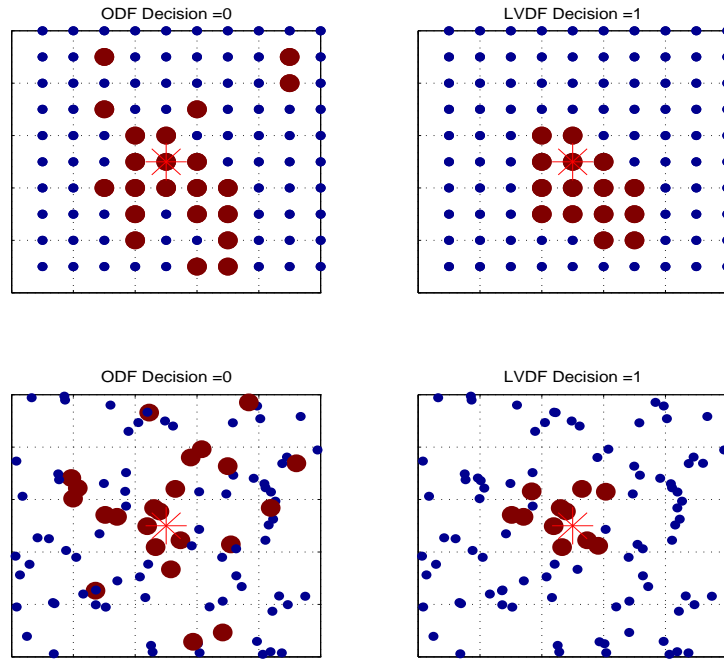


Figure 3.2: Ordinary vs Local Vote decision fusion under a square grid design (top panels) and random deployment (bottom panels). The network is comprised of 100 sensors, with individual sensor false alarm probability $\gamma = 0.2$, system-wide false alarm probability $F = 0.1$ and a target located at the center of the monitored region R . The signal is generated by the model $S_i = S_0 \exp(-\|s_i - v\|^2/\eta^2)$, with $S_0 = 2$, $\eta = 0.1$, and the measured energy is corrupted by Gaussian noise with $\sigma = 0.4$.

canceling out false positives it reduces the number of positive decisions which need to be communicated to the fusion center.

3.2.3 LVDF: threshold selection

We derive next the system-wide threshold value T_ℓ for LVDF that guarantees a false alarm probability F . The strategy is to derive a normal approximation for F for large sensor networks. However, unlike the initial decisions, the updated ones exhibit dependences, a fact that introduces certain technical challenges that are resolved next.

We start by calculating the expected value and variance of the updated decision Z_i under H_0 :

$$(3.4) \quad \mu_i = P(Z_i = 1) = \sum_{j=[M_i/2]+1}^{M_i} \binom{M_i}{j} \gamma^j (1 - \gamma)^{M_i-j},$$

where $[x]$ denotes the largest integer smaller than or equal to x . The variance is given by $\sigma_i^2 = \text{Var}(Z_i) = \mu_i(1 - \mu_i)$.

The dependence between Z_i and Z_j , $j \neq i$ comes from the intersection of their respective neighborhoods $U(i)$ and $U(j)$, as shown on the Figure 3.3. Let n_{ij} denote the number of sensors in the intersection $U(i) \cap U(j)$.

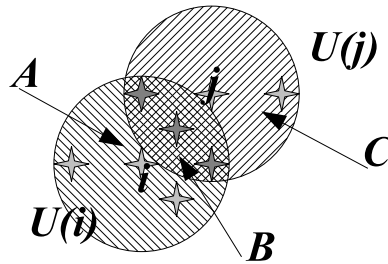


Figure 3.3: Example of sensor neighborhoods with $M_i = 6$, $M_j = 5$ and $n_{ij} = 3$.

In order to calculate the covariance between Z_i and Z_j we first compute $E(Z_i Z_j) = P(Z_i = Z_j = 1)$. Let A be the number of positive decisions in $U(i) \cap U(j)$, B the

number of positive decisions in $U(i)$ but not in $U(j)$, and C the number of positive decisions in $U(j)$ but not in $U(i)$, and note that A , B , and C are independent. Then we can write (letting $\binom{a}{b} \equiv 0$ if $b < 0$)

$$\begin{aligned}
E(Z_i Z_j) &= \sum_{k=0}^{n_{ij}} P(A = k) P(B > \frac{M_i}{2} - k) P(C > \frac{M_j}{2} - k), \quad \text{where} \\
P(A = k) &= \binom{n_{ij}}{k} \gamma^k (1 - \gamma)^{n_{ij} - k}, \\
P(B > \frac{M_i}{2} - k) &= \sum_{q=\lfloor \frac{M_i}{2} \rfloor - k + 1}^{M_i - n_{ij}} \binom{M_i - n_{ij}}{q} \gamma^q (1 - \gamma)^{M_i - n_{ij} - q}, \\
(3.5) \quad P(C > \frac{M_j}{2} - k) &= \sum_{q=\lfloor \frac{M_j}{2} \rfloor - k + 1}^{M_j - n_{ij}} \binom{M_j - n_{ij}}{q} \gamma^q (1 - \gamma)^{M_j - n_{ij} - q}.
\end{aligned}$$

The term A is the probability that enough positive decisions for both sensors i and j to make decisions $Z_i = Z_j = 1$ are present in the intersection of their neighborhoods $U(i) \cap U(j)$. The term B_k is the probability that there are exactly k positive decisions in $U(i) \cap U(j)$ (but not enough to make both Z_i and Z_j positive automatically); and the terms C_k and D_k are the probabilities that there are enough positive decisions outside of the intersection to make $Z_i = 1$ and $Z_j = 1$, respectively.

The covariance is then given by

$$(3.6) \quad \text{Cov}(Z_i, Z_j) = [E(Z_i Z_j) - \mu_i \mu_j] I(n_{ij} \neq 0).$$

Under the assumption that the target is absent, the system's false alarm probability is given by

$$(3.7) \quad F = P\left(\sum_{i=1}^N Z_i \geq T_\ell\right),$$

where T_ℓ denotes the local-vote decision fusion threshold. The updated decisions $\{Z_i; i = 1, \dots, N\}$ form a dependent random field. The central limit theorem applies

to the $\sum Z_i$ (see Section 3.2.4), both for sensors deployed on a regular grid or at random. The following approximation then holds:

$$(3.8) \quad F \approx 1 - \Phi \left(\frac{T_\ell - \sum_{i=1}^N \mu_i}{\sqrt{\sum_{i=1}^N \sigma_i^2 + \sum_{i \neq j, n_{ij} \neq 0} \text{Cov}(Z_i, Z_j)}} \right).$$

Remark 1: Fixed neighborhood size

In some settings (e.g., dense deployments or regular grids) the number of neighbors may be fixed to a pre-specified number with $|U(i)| = M$ for all i . In this case we have $Z_i = I \left\{ \sum_{j \in U(i)} Y_j > \frac{M}{2} \right\}$, which shows that the Z_i 's are dependent but now *identically* distributed. Hence, the mean $E(Z_i) = \mu$ and the variance $\text{Var}(Z_i) = \sigma^2$ can be calculated using (3.4). Then, $E(Z_i Z_j)$ can be calculated from (3.5) with $M_i = M_j = M$ and the resulting covariance is given by $\text{Cov}(Z_i, Z_j) = [E(Z_i Z_j) - \mu^2]I(n_{ij} \neq 0)$. The normal approximation simplifies to

$$(3.9) \quad F \approx 1 - \Phi \left(\frac{T_\ell - N\mu}{\sqrt{N\sigma^2 + \sum_{i \neq j, n_{ij} \neq 0} \text{Cov}(Z_i, Z_j)}} \right).$$

Remark 2: Regular Grids

In some applications, it may be possible to deploy the sensors along a regular grid. In this case the false alarm approximation (3.8) further simplifies under the assumption that each sensor has exactly M neighbors to consult including itself (ignoring edge effects). In practice, this can be achieved by ignoring corrected decisions of sensors on the edges, effectively reducing the grid size. On a regular grid, the one-hop neighborhood contains either $M = 5$ (diamond-shaped neighborhood) or 9 neighbors (square neighborhood), depending on whether diagonally located nearest neighbors are included or not, and $M = 7$ (hexagonal neighborhood), the three most common designs considered in classical random fields theory[28] (see Figure 3.4).

(i) **Square neighborhood:** The number of hops (layers) away from the sensor at the center determines the size of the neighborhood. Let m denote the number

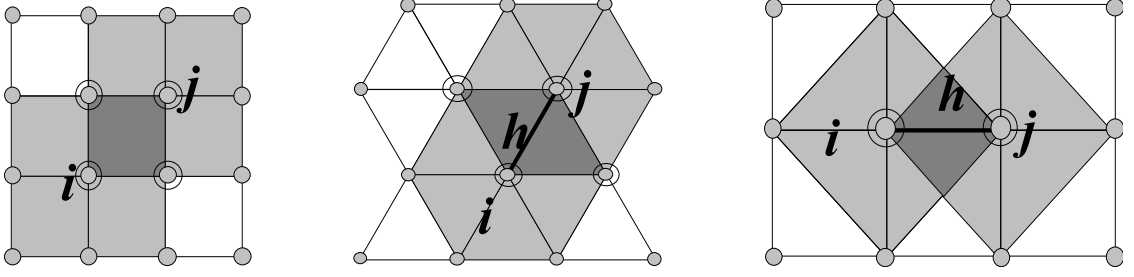


Figure 3.4: Square (left panel), hexagonal (center panel) and diamond-shaped (right panel) neighborhoods on a regular grid.

of layers considered. Then, the size of the square neighborhood $U(i)$ is given by $M = (2m + 1)^2$. Let $t = (t_1, t_2)$ be a location shift and $U(i + t)$ the neighborhood of the sensor located at $s_i + t$. Then the number of common sensors in $U(i)$ and $U(i + t)$ is given by $n_{i,i+t} = (2m + 1 - |t_1|)(2m + 1 - |t_2|)$, with $0 \leq |t_1|, |t_2| \leq 2m$, and $n_{i,i+t} = 0$ otherwise. The covariance is given by

$$(3.10) \quad \text{Cov}(Z_i, Z_{i+t}) = \text{Cov}(t) = [E(Z_i Z_{i+t}) - \mu^2] I(0 \leq |t_1|, |t_2| \leq 2m)$$

and the normal approximation of F can be obtained as before.

The previous formula does not reflect the presence of edge effects, that are taken into account in the following formula:

$$(3.11) \quad n_{i,i+t} = \left| U(i) \cap U(i+t) \right| = (\max(m_i^L, m_{i+t}^L) - |t_1|)(\max(m_i^W, m_{i+t}^W) - |t_2|),$$

where m_i^L and m_i^W are the length and the width of the neighborhood of sensor i , i.e. $M_i = m_i^L \cdot m_i^W$. For each sensor i the length m_i^L and the width m_i^W can be calculated as follows:

$$(3.12) \quad \begin{aligned} m_i^L &= \min(t_{i,1}^{max} - t_{i,1}^{min}, 2m + 1), \\ m_i^W &= \min(t_{i,2}^{max} - t_{i,2}^{min}, 2m + 1), \end{aligned}$$

where

$$\begin{aligned} t_{i,1}^{min} &= \max(-2m, -(i_1 - 2m + 1)), \quad t_{i,1}^{max} = \min(2m, n - i_1), \\ t_{i,2}^{min} &= \max(-2m, -(i_2 - 2m + 1)), \quad t_{i,2}^{max} = \min(2m, n - i_2), \end{aligned}$$

and sensor location index i is a pair of indexes $i = (i_1, i_2)$ along horizontal and vertical dimensions. The formula for the covariance can be written as:

$$(3.13) \quad \text{Cov}(Z_i Z_{i+t}) = \begin{cases} E(Z_i Z_{i+t}) - \mu_i \mu_{i+t}^2, & 0 < t_{i,j}^{min} \leq |t_j| \leq t_{i,j}^{max} < 2m, \quad j \in 1, 2 \\ \sigma_i^2, & |t_1| = |t_2| = 0 \\ 0, & |t_j| > t_{i,j}^{max}, \quad j \in 1, 2 \end{cases}$$

and the normal approximation of F can be obtained as before. However, for large networks, the edge effect is negligible. Simulation results show that there is no significant difference in a quality approximation for network size $n \leq 10$.

(ii) Diamond-shaped neighborhood: We only consider the single-layer neighborhood with $M = 5$. The possible values for the size of non-empty intersections of $U(i)$ and $U(j)$ are

$$(3.14) \quad n_{ij} = |U(i) \cap U(j)| = \begin{cases} 1, & \|s_i - s_j\| = 2h \\ 2, & h \leq \|s_i - s_j\| \leq h\sqrt{2} \end{cases}$$

where $2h$ is the size of the diamond's diagonal. The approximation for F can then be straightforwardly obtained.

(iii) Hexagonal neighborhood. For a hexagonal grid design, let h denote the side of the hexagon. Here we only consider the single-layer neighborhood with $M = 7$. The only possible values for the size of non-empty intersections of $U(i)$ and $U(j)$ are

$$(3.15) \quad n_{ij} = |U(i) \cap U(j)| = \begin{cases} 1, & \|s_i - s_j\| = 2h \\ 2, & \|s_i - s_j\| = h\sqrt{3} \\ 4, & \|s_i - s_j\| = h \end{cases}$$

and the corresponding approximation formula is

$$(3.16) \quad F \approx 1 - \Phi \left(\frac{T_\ell - N\mu}{\sqrt{N(\sigma^2 + 6 \sum_{n_{ij} \in \{1,2,4\}} \text{Cov}(n_{ij}))}} \right),$$

where the factor of 6 comes again from the symmetry of the grid.

3.2.4 LVDF: Central Limit Theory for threshold approximation

Here we establish the central limit theorem for correlated LVDF decisions Z_i which was used as the basis for the approximation (3.8). The key property of the variables Z_i we use here is that even though they are correlated, correlations are only present between 'nearby' locations.

Regular grids. The concept of correlations between 'nearby' locations is formalized in the notion of mixing. Let $\{X_i, i \in \mathbb{Z}^d\}$, $d \geq 1$, be a real-valued random field, i.e., a set of random variables defined on a d -dimensional lattice (in the present context $d = 2$). For any set of indices $E \subset \mathbb{Z}^d$, let $\mathcal{F}(X_E) = \mathcal{F}\{X_i; i \in E\}$ be the σ -field generated by the collection $\{X_i : i \in E\}$. Informally, $\mathcal{F}(X_E)$ contains all events related to the variables $\{X_i : i \in E\}$. Define

$$\alpha_{u,v}(k) = \sup\{|P(A \cap B) - P(A)P(B)| : \begin{array}{l} A \in \mathcal{F}(X_E), B \in \mathcal{F}(X_F), \\ |E| \leq u, |F| \leq v, \text{dist}(E, F) \geq k \end{array}\},$$

where the distance between two sets of indices E, F is defined as

$$\text{dist}(E, F) = \max\{\|x - y\|, x \in E, y \in F\},$$

$$\|x - y\| = \max_{j=1, \dots, d} |x_j - y_j|.$$

Then the field $\{X_i\}$ is called α -mixing if for all u and v ,

$$\alpha_{u,v}(k) \rightarrow 0, \text{ as } k \rightarrow 0.$$

The theory of mixing random fields has been studied extensively [28, 40]. We rely on the following theorem to establish the result.

Theorem 1: ([40], p.111). Let $X = \{X_i, i \in \mathbb{Z}^d\}$ be a real-valued field with $EX_i = 0$ (not necessarily stationary) and $\{D_n\}$ a sequence of strictly increasing finite domains of \mathbb{Z}^d . Let $S_n = \sum_{i \in D_n} X_i$ and let $\sigma_n^2 = \text{Var}(S_n)$. Assume that

- (i) $\sum_{k=1}^{\infty} k^{d-1} \alpha_{u,v}(k) < \infty$, if $u + v \leq 4$ and $\alpha_{1,\infty}(k) = o(k^{-d})$.
- (ii) There exists $\delta > 0$ such that $\sup_i \|X_i\|_{2+\delta} < \infty$, and $\sum_{k=1}^{\infty} k^{d-1} \alpha_{1,1}(k)^{\frac{\delta}{2+\delta}} < \infty$.
- (iii) $\liminf_n |D_n|^{-1} \sigma_n^2 > 0$.

Then,

$$(3.17) \quad \sigma_n^{-1} S_n \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1).$$

Based on this result, we establish

Theorem 2: Let $\{Z_i, i = 1, \dots, N\}$ be the LVDF corrected decisions on a regular grid with N nodes over the unit square, $D_N = (\frac{1}{\sqrt{N}}[1 : \sqrt{N}])^2$. Let $\mu_i = E(Z_i)$ and $S_N = \sum_{i=1}^N Z_i$. Then, as $N \rightarrow \infty$,

$$(3.18) \quad \frac{S_N - \sum_{i=1}^N \mu_i}{\sqrt{\text{Var}(S_N)}} \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1).$$

We examine the approximation for two variants of LVDF. The first one is *distance based local vote decision fusion* (D-LVDF), which defines the neighborhood $U(i)$ as all sensors within a circle of fixed radius r from sensor i , i.e., $U(i) = \{j : \|s_i - s_j\| \leq r\}$. The second one called *nearest neighbor local vote decision fusion* (NN-LVDF) fixes the number of neighbors to be considered; i.e., $U(i) = \{i\} \cup \{M-1 \text{ nearest neighbors of } i\}$. The theorem holds for both D-LVDF and NN-LVDF.

Proof: The proof is carried out in two steps. The result is established first for the variables Z_i defined on a rescaled version of D_N , $\tilde{D}_N = N D_N = [1 : \sqrt{N}]^2$. Subsequently, it is extended to the original domain D_N .

The definition of the neighborhoods $U(i)$ (either circles of fixed radius as in D-LVDF, or M nearest neighbors as in NN-LVDF) implies that the mixing coefficients $\alpha_{u,v}(k) \equiv 0$ for all $k \geq k_0$, with k_0 being a fixed constant independent of N . Hence, all the conditions on the mixing coefficients in (i) and (ii) are automatically satisfied.

Further, Z_i are binary and thus $\|Z_i\|_{2+\delta} \leq 1$ for all i and δ , which implies that condition (ii) holds as well.

Next, we establish condition (iii) holds in the present setting. Note that for all i and j , $P(Z_i = 1|Z_j = 1) \geq P(Z_i = 1)$, with strict inequality if $n_{ij} > 0$ and equality if $n_{ij} = 0$. Thus,

$$(3.19) \quad \text{Cov}(Z_i, Z_j) = (P(Z_i = 1|Z_j = 1) - P(Z_i = 1))P(Z_j = 1) \geq 0.$$

We also have

$$(3.20) \quad \sigma_N^2 \equiv \text{Var}(S_N) \geq \sum_{i=1}^N \text{Var}(Z_i) \geq N \inf_i \sigma_i^2 = N\sigma_0^2.$$

For the fixed number of neighbors case, we have $\mu_i = \mu = \sum_{j=[M/2]+1}^M \binom{M}{j} \gamma^j (1-\gamma)^{M-j}$ and $0 < \mu < 1$. For the fixed radius case on a regular grid, the only possible values for M_i are in the set $\{1, \dots, M\}$, where M is the full neighborhood size (unaffected by edges). Hence, there is a finite, at most M , number of μ_i and for each one $0 < \mu_i < 1$, so that $\sigma_0^2 = \inf_i \mu_i(1 - \mu_i) > 0$, which in turn implies that $\lim_{N \rightarrow \infty} |\tilde{D}_N|^{-1} \sigma_N^2 = \lim_{N \rightarrow \infty} N^{-1} \sigma_N^2 > 0$.

On the original domain D_N (grid over the unit square), consider first the case of M nearest neighbors. The joint distribution of the $\{Z_i\}$ s over D_N is identical to that over \tilde{D}_N for each N , which shows that the same limit (3.18) holds. For the fixed radius neighborhood case, in order to make the joint distribution of the $\{Z_i\}$ s invariant to rescaling of the grid, select $r = r_N \rightarrow 0$ so that $r_N = O(\sqrt{N})$. A convenient choice is to set $N\pi r_N^2 = M = \text{const}$ as before, thus fixing the size of the neighborhoods not adjacent to the edges of the unit square. Then, the limit (3.18) holds for D-LVDF as well.

Random deployments. For the case of random deployments, we need a central limit theorem for a functional $H(\mathcal{X})$ of a marked binomial point process $\mathcal{X} = \{X_i =$

$(s_i, Y_i), i = 1, \dots, N\}$, where s_i are the random sensor locations and Y_i are the original i.i.d. decisions (the marks). A general CLT for unmarked binomial and Poisson processes was obtained in [89] and extended to marked point processes in [90]. The functional H must be translation-invariant and satisfy two conditions:

- (i) H is *strongly stabilizing*, which, informally speaking, means that the “add one cost” $\Delta(\mathcal{X}) = H(\mathcal{X} \cup \{0\}) - H(\mathcal{X})$, i.e., the change in H caused by inserting an extra point at the origin, is not affected by changes in \mathcal{X} that are far from the origin; and
- (ii) Certain moment bounds hold for $\Delta(\mathcal{X})$ and $H(\mathcal{X})$.

Here, we do not state or check these conditions formally, but they are easy to verify for LVDF: (i) is implied by the neighborhood structure, and (ii) follows from the fact that all the variables are binary. Rescaling to the unit square from an increasing domain is done in the same way as for regular grids.

3.3 Performance Evaluation

In this section we evaluate the proposed local vote decision fusion scheme. The models used to generate signals are:

$$(3.21) \quad \text{M1: } S_i = S_0 \exp(-\|s_i - v\|^2/\eta^2), \quad \text{M2: } S_i = \frac{S_0}{1 + (\|s_i - v\|/\eta)^3},$$

while measured energies are contaminated by Gaussian noise with mean zero and variance σ^2 . The main difference between the two models is that under M1 the signal decays exponentially fast, while under M2, polynomially fast. Notice that the parameter η scales the attenuation of the signal within the monitored region R and essentially determines the effective size of the target. These models capture the characteristics of physical processes. For example, M1 with its exponential rate is

most appropriate for temperature signals, due to Newton’s law of cooling. On the other hand, M2 has been used in the literature as a model for acoustic signals, with the exponent in the denominator ranging between 2 and 5 [64, 21]. Finally, for ease of interpretation in simulations, we give the values of the baseline signal strength S_0 and the signal-to-noise ratio expressed in dB units, given by $\text{SNR}=10 \log_{10}(S_0/\sigma)$.

Quality of the approximation: We start by examining the quality of the normal approximation of the network’s false alarm probability. The left panel of Figure 3.5 shows the approximation error for regular n by n grid designs, for $M = 9$ and $M = 5$ size neighborhoods for LVDF, together with ODF, which corresponds to $M = 1$. The true values of false alarm are based on 3,000 simulations of noise, with Monte Carlo error present in the 4th significant digit. The approximation for ODF is simply the normal approximation to the binomial distribution, and thus is the most accurate. As M increases, the dependencies among the decisions become stronger, and the quality of the approximation deteriorates. On the other hand, as the size n of the grid increases, the approximation improves. Nevertheless, the quality of the approximation remains very good even for moderate network sizes. A quantile-quantile plot of the calculated false alarm probability obtained from 3,000 simulations vs. those from a standard normal distribution for a 25×25 grid, with $\gamma = 0.2$ and the neighborhood size $M = 5$, is shown in the right panel of Figure 3.5. It can be seen that the approximation is very good throughout the domain of the distribution, with a slight deterioration in the tails.

We examine next the quality of the approximation for random deployments for two variants of LVDF (Table 3.1). In order to make the two methods comparable, for D-LVDF we set M to be the average number of points in a circle of radius r ; hence, $M = N(\pi r^2)$ and $r = \sqrt{M/(N\pi)}$.

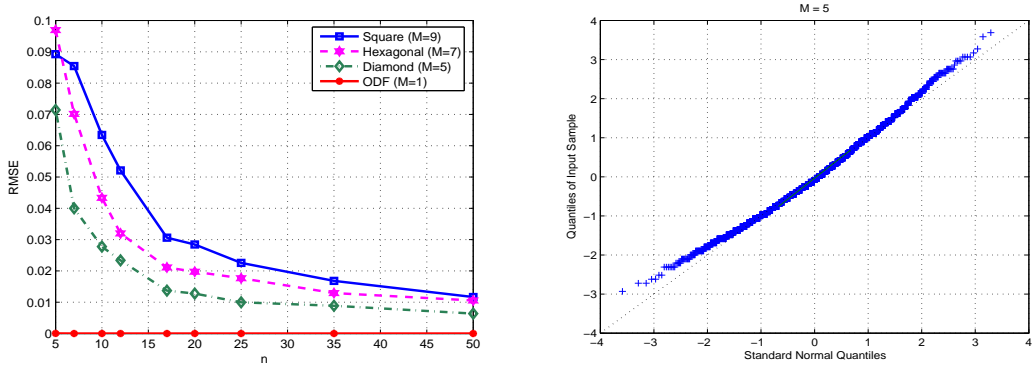


Figure 3.5: Left panel: Root mean squared error (RMSE) of the normal approximation of the system’s false alarm probability for grids with neighborhood sizes $M = 5, 9$, together with ODF, as a function of the grid size n (number of sensors $N = n^2$). The individual sensor’s false alarm $\gamma = 0.2$, and the RMSE is computed over the range of $F = 0 \dots 0.5$. Right panel: A plot of the calculated false alarm probability quantiles against the theoretical ones of a standard normal distribution for a 25×25 grid, with $\gamma = 0.2$ and $M = 5$.

Columns A1 and A2 in Table 3.1 give the RMSE of the general approximation formula (3.8) for D-LVDF and NN-LVDF, averaged over 100 random deployments (Monte Carlo error is in the 4th digit). The true false alarms for each deployment are obtained from 3000 noise simulations. The settings used are $M = 5$, $r = \sqrt{M/(N\pi)}$ for the size of the NN- and D-LVDF neighborhoods, $\gamma = 0.2$; the RMSE is computed over the range $F = 0 \dots 0.5$. As expected, the quality of the approximation improves for larger network sizes $N = n^2$, but the approximation is reasonable even for smaller networks.

The general approximation (3.8) depends on sensor locations, which we assume are known or can be obtained through a localization algorithm. However, examination of the approximation (3.8) shows that it depends on sensor locations only through the *distribution* of neighborhood sizes M_i and their intersections n_{ij} ; and, while the actual locations will change from deployment to deployment, the distribution of neighborhood sizes does not change much. Columns B1 and B2 in Table 3.1 show approximation errors obtained by computing the threshold (3.8) from a

n	D - LVDF			NN - LVDF		
	A1	B1	C1	A2	B2	C2
10	0.023	0.026	0.099	0.025	0.025	0.026
25	0.009	0.019	0.260	0.014	0.014	0.015
50	0.006	0.016	0.443	0.011	0.011	0.011

Table 3.1: RMSE for random deployments with $M = 5$. A: approximation (3.8); B: approximation (3.8) based on a single fixed deployment; C: approximation for a $M = 5$ size neighborhood on a regular grid (Monte Carlo error present in the 4th digit).

single arbitrary random deployment and then averaging the errors over 100 different random deployments; it can be seen that the differences are very small and therefore localization can be avoided in practice.

An even greater simplification would be to approximate the random deployment with a fixed grid. Columns C1 and C2 in Table 3.1 show that this approximation is reasonable for NN-LVDF but not D-LVDF. This is expected, since the distributions of neighborhood sizes are very similar for the grid and NN-LVDF (all $M_i = M$, so only the intersections vary), but not at all similar for D-LVDF and the grid.

Target detection probability: We examine next the probability of target detection D for a target located at the center of region R and a network comprised of 100 sensors. The schemes compared are ordinary decision fusion (ODF) and both variants of local vote decision fusion, D-LVDF and NN-LVDF, using $M = 9$ under random and square grid deployments. The signal for both models M1 and M2 is generated with $S_0 = 2$ and $\eta = 0.1$. The individual sensor's false alarm probability is set to $\gamma = 0.2$ and that of the system to $F = 0.1$. The differences between D-LVDF and NN-LVDF for these settings are so small that their respective curves in Figures 3.6 and 3.7 are practically indistinguishable; hence, only those corresponding to D-LVDF are shown. The plots, based on averages from 1000 simulations, show that for all SNRs, LVDF outperforms ODF, with the detection probability exceeding in general 0.9 for $\text{SNR} \geq 10$ for model M1 and for $\text{SNR} \geq 7$ for M2. This is expected in light of

the polynomial decay of tails in the latter model. In general, detection is lower for more localized signals (target following M1 is harder to detect than that following M2, and for each model smaller η results in lower detection), but the corresponding gains from LVDF are larger. Therefore, the local vote schemes are particularly beneficial for small targets or signal with low attenuation. On the other hand, ODF does not prove to be competitive in the cases considered above. Under the above settings,

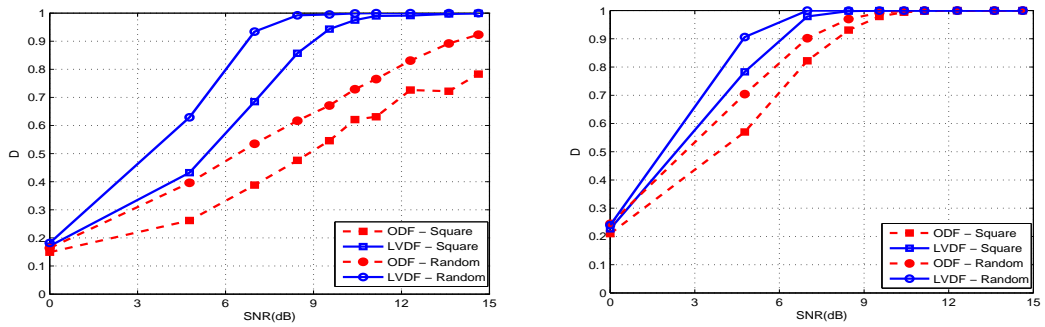


Figure 3.6: Probability of target detection as a function of SNR (in dB) for models M1 (left panel) and M2 (right panel).

the receiver operating characteristic (ROC) curves for these decision fusion schemes are shown in Figure 3.7 for fixed $\text{SNR} = 7$. Once again, the local vote schemes clearly outperform ordinary decision fusion, with larger gains obtained for more localized targets (M1). Specifically, for values of F in the range 0-0.2 that are most relevant in practice, the gains in detection probability exceed .3 for both models and both types of deployment.

It is worth noting that both LVDF and ODF exhibit a superior performance under random deployments. This is due to the discretization of the grid – our isotropic signal models generate a circular target, which is covered by fewer sensors from a square grid than from a random deployment. The differences become negligible for denser networks ($N \geq 400$ – results not shown here).

The results shown in Figures 3.6 and 3.7 are consistent across a range of the γ

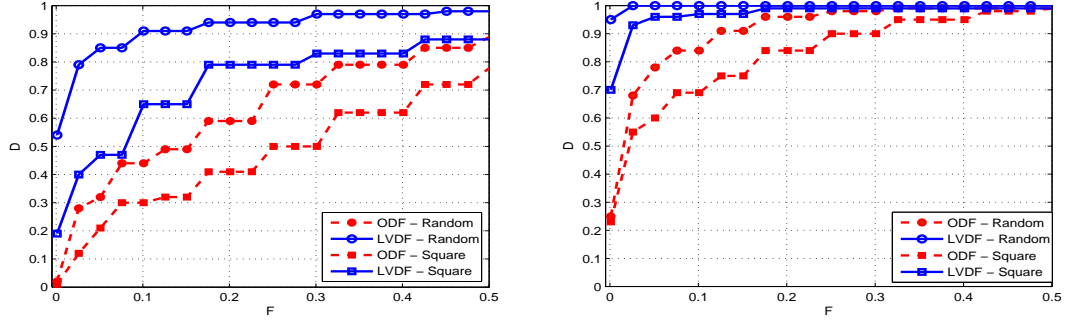


Figure 3.7: Probability of detection as a function of the system-wide false alarm probability F (ROC curve) for models M1 (left panel) and M2 (right panel).

and η parameters. The plots (Figure 3.8), generated for $\eta = 0.08$, also show that LVDF outperforms ODF, with the detection probability exceeding in general 0.9 for $\text{SNR} \geq 10$. As expected, the performance deteriorates for more localized signals ($\eta = 0.08$) as compared to $\eta = 0.1$, but the corresponding gains from LVDF are larger. Therefore, the local vote schemes are particularly beneficial for small targets (lower values of η).

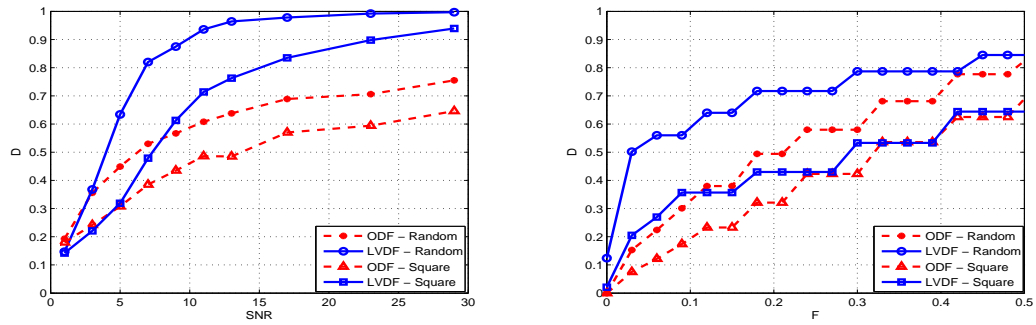


Figure 3.8: Probability of detection as a function of SNR for signal decay parameter (left panel) and the system-wide false alarm probability F (ROC curve) (right panel) for $\eta = 0.08$.

Energy Savings: The exact amount of energy savings will depend on the system set-up and the cost of in-network transmissions relative to transmissions to fusion center. However, since LVDF leads to better detection regardless of whether it can provide energy savings, it can be performed at the fusion center after the original decisions are transmitted. If transmissions to fusion center are more expensive, then

the energy savings potential of LVDF over ODF can be assessed from the number of positive decisions transmitted to the fusion center. These are shown in Fig. 3.9 for both algorithms as a function of α and η for model M1 (the pattern for M2 is very similar).

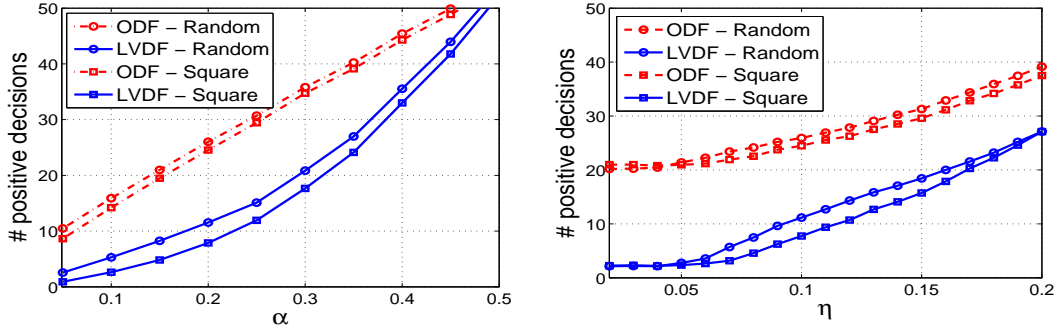


Figure 3.9: Number of positive decisions as a function of the sensor's false alarm α (left panel, with $\eta = 0.1$) and as a function of the signal decay parameter η (right panel, with $\alpha = 0.2$) for model M1 with $F = 0.1$ and $SNR = 7$.

System design: We examine next the sensitivity of the target detection probability to the size of the local vote neighborhood, since this parameter can impact the operational characteristics of the sensor network. The setting is as follows: the target is located in the center of the region R monitored by 100 sensors deployed on a square grid, with individual false alarm probabilities $\gamma = 0.2$. The signal is generated by model M1 with $S_0 = 2$, $SNR=7$ and $\eta \in \{0.05, 0.10, 0.15\}$. In Figure 3.10, the probability of target detection D as a function of M is shown for these three levels of the signal decay parameter η . It can be seen that when the signal is highly localized ($\eta = 0.05$) the detection probability does not exceed 0.5 and is maximized for small values of M , whereas when the signal diffuses across R , the detection probability is almost 1 for almost all values of M examined. For $\eta = 0.1$, the optimal neighborhood size is around 10, and the drop in D for larger values of M occurs when the size of the neighborhood becomes larger than the target. In this

case, only a few sensors in the neighborhood pick up the signal and thus the majority vote tends to indicate the target is absent. A similar pattern occurs for model M2.

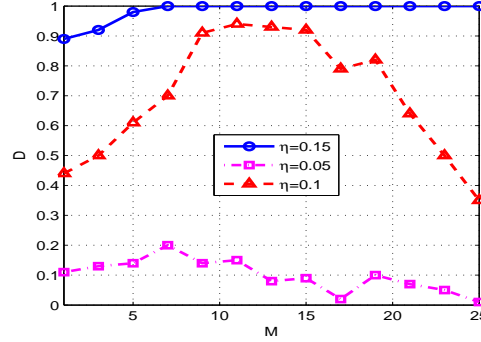


Figure 3.10: Target detection probability for various levels of signal decay η .

These plots indicate that when designing a wireless sensor network employing a LVDF mechanism, one should choose the size of the neighborhood comparable to the size of the smallest target one is interested in detecting, since large targets will be easy to spot.

3.4 Temporal Decision Fusion

In previous sections, the problem of decision fusion for target detection by a sensor network has been studied when only one set of energy measurements is available. However, target detection is often performed continuously over time, as the sensor network collects information and updates its decisions. We examine next the problem of temporal decision fusion. We assume that at every time slot $t = 0, 1, 2, \dots$ the sensors obtain energy measurements and make decisions, and extend the proposed local vote decision fusion framework to decisions made over time.

Denote by Z_i^t the decision of sensor i at time t corrected according to the proposed LVDF scheme. The values Z_i^t are stored and processed by the fusion center. We propose combining decisions over time with an exponentially weighted moving

average with parameter λ , $0 \leq \lambda \leq 1$: for all $i = 1, \dots, N$, let $Y_i^0 = Z_i^0$ and

$$(3.22) \quad Y_i^t = \lambda Z_i^t + (1 - \lambda)Y_i^{t-1}, \quad t = 1, 2, \dots$$

The parameter λ determines how much weight to give to the present vs. the past: in slowly changing environments or for slowly moving targets smaller λ will be better, and vice versa. The decision of the network at time t is given by $\tilde{D}_t = I\left(\sum_{i=1}^N Y_i^t \geq \tilde{T}_\ell\right)$. As before, the goal becomes to obtain an approximation for the system-wide false alarm probability that allows us to calculate \tilde{T}_ℓ . We examine the case where the energy measurements E_i and consequently the corrected decisions Z_i^t are independent over time; the case where the Z_i^t s are correlated over time, due to correlated noise in the energy measurements, is studied through simulation in Section 3.5.

Straightforward algebra shows that, for all $i, j = 1 \dots N$, and all $t = 1, 2, \dots$, $E(Y_i^t) = E(Z_i^t) = \mu_i$, and

$$(3.23) \quad \text{Cov}(Y_i^t, Y_j^t) = \lambda^2 \text{Cov}(Z_i^t, Z_j^t) + (1 - \lambda)^2 \text{Cov}(Y_i^{t-1}, Y_j^{t-1}).$$

Iterating the recursive covariance formula (3.23) we obtain

$$(3.24) \quad \text{Cov}(Y_i^t, Y_j^t) = ((1 - \lambda)^{2t} + \lambda^2 \sum_{k=0}^{t-1} (1 - \lambda)^{2k}) \text{Cov}(Z_i^k, Z_j^k),$$

and letting $t \rightarrow \infty$ we finally get, for large t ,

$$(3.25) \quad \text{Cov}(Y_i^t, Y_j^t) = \frac{\lambda}{2 - \lambda} \text{Cov}(Z_i^0, Z_j^0).$$

The geometric series converge very quickly for reasonable values of λ , at the rate $\mathcal{O}(1 - \lambda)^{2t}$. The approximation formula for the system-wide false alarm probability F is given by

$$(3.26) \quad F \approx 1 - \Phi\left(\frac{\tilde{T}_\ell - \sum_{i=1}^N \mu_i}{\sqrt{\frac{\lambda}{2 - \lambda} \sum_{i=1}^N \sum_{j=1}^N \text{Cov}(Z_i^0, Z_j^0)}}\right)$$

3.5 Performance Evaluation for Temporal LVDF

We examine next the performance of temporal LVDF in terms of target detection, using the approximation (3.26). The results shown are based on a network comprised of $N = 100$ sensors, with the signal generated by model M1, the energy by $E_i^t = S_i + \epsilon_i^t$ and a stationary target located at the center of the region \mathcal{R} . Two scenarios are studied and compared: (1) Independent noise over time; i.e., $\text{Cov}(\epsilon_i^t, \epsilon_i^{t-1}) = 0$, for all time slots t , with the noise $\epsilon_i^t \sim \mathcal{N}(0, \sigma^2)$ as before; (2) Temporally correlated noise according to the autoregressive model $\epsilon_i^t = \rho\epsilon_i^{t-1} + \xi_i$, with $\xi_i \sim \mathcal{N}(0, (1-\rho^2)\sigma^2)$, and $\epsilon_i^t \sim \mathcal{N}(0, \sigma^2)$ as before.

In Figure 3.11, ROC curves based on 1000 simulations for temporal LVDF under random deployment are shown for uncorrelated and correlated noise for time slots $t = \{0, 1, 2, 10\}$. The settings are $S_0 = 2$, $\eta = 0.1$, $\text{SNR} = 7$, $\gamma = 0.2$, $M = 9$, and $\lambda = 0.5$. As expected, for uncorrelated noise (left panel of Figure 3.11) the detection probability improves uniformly over time (given that the target is stationary) with gains of over 15% for small false alarm rates. More substantial gains are obtained for smaller targets (smaller values of η – results not shown here). It is interesting to note that most of the improvement occurs over the first two time slots, with rather minimal gains at subsequent times. Qualitatively, the situation is analogous if different weight factors λ are used, as well as in the case of square grid deployment.

For correlated noise, the ROC curves for time slot $t = 10$, under the above system settings and for values of $\rho = 0$ (uncorrelated noise), 0.4 and 0.9 are shown in the right panel of Figure 3.11. It can be seen that the performance degrades for larger values of the temporal correlation, a result which is consistent for other time slots as well. Nevertheless, for moderate values of ρ , the decrease in the detection probability is

rather small, especially for smaller false alarm rates; this suggests that the procedure is fairly robust and hence useful in practice for weak temporal dependence scenarios.

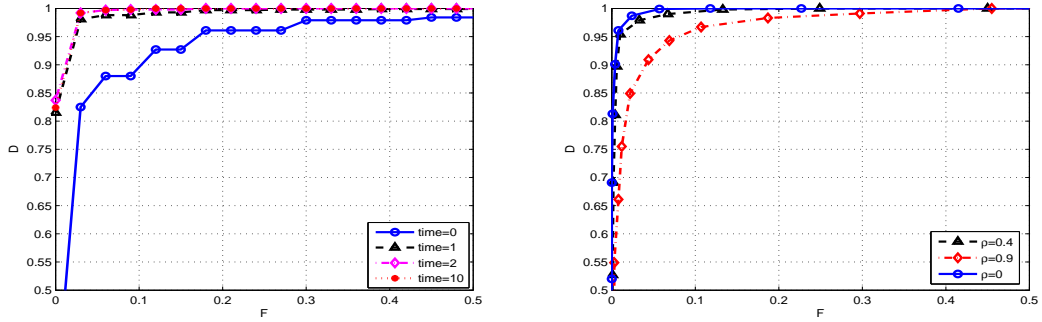


Figure 3.11: Temporal LVDF probability of detection for different time slots and uncorrelated noise (left panel), and for a fixed time slot with different values of the correlation coefficient for correlated noise (right panel).

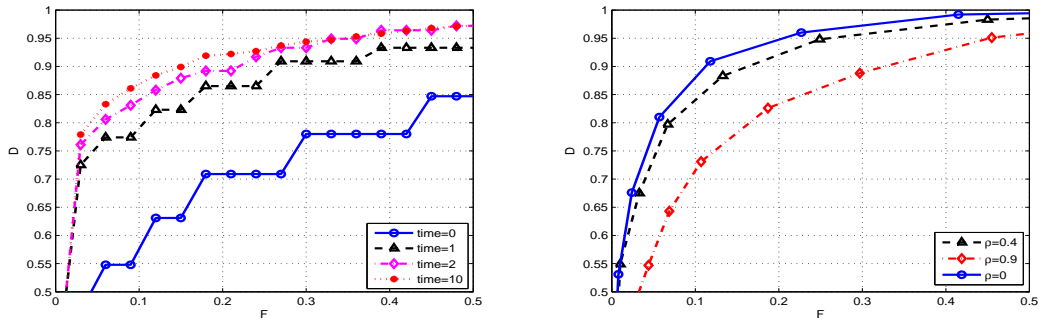


Figure 3.12: Temporal LVDF probability of detection for different time slots and uncorrelated noise (left panel), and for a fixed time slot with different values of the correlation coefficient for correlated noise (right panel).

We examine next the performance of temporal LVDF for different values of the weight parameter λ for moving targets across the monitored region R . Specifically, a 100-sensor network is deployed along a grid and a target crosses R horizontally moving west to east. The target's signal is generated by model M1 with $S_0 = 2$, $\eta = 0.1$, SNR= 7 and the monitoring period is 30 time units. The target appears at $t = 10$ and two scenarios for the target's speed are considered: in the first one, the target moves one hop every time unit and it takes 10 time periods to cross R ;

in the second scenario, the target moves 3 hops per time unit and it takes 3 time periods to cross R . The detection probability over the monitoring period for these two scenarios and for different values of λ is shown in Figure 3.13. It can be seen that for both slow (left panel) and fast (right panel) moving targets, the pattern is similar across λ : the detection probability rises fast once the target appears, stabilizes during its presence in R and fades once the target departs. The increase in the detection probability is similar for most values of λ with the exception of $\lambda = 0.2$, where a small lag is observed. On the other hand, there are substantial differences once the target departs; for example, for $\lambda = 0.2$ the detection probability fades slowly, since most of the weight is placed in past decisions, while for $\lambda = 1$ where all the weight is on the current decision, it drops to the false alarm rate almost immediately. Finally, one can conclude that the intermediate value of $\lambda = 0.5$ is a reasonable and robust choice for most situations, since it is sensitive to both the emergence and departure of the target, and achieves a very high detection probability.

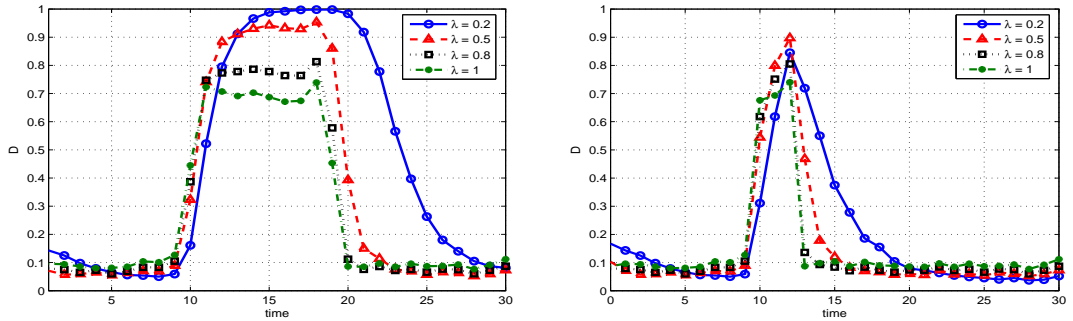


Figure 3.13: Temporal LVDF probability of detection for slow (left panel) and fast moving targets (right panel) under a grid deployment.

Remark: The scenario involving a moving target could be formulated as a change-point problem, especially under the hypothesis testing framework discussed in the Introduction. This formulation has been studied extensively, and optimal procedures can be derived as in, e.g., [105, 106].

CHAPTER IV

Localization and Tracking of a Single Target with Wireless Sensor Networks

4.1 Introduction

In this chapter we focus on the tasks of a single target localization and tracking. The canonical signal processing problems of target localization have received an increasing degree of attention over the last few years. The literature on these problems goes back to radar systems (see e.g. [1]), where localization was mostly performed via beam-forming methods. Existing localization algorithms for WSNs can be divided into two general classes: those based on energy readings E_i [64, 99, 52, 12] and those based on binary decisions Y_i [82, 85, 33]. In [64], non-linear least squares are used to localize the target, assuming an isotropic exponentially decaying signal model. For acoustic energy measurements, a maximum likelihood (ML) estimation method based on the Expectation-Maximization (EM) algorithm and a projection solution for the problem of target localization was proposed in [99]. The EM algorithm was used to fit the mixture model for energies coming from multiple targets. These methods proved to be more accurate than nonlinear least squares estimates, but more computationally intensive. Compared to techniques that depend on such physical variables as direction of arrival (DOA) and time delay of arrival (TDOA)

[52, 16, 78], energy based methods do not require a very accurate synchronization among the sensors and provide accurate estimation of target locations. However, energy-based techniques require transmission of real value data from all the sensors, which may not be feasible under a set of communication constraints. Moreover, methods in [99] require transmission of the mean and variance of the background noise, which often are unknown and may have to be estimated together with the target location and signal amplitude. Options for reducing communications cost to some extent include implementation of an optimization-based localization algorithm in a distributed manner [12], or obtaining energy information only from cluster heads rather than all sensors [128]. Specifically, in [128], a two-step communication protocol between the cluster head and the sensors within the cluster is used: sensors send binary decisions to the cluster head, which determines candidate target locations and sends requests for stored energy readings to selected sensors.

Binary decision transmission offers significant cost savings, since only positive one-bit detection notifications are sent to the fusion center. In [82], maximum likelihood target location estimation from binary and multi-bit discrete data was developed, along with the corresponding Cramer-Rao bound. The MLE approach reduces the problem of localization to that of non-linear function optimization, which may suffer from existence of local maxima, slow convergence and high computational complexity. In [85], an improved MLE approach using the same likelihood as in [82] maximized by particle swarm optimization techniques was shown to outperform deterministic quasi Newton-Raphson schemes. Another recent approach used distributed false discovery rate to select the most informative sensors to communicate with the fusion center, although it relies on multiple within network communications on each step of the localization [33]. Other related papers include target localization in relation to the

coverage problem [116], and a Bayesian approach to target localization and sensor selection and placement [117].

In this chapter, we develop a target localization technique based on binary decisions; the original decisions Y_i and corrected decisions that incorporate information from neighboring sensors through Local Vote Decision Fusion (LVDF) proposed in Chapter III (described also in [54]). The main effect of LVDF is de-noising the original decisions, which was shown to give a robust procedure for target detection, particularly in noisy environments with low signal-to-noise ratio [54]. However, the corrected decisions are correlated, which makes likelihood computations intractable. Instead, we adopt a pseudo-likelihood approach, and develop a localization and signal estimation procedure for LVDF that enjoys the same robustness properties as the LVDF detection algorithm. We also derive an EM algorithm for maximum likelihood estimation from binary decisions, both for the original decisions Y_i and the LVDF-corrected ones. We show that, in both cases, EM provides much more accurate estimates than direct optimization of the likelihood of binary decisions, but has higher computation complexity. We also discuss properties of the estimators and provide a bootstrap procedure for uncertainty assessment.

In addition to the proposed methods above and under the assumption of known energy readings from sensors with positive original decisions Y_i or corrected decisions Z_i , we develop hybrid (pseudo-) maximum likelihood and expectation-maximization algorithms. Numerical results show that the proposed framework significantly improves the accuracy in target location estimation, as well as signal magnitude estimation. Simulations using different signal models are performed and the problem of model and noise distribution misspecification is explored. We also investigate the advantage of applying the Largest Connected Component technique to define sub-

regions of the candidate target locations. Extensions to tracking a target over time are considered as well.

The remainder of the chapter is organized as follows. In Section 4.2 the likelihood framework and the EM algorithm for binary decisions are developed, and properties of these estimates and construction of confidence intervals are discussed. In Section 4.3, numerical results on the performance of the various methods are presented, along with a discussion of implementation issues. The problem of tracking a target over time is briefly examined in Section 4.4.

4.2 Methods and Algorithms

In this section we present methods for target localization by a WSN deployed over a region R , based on binary measurements – either the original decisions Y_i , $i = 1, 2, \dots, N$ or LVDF-corrected decisions Z_i . The maximum likelihood estimator for location from Y_i has been derived before (see e.g. [82]), but we include a summary for completeness.

It is assumed that the sensor locations s_i are known and that the attenuation of the target's signal is a known function which is monotonically decreasing in the distance from the target $\delta_i(v) = \|s_i - v\|$, and also depends on an attenuation parameter η . That is, the signal at location s_i is given by

$$(4.1) \quad S_i(v) = S_0 C_\eta(\delta_i(v)),$$

with $C_\eta(0) = 1$ and $S_0 \in [0, \infty)$ denoting the signal strength at the target's location v . The noise is assumed to be Gaussian with mean zero and variance σ^2 . The primary parameters of interest are the target's location v and the signal strength S_0 ; obviously, the noise variance σ^2 and the attenuation parameter η affect the estimation problem.

4.2.1 Localization from Original Decisions

As outlined above, each sensor makes a decision $Y_i \in \{0, 1\}$ regarding the presence of the target in R . We do not treat the signal itself as random, so the only randomness comes from the i.i.d. noise. Define the vector of unknown parameters $\theta = (v_x, v_y, S_0, \sigma, \eta)$. Then decisions $\{Y_i\}$ are independent Bernoulli random variables with probability of success given by

$$\mathbb{P}(Y_i = 1) \equiv \alpha_i(\theta) = 1 - \Phi(A_i(\theta)),$$

where $\Phi(\cdot)$ denotes the Gaussian cumulative distribution function and $A_i(\theta)$ is the standardized excess energy level given by

$$A_i(\theta) = \frac{\tau - S_0 C_\eta(\delta_i(v))}{\sigma}.$$

The log-likelihood function of $\{Y_i\}$ is given by:

$$(4.2) \quad \ell_Y(\theta) = \sum_{i=1}^N [Y_i \log \alpha_i(\theta) + (1 - Y_i) \log(1 - \alpha_i(\theta))].$$

There are two options for obtaining estimates of the unknown parameters: direct numerical maximization of the log-likelihood function (5.6) (no closed form solution exists) or the Expectation-Maximization (EM) algorithm [26]. The EM can be applied here by viewing $Y_i = I(E_i > \tau)$ as incomplete data on the true energy readings E_i , and consists of an expectation step (E-step), where expected likelihood of the full data conditional on the available data is obtained, and a maximization step (M-step) where the parameters are estimated by maximizing the likelihood from the E-step. The full log-likelihood of the energies, up to an additive constant, is given by

$$(4.3) \quad \ell_E(\theta) = -\frac{n}{2} \log \sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^N [E_i - S_0 C_\eta(\delta_i(v))]^2.$$

Maximizing this over S_0 and σ^2 can be done in closed form. This gives the

M-step:

$$(4.4) \quad \hat{S}_0 = \frac{\sum_{i=1}^N E_i C_\eta(\delta_i(v))}{\sum_{i=1}^N C_\eta^2(\delta_i(v))},$$

$$(4.5) \quad \hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^N (E_i - \hat{S}_0 C_\eta(\delta_i(v)))^2.$$

The other parameters (v and η) are found by numerical optimization of (4.3) with (4.4) and (4.5) plugged in.

The likelihood (4.3) is a curved exponential family in θ , so the M-step shows that there are just two quantities that need to be computed in the E-step: $\hat{E}_i = \mathbb{E}[E_i|Y] = \mathbb{E}[E_i|Y_i]$ and $\hat{E}_i^2 = \mathbb{E}[E_i^2|Y] = \mathbb{E}[E_i^2|Y_i]$. Note that each E_i only depends on Y_i rather than all Y because S_i is not random, and ϵ_i 's are independent. These expectations are straightforward to derive: for instance,

$$(4.6) \quad \begin{aligned} \mathbb{E}[E_i|Y_i = 0] &= \frac{\int_{-\infty}^{\tau} x p_{E_i}(x) dx}{\int_{-\infty}^{\tau} p_{E_i}(x) dx} = \\ &= S_0 C_\eta(\delta_i(v)) - \frac{\sigma \exp\left(-\frac{A_i(\theta)^2}{2}\right)}{\sqrt{2\pi} \Phi\left(-\frac{A_i(\theta)^2}{2}\right)} \end{aligned}$$

Combining analogous computations for $\mathbb{E}[E_i|Y_i = 1]$ and $\mathbb{E}[E_i^2|Y_i]$ gives the

E-step:

$$(4.7) \quad \hat{E}_i = S_0 C_\eta(\delta_i(v)) + \frac{\sigma \exp\left(-\frac{A_i(\theta)^2}{2}\right)}{\sqrt{2\pi}} B_i(\theta, Y)$$

$$(4.8) \quad \hat{E}_i^2 = S_0 C_\eta(\delta_i(v))(\tau - \hat{E}_i) - \hat{E}_i \tau,$$

where

$$B_i(\theta, Y) = \frac{Y_i - 1}{\Phi\left(-\frac{A_i(\theta)^2}{2}\right)} + \frac{Y_i}{1 - \Phi\left(-\frac{A_i(\theta)^2}{2}\right)}.$$

The EM algorithm consists of iterating between the E-step and the M-step until convergence, which tends to be computationally more expensive than direct numerical

optimization of the likelihood; however, it tends to produce much more accurate results (see Section 4.3). In both cases, good initial values for the parameters are important; we briefly discuss this issue in Section 4.3.

4.2.2 Localization from LVDF Decisions

The adjusted decisions Z_i produced by the LVDF algorithm are correlated, which renders the form of the likelihood function in (5.6) invalid. For this reason we adopt a pseudo-likelihood formulation [10], by assuming that all adjusted decisions Z_i are *independent*. Further, we make a simplifying

Assumption: For neighbors $j \in U(i)$, $\mathbb{P}(Y_j = 1) \approx \mathbb{P}(Y_i = 1)$.

Letting $\beta_i(\theta) = \mathbb{P}(Z_i = 1)$, this gives

$$\beta_i(\theta) = \mathbb{P}\left(\sum_{j \in U(i)} Y_j \geq \frac{M}{2}\right) \approx \sum_{k=\lceil M/2 \rceil}^M \binom{M}{k} \alpha_i^k (1 - \alpha_i)^{M-k}.$$

The pseudo-loglikelihood function for the adjusted decisions Z_i is given by:

$$(4.9) \quad \ell_Z(\theta) = \sum_{i=1}^N [Z_i \log \beta_i(\theta) + (1 - Z_i) \log(1 - \beta_i(\theta))].$$

Maximum likelihood estimates based on (5.9) can again be obtained through direct maximization. For the EM algorithm, the M-step is the same as before. The E-step requires calculating the first and second conditional moments $\mathbb{E}[E_i|Z]$ and $\mathbb{E}[E_i^2|Z]$.

We first compute the conditional distribution of E_i given all the decisions Z . Write

$$(4.10) \quad \begin{aligned} \mathbb{P}[E_i|Z] &= \frac{1}{\mathbb{P}(Z)} \sum_{k=0,1} \mathbb{P}(E_i, Z|Y_i = k) \mathbb{P}(Y_i = k) = \\ &= \frac{1}{\mathbb{P}(Z)} \sum_{k=0,1} \mathbb{P}(E_i|Y_i = k) \mathbb{P}(Z|Y_i = k) \mathbb{P}(Y_i = k) \end{aligned}$$

where the last equality follows because conditional on the value of Y_i the energy reading E_i is independent of the vector of corrected decisions Z (recall again that all

randomness comes from the noise ϵ_i , not the signal). Integrating (4.10) gives

$$(4.11) \quad \mathbb{E}[E_i|Z] = \sum_{k=0,1} \mathbb{E}(E_i|Y_i = k)\mathbb{P}(Y_i = k|Z)$$

$$(4.12) \quad \mathbb{E}[E_i^2|Z] = \sum_{k=0,1} \mathbb{E}(E_i^2|Y_i = k)\mathbb{P}(Y_i = k|Z)$$

Since we have already obtained $\mathbb{E}[E_i|Y_i]$ and $\mathbb{E}[E_i^2|Y_i]$ in the E-step for ODF, all that remains to be calculated is

$$(4.13) \quad \mathbb{P}(Y_i = 1|Z) = \frac{\mathbb{P}(Y_i = 1)\mathbb{P}(Z|Y_i = 1)}{\mathbb{P}(Z)}$$

$$(4.14) \quad \approx \alpha_i \prod_{j=1}^N \frac{\mathbb{P}(Z_j|Y_i = 1)}{\mathbb{P}(Z_j)}$$

$$(4.15) \quad = \alpha_i \prod_{j:i \in U(j)} \frac{\mathbb{P}(Z_j|Y_i = 1)}{\mathbb{P}(Z_j)},$$

where (4.13) is the Bayes rule, (4.14) is the pseudo-likelihood approximation, and (4.15) follows because only corrected decisions that come from a neighborhood containing sensor i depend on Y_i . Once again using the assumption $\alpha_j \approx \alpha_i$ for $j \in U(i)$, we get

$$(4.16) \quad \begin{aligned} \tilde{\beta}_{ji} &= \mathbb{P}(Z_j = 1|Y_i = 1) = \mathbb{P}\left(\sum_{k \in U(j), k \neq i} Y_k \geq \frac{M}{2} - 1\right) \\ &\approx \sum_{q=[M/2-1]}^{M-1} \binom{M-1}{q} \alpha_j^q (1 - \alpha_j)^{M-1-q} \end{aligned}$$

and finally

$$(4.17) \quad \mathbb{P}(Y_i = 1|Z) = \alpha_i \prod_{j:i \in U(j)} \left(\frac{\tilde{\beta}_{ji}}{\beta_j}\right)^{Z_j} \left(\frac{1 - \tilde{\beta}_{ji}}{1 - \beta_j}\right)^{1-Z_j}$$

Substituting (5.10) into (4.11) and (4.12) completes the E-step for the LVDF decisions.

4.2.3 Hybrid Maximum Likelihood Estimates

Hybrid maximum likelihood estimation is motivated by the trade-off between energy consumption and accuracy of target localization. Maximum likelihood estimates based on the energy measurements E_i encompass all available information, and are considered to be the most accurate. Decision based estimates contain less information (although in low SNR environments they tend to perform better due to relative robustness of binary decisions). However, transmission of one bit decisions offers significant communication cost savings. The main idea of hybrid methods is to use energy information from the sensors with positive decisions and model energies for the rest of the network. By using energy readings from sensors with positive initial or updated decisions, we both reduce significantly communication cost compared to transmission of the full energy measurements from all sensor nodes and improve the decision based localization.

Hybrid expectation maximization (HEM) algorithm is an extension of the original EM algorithm. Since each E_i only depends on Y_i , for ODF, hybrid EM formulas are given as:

If $Y_i = 1$,

$$(4.18) \quad \hat{E}_i = E_i,$$

$$(4.19) \quad \hat{E}_i^2 = E_i^2.$$

Otherwise ($Y_i = 0$),

$$(4.20) \quad \hat{E}_i = S_0 C_\eta(\delta_i(v)) - \frac{\sigma \exp\left(-\frac{A_i(\tau)^2}{2}\right)}{\sqrt{2\pi}\Phi\left(-\frac{A_i(\tau)^2}{2}\right)}$$

$$(4.21) \quad \hat{E}_i^2 = S_0^2 C_\eta^2(\delta_i(v))(\tau - \hat{E}_i) - \hat{E}_i \tau.$$

Analogously, for LVDF hybrid EM version, we model only the energies that corre-

spond to $Z_i = 0$ using (5.10),(4.11), (4.12) and use the available energies for $Z_i = 1$.

Although HEM proves to be competitive in terms of the accuracy of localization and less computationally expensive than the original EM algorithm, it sometimes fails to converge. Another option is to replace energies corresponding to zero decisions by the threshold τ and maximize the energy-based likelihood (4.3), which avoids iterative computations, but suffers in accuracy of location and signal estimates. We refer to this method as hybrid maximum likelihood estimation (HML).

4.2.4 Properties of Maximum Likelihood Estimates

We briefly discuss the properties of the ML and EM estimates for the ODF and the LVDF mechanisms. Under the following assumptions on the log-likelihood function [27]: (i) the log-likelihood function is distinct when $\theta_1 \neq \theta_2$, (ii) the true parameter θ_0 is in the interior of the parameter space and (iii) the log-likelihood function is differentiable in θ , the estimate $\hat{\theta}$ is *consistent*. Since (ii) is straightforward, we show that both log-likelihood functions (5.6) and (5.9) satisfy (i) and (iii).

(i) Let $\theta_1 \neq \theta_2$, where $\theta_j = (S_0^j, v^j)$, $j = 1, 2$, and let $I(\theta_1, \theta_2) = \{i : A_i(\theta_1) \neq A_i(\theta_2)\} \subset \{1, 2, \dots, N\}$.

The set I is not empty, because otherwise it would follow that $S_0^1 C_\eta(\delta_i(v^1)) = S_0^2 C_\eta(\delta_i(v^2))$ for all i , and $\theta_1 = \theta_2$, which contradicts the assumption above.

Therefore, $\alpha_i(\theta_1) \neq \alpha_i(\theta_2)$ for $i \in I$ and hence $\ell_Y(\theta_1) \neq \ell_Y(\theta_2)$. A similar argument applies to LVDF log-likelihood function ℓ_Z . Assumption (iii) follows from the fact that the log-likelihood function ($\ell_Y(\theta)$ or $\ell_Z(\theta)$) is a continuous and bounded function.

Under additional assumptions on the log-likelihood function and its derivatives, asymptotic normality of the estimates can also be established, which can be used to provide a measure of uncertainty for the estimates. Whether these assumptions hold

will depend on the exact form of the signal decay function C . The EM algorithm will converge to a local maximum of the energy likelihood (4.3); additional properties can be established depending on the function C .

Assuming the assumptions on the likelihood function that guarantee asymptotic normality hold, one can obtain confidence regions for the parameters of interest. We show next how to construct a two-dimensional confidence region for the main parameter of interest, target location v . Let $\hat{v} = (\hat{v}_x, \hat{v}_y)$ be the coordinates of the estimate of the true target location, with $\hat{v} \sim \mathcal{N}(v, \Sigma_v)$, with $\Sigma_v = \text{Var}(\hat{v})$. A two-dimensional confidence region Q satisfies $\mathbb{P}(v \in Q) = 1 - \zeta$, with $1 - \zeta$ denoting the confidence level. Standardizing the location estimate yields

$$(4.22) \quad \tilde{v} = \Sigma_v^{-1/2}(\hat{v} - v) \sim \mathcal{N}(0, I_2),$$

which in turn implies that the desired confidence region \tilde{Q} for \tilde{v} is a circle of radius r that satisfies $\mathbb{P}(\|\tilde{v}\|^2 \leq r^2) = 1 - \zeta$. The appropriate value of r is given by the $(1 - \zeta)$ -quantile of the χ^2 distribution with two degrees of freedom. The region \tilde{Q} can then be inverted to obtain Q using (4.22).

This procedure requires an estimate of the covariance matrix $\Sigma = \text{Var}(\hat{\theta})$. The asymptotic Cramer-Rao bound may be used to estimate Σ via the Fisher information matrix, but it may not be sufficiently accurate for smaller samples, particularly for the pseudo-likelihood.

An alternative way of obtaining a measure of variability of the estimates is through a parametric bootstrap procedure [31], as follows.

(i) Energies are simulated from the posited model with parameters set to the maximum likelihood estimates: simulate M samples from the assumed signal attenuation model to obtain

$$E_{i,m}^* = \hat{S}_0 C_{\hat{\eta}}(\delta_i(\hat{v})) + \epsilon_{i,m}^*,$$

where $\epsilon_{i,m}^* \sim \mathcal{N}(0, \hat{\sigma}^2)$ are i.i.d. noise, $i = 1, \dots, N$, $m = 1, \dots, M$.

(ii) The simulated energies are used to obtain bootstrap estimates of the parameters of interest $\hat{\theta}_m$, $m = 1, \dots, M$.

(iii) The empirical covariance of the estimates $\hat{\theta}_m$ across the M samples gives an estimate for Σ .

This is the procedure we followed in obtaining uncertainty estimates presented in the next section.

4.3 Performance Evaluation

We examine next the performance of the proposed algorithms, focusing on the target location v and the signal amplitude S_0 . In order to limit the number of comparisons, it is assumed that the parameters η and σ^2 are known, although the proposed likelihood framework allows their estimation.

The same models as in Chapter III, were used to generate the target's underlying signal:

$$\text{M1: } S_i = S_0 \exp(-\|s_i - v\|^2/\eta^2), \quad \text{M2: } S_i = \frac{S_0}{1 + (\|s_i - v\|/\eta)^3}.$$

In the simulation study, the signal at each sensor location s_i was contaminated by mean zero, variance σ^2 Gaussian noise and the following procedures were compared: the maximum likelihood estimates based on the original decisions Y_i (ODF) obtained through direct optimization and through the EM algorithm, as well as the hybrid versions, denoted by ML(Y), EM(Y), HML(Y) and HEM(Y), respectively, and the corresponding estimates based on the adjusted decisions (LVDF), denoted by ML(Z), EM(Z), HML(Z) and HEM(Z). In addition, maximum likelihood estimates based on the measured energies (ML(E)) were obtained to serve as a 'gold standard' for

comparison purposes. Their computation is equivalent to the M-step in the EM algorithms, except real rather than expected energies are used.

Throughout this section, the results will be shown for either a WSN deployed on a 20×20 grid in the unit square, or 400 sensors deployed at random (uniformly distributed in the unit square). We fix the same arbitrarily selected random deployment throughout; additional simulations confirmed that the results are very similar if we average over many random deployments instead. The true target location is set to $v = (1/4, 1/4)$, $S_0 = 2$, the individual sensor's false alarm $\gamma = 0.1$, and the network's false alarm $F = 0.1$. The value of σ is determined from the selected value of the signal-to-noise ratio $\text{SNR} = S_0/\sigma$, which varied from 2 to 10. The target "size" parameter η ranged from 0.03 to 0.15. Note that if the target is large enough, it is usually detected by several sensors. On the other hand, if the sensor is so small that only one or two sensors can detect it, there is no advantage using local voting and localization accuracy could be fairly low. In such a situation, the density of the sensor network needs to increase to compensate.

4.3.1 Detection Performance

Notice that from the detection point of view, there are only three distinct algorithms: value fusion (energies), ODF (original decisions), and LVDF. Their detection performance was compared extensively in the previous chapter (also in [54]), and thus not included in this study. However, it is important to put all the algorithms on an equal footing when comparing their localization performance, since they all have different detection rates. To accomplish this, 500 replications of the noise were generated, and only those cases where all three procedures were able to detect the target were kept for further calculations. Table 4.1 shows how many cases were detected by each method, and how many were detected by all in each case. It can be seen that

LVDF exhibits a superior detection performance overall, particularly for low SNRs.

SNR	VF	ODF	LVDF	all	VF	ODF	LVDF	all
	Model M1				Model M2			
	Grid deployment							
2	240	181	378	115	435	340	474	315
5	481	426	500	416	500	499	500	499
10	500	492	500	492	500	500	500	500
	Random deployment							
2	218	163	378	100	418	306	433	259
5	460	369	500	352	499	500	500	499
10	500	491	500	491	500	500	500	500

Table 4.1: Number of times (out of 500) the target is detected by different methods (VF, ODF, LVDF) and all methods together (all), with $\eta = 0.1$.

4.3.2 Localization and Signal Estimation Accuracy

The accuracy of the various algorithms for target localization is given in Table 4.2. The results are obtained for a low (2), medium (5) and high (10) SNR scenarios, with η set to 0.1. It can be seen that the LVDF localization algorithms clearly outperform their ODF counterparts for both models, with the exception of model M2 at SNR=10, where the EM version of ODF performs slightly better than the corresponding LVDF algorithm. Further, in the low SNR regime they clearly outperform the “gold standard” ML(E), while for the medium and high SNR regimes they exhibit a competitive performance. The HEM algorithms tend to be the most accurate, followed by EM, while both ML and HML tend to be less accurate. All algorithms do somewhat better on model M2, since the slower signal decay allows more sensors to pick up the target. It is also worth noting that for the ODF based algorithms, the EM version significantly outperforms the one based on numerical optimization. The poor performance of ML using Y , particularly at low SNR, is primarily due to the sensitivity of the numerical solver to the selection of initial values, which in the case of the adjusted decisions is not an issue due to the de-noising nature of LVDF

(see discussion in 4.3.3). As expected, for larger values of SNR the accuracy of all the algorithms improves, and for random deployments the pattern remains the same but all methods are somewhat less accurate.

SNR	ML(E)	ML(Y)	EM(Y)	HML(Y)	HEM(Y)	ML(Z)	EM(Z)	HML(Z)	HEM(Z)
Model M1, grid deployment									
2	0.208	0.576	0.223	0.329	0.236	0.077	0.056	0.075	0.089
5	0.011	0.502	0.113	0.275	0.066	0.019	0.020	0.012	0.012
10	0.005	0.421	0.028	0.221	0.006	0.019	0.016	0.010	0.006
Model M2, grid deployment									
2	0.164	0.388	0.119	0.226	0.111	0.066	0.050	0.093	0.049
5	0.011	0.101	0.018	0.031	0.012	0.022	0.021	0.012	0.012
10	0.005	0.064	0.017	0.021	0.005	0.020	0.020	0.006	0.005
Model M1, random deployment									
2	0.230	0.744	0.346	0.424	0.326	0.128	0.077	0.221	0.082
5	0.012	0.621	0.233	0.300	0.171	0.052	0.037	0.095	0.020
10	0.006	0.488	0.078	0.283	0.020	0.043	0.024	0.090	0.008
Model M2, random deployment									
2	0.191	0.625	0.197	0.329	0.187	0.092	0.052	0.158	0.052
5	0.012	0.170	0.040	0.216	0.023	0.049	0.031	0.106	0.017
10	0.007	0.096	0.016	0.109	0.006	0.025	0.021	0.043	0.006

Table 4.2: Average distance from the true location v as a function of SNR with $\eta = 0.1$.

In Table 4.3, the quality of the estimates of the signal S_0 is given in terms of root mean squared error (RMSE). Once again, the LVDF based algorithms outperform their ODF counterparts in almost all situations, and for low SNR they exhibit smaller RMSE than the gold standard. The only exception again is model M2 at SNR=10 with the EM algorithm. Finally, for larger values of SNR all the algorithms become more accurate, with the largest improvement for the gold standard. For both ODF and LVDF, the EM algorithm outperforms the corresponding hybrid EM version estimates for low SNR. For medium and high SNR, the HEM(Z) algorithm exhibits the best performance in terms of accuracy of localization and signal estimation (essentially the same as the "gold standard") (Table 4.2).

We compare next the accuracy of target localization for the various algorithms as a function of effective target size η for a low SNR=2 regime and a moderate one with SNR=5 (see Figures 4.1). The HML(Z) and HML(Y) algorithms are omitted from

SNR	ML(E)	ML(Y)	EM(Y)	HML(Y)	HEM(Y)	ML(Z)	EM(Z)	HML(Z)	HEM(Z)
Model M1, grid deployment									
2	0.923	1.297	0.933	1.068	0.736	0.634	0.718	1.017	0.537
5	0.163	1.555	0.923	0.722	0.450	0.558	0.521	0.223	0.169
10	0.077	1.702	0.546	0.861	0.083	0.661	0.460	0.355	0.082
Model M2, grid deployment									
2	0.798	1.104	0.717	1.798	0.747	0.475	0.464	1.491	0.487
5	0.120	0.733	0.314	0.446	0.183	0.415	0.345	0.354	0.124
10	0.060	0.623	0.267	0.262	0.082	0.325	0.325	0.119	0.065
Model M1, random deployment									
2	0.924	1.377	1.016	1.317	0.703	0.729	0.608	1.134	0.718
5	0.197	1.599	1.218	0.663	0.782	0.773	0.734	0.454	0.233
10	0.092	1.599	0.825	0.916	0.308	0.702	0.651	0.573	0.115
Model M2, random deployment									
2	0.796	1.328	0.868	2.449	0.877	0.423	0.518	2.003	0.465
5	0.151	0.855	0.406	0.521	0.276	0.544	0.489	0.458	0.160
10	0.076	0.686	0.234	0.531	0.097	0.314	0.264	0.377	0.079

Table 4.3: Root Mean Squared Error of estimating S_0 as a function of SNR with $\eta = 0.1$.

the figures to reduce clutter since they are outperformed by HEM(Z) and HEM(Y), respectively, in every case. In general, larger η corresponds to a bigger target which is easier to detect and localize. The patterns for both models are very similar, and the LVDF algorithms outperform the ODF ones; moreover, in a very noisy environment (SNR=2) the LVDF algorithms also outperform the energy based MLE, while for the case of SNR=5 they perform equally well, especially for larger targets.

The plots in Figure 4.2 assess the quality of the estimates for S_0 as a function of the target size. One can see that for larger targets the LVDF based algorithms perform well for both models, and somewhat outperform the gold standard at SNR = 2. For smaller size targets, the pattern is not that clear, although all the decision based algorithms exhibit similar performance. As before, the HEM versions tend to be the most accurate ones for both algorithms.

4.3.3 Starting Values

All the decision based algorithms are iterative in nature and require starting values for the parameters of interest. Experience shows that an inferior choice of starting values can slow down convergence and/or lead to poor quality estimates; in

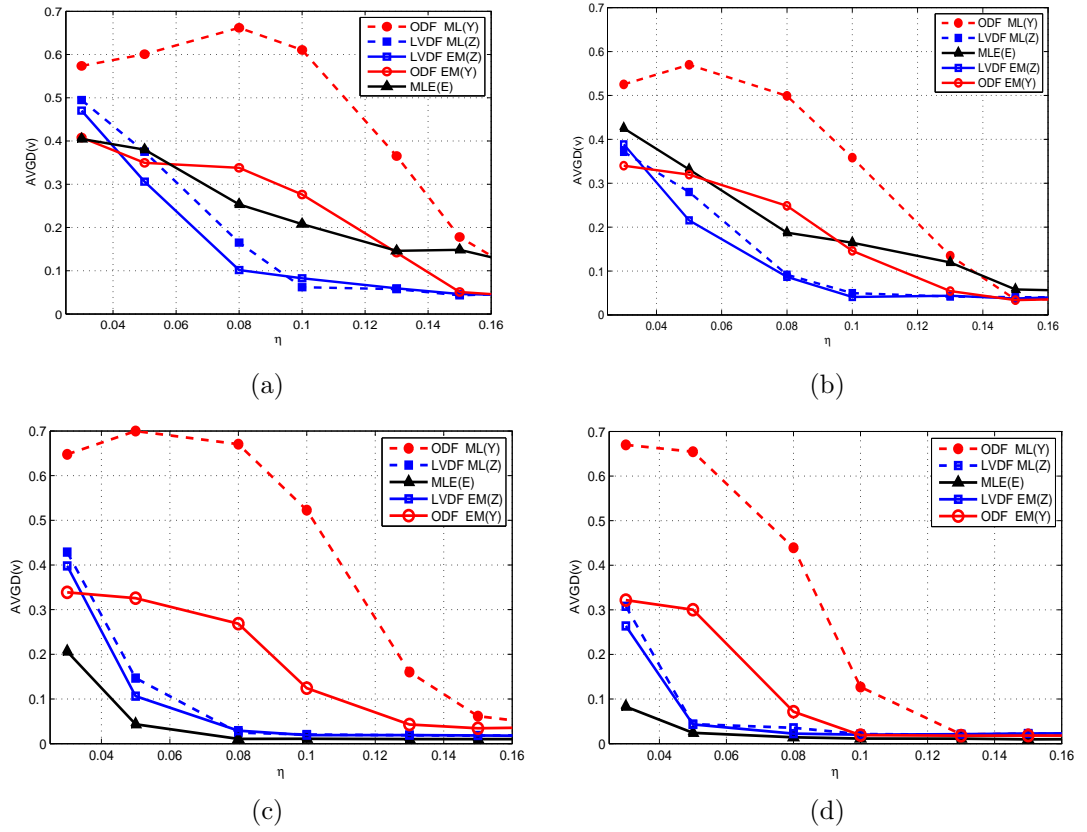


Figure 4.1: Average distance from true target location v as a function of η for square grid deployment. (a) SNR = 2, model M1; (b) SNR = 2, model M2; (c) SNR = 5, model M1; (d) SNR = 5, model M2.

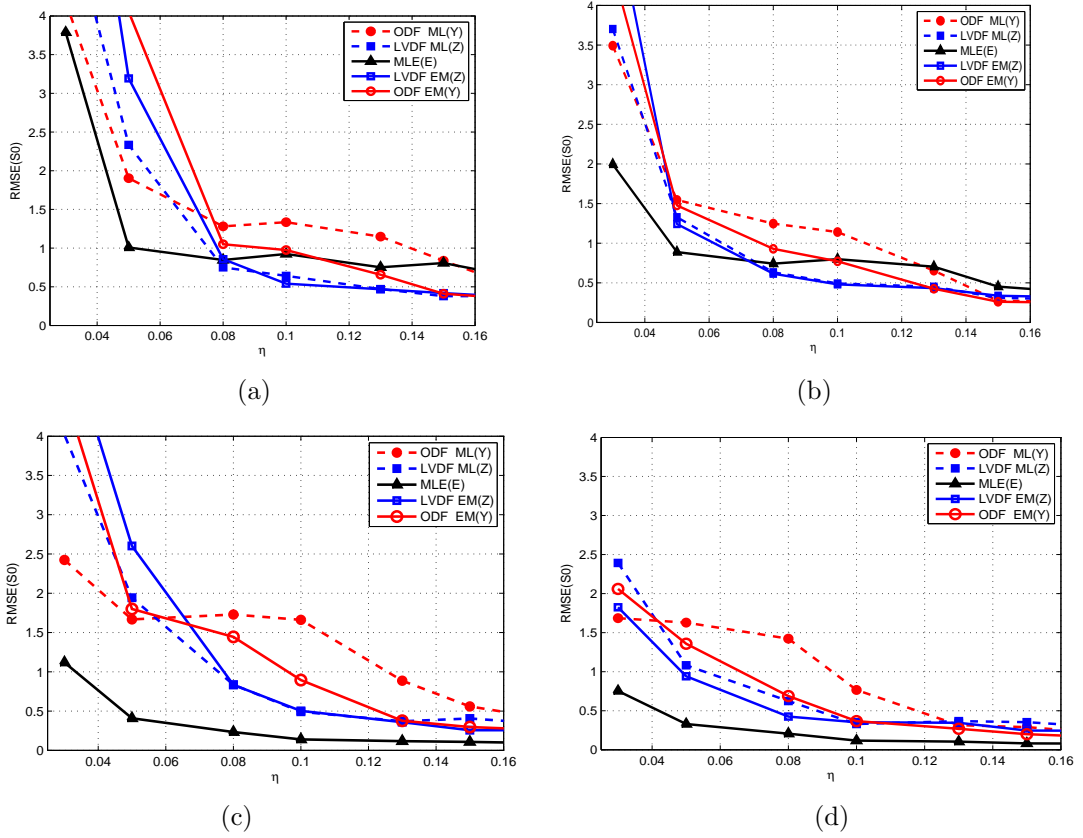


Figure 4.2: Root Mean Squared Error of estimating S_0 as a function of η for square grid deployment. (a) SNR = 2, model M1; (b) SNR = 2, model M2; (c) SNR = 5, model M1; (d) SNR = 5, model M2.

fact, the poor performance of localization based on Y is primarily due to this issue. Notice that the starting values have to be a function of the information available for the method, and a good initial guess for the target's location is the centroid of the positive decisions, given by

$$v_0(Y) = \frac{\sum_i s_i I(Y_i = 1)}{\sum_i I(Y_i = 1)}$$

for ODF and

$$v_0(Z) = \frac{\sum_i s_i I(Z_i = 1)}{\sum_i I(Z_i = 1)}$$

for LVDF. Because LVDF eliminates many distant false positives, $v_0(Z)$ tends to be significantly more accurate than $v_0(Y)$. For the benchmark ML(E), where all energies are available, a natural choice of starting value is the location $v_0(E)$ of the maximum energy reading $\max_i E_i$. Table 4.4 gives average distance from the starting location for each class of methods (energies, original decisions, and LVDF decisions) to the truth. It can be seen that all methods improve at higher SNR, but the starting value for Y is, on average, much further from the truth than the starting value for Z ; for energies, the starting value based on maximum energy works well at higher SNRs, but not at SNR=2.

SNR	$v_0(E)$	$v_0(Y)$	$v_0(Z)$	$v_0(E)$	$v_0(Y)$	$v_0(Z)$
	Model M1			Model M2		
	Grid deployment					
2	0.244	0.306	0.094	0.214	0.282	0.086
5	0.033	0.266	0.033	0.039	0.220	0.030
10	0.024	0.231	0.022	0.031	0.183	0.026
	Random deployment					
2	0.252	0.321	0.104	0.170	0.305	0.107
5	0.027	0.285	0.070	0.029	0.236	0.057
10	0.022	0.245	0.048	0.024	0.196	0.031

Table 4.4: Average distance of starting values from the true location as a function of SNR with $\eta = 0.1$.

If better starting values are available from some prior information or external

knowledge, performance of all methods will improve, but particularly that of Y . We found that if the search is initialized very close to the truth, the $\text{ML}(\text{E})$ generally provides the best localization, as expected, followed closely by Y and Z . However, the starting values we use are integral to the methods and are unlikely to be improved upon without extra information.

4.3.4 Robustness to Model Misspecification

The performance of all algorithms may change when aspects of the true model are misspecified. Of particular interest is the change in performance of the best algorithms, and changes in relative performance of the different algorithms. First, we test robustness to signal model misspecification by investigating how the quality of the estimates is affected when the signal is generated by model M2, but the assumed signal model is M1. The performance of the algorithms relative to each other remains exactly the same (results not shown, but see the pattern in Figures 4.1 and 4.2): LVDF is still the most accurate for low SNRs followed by $\text{ML}(\text{E})$ and ODF, and $\text{ML}(\text{E})$ is the best for higher SNRs, closely followed by LVDF. To assess changes in absolute accuracy of each algorithm, the differences between the error under misspecified model and the error under the true model are shown in Figure 4.3, for $\text{SNR} = 5$. The two binary likelihood methods ($\text{ML}(\text{Y})$ and $\text{ML}(\text{Z})$) are omitted as the least accurate in their class. Note also that we plot absolute rather than relative differences because some of the errors under the true model are very small, and will lead to unstable ratios. One can see that for target localization, the performance of both $\text{ML}(\text{E})$ and LVDF is very robust, whereas ODF performs somewhat worse, though the differences are small. For signal estimation, methods that use at least some energy information ($\text{ML}(\text{E})$, $\text{HEM}(\text{Y})$, and $\text{HEM}(\text{Z})$) do well, whereas both methods based on binary decisions only ($\text{EM}(\text{Y})$ and $\text{EM}(\text{Z})$) deteriorate more. These

differences may be larger for more drastically different models.

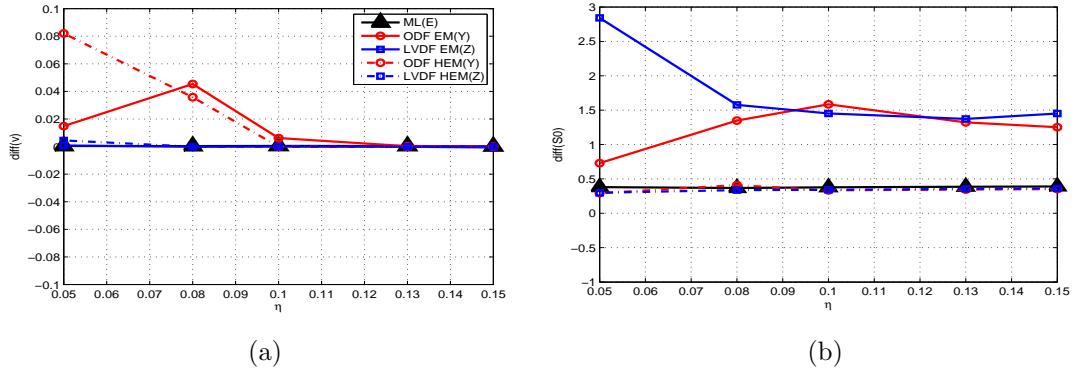


Figure 4.3: True model M2 misspecified as M1 with SNR=5. (a) The difference between average distances from true v for misspecified and true models; (b) The difference between RMSE of \hat{S}_0 for misspecified and true models.

We next look at another type of potential misspecification, which is when the noise distribution is misspecified. In the simulation, M1 generates the signal and is used by the algorithms, but the noise comes from a t -distribution with 3 degrees of freedom, while Gaussian distribution is assumed by the algorithms. Note that this t -distribution has significantly heavier tails than the Gaussian, but the false alarm rate of the individual sensor is fixed at the same value $\gamma = 0.1$, and the two distributions are scaled to have the same variance. In this case, both versions of the LVDF algorithm perform very well and prove the most robust. The ODF errors also remain similar, with somewhat erratic behavior for smaller targets where ODF may perform better under a misspecified noise model. The energy-based MLE is the most sensitive to distribution misspecification, as one would expect.

4.3.5 Confidence Region Estimation

The coverage and area of bootstrap confidence regions for the target location v are summarized in Table 4.5. The setting was a target located at $v = (1/4, 1/4)$, with sensor false alarm rate $\gamma = 0.1$ and target size determined by $\eta = 0.1$. The table gives

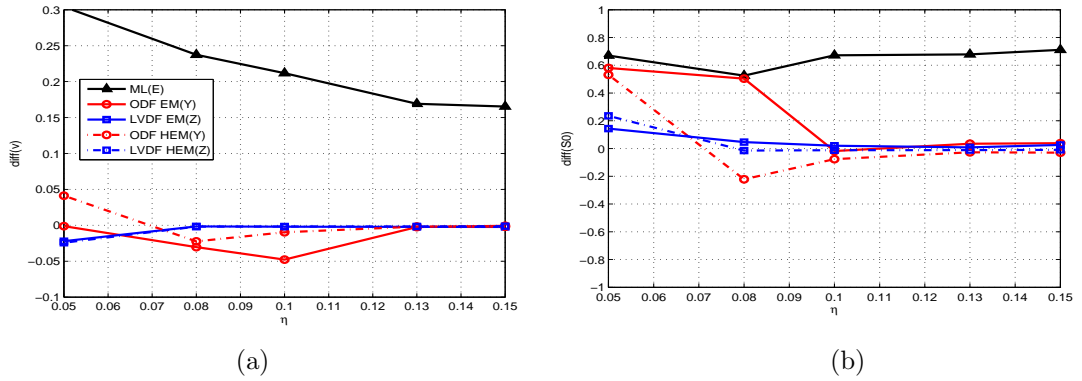


Figure 4.4: True noise distribution t_3 misspecified as Gaussian, with SNR=5. (a) The difference between average distances from true v for misspecified and true models; (b) The difference between RMSE of \hat{S}_0 for misspecified and true models.

the coverage for the various methods for models M1 and M2. It can be seen that the energy based ML algorithm, together with all variants of the LVDF algorithm except HML, approximately achieve the nominal level of 95% for all SNRs examined, whereas the ODF ones fall short, particularly for the more localized target of model M1. The table also gives the average area of the confidence regions. For SNR=2, the gold standard is outperformed by all of LVDF algorithms in all cases, and in some cases by some of the ODF methods as well. For larger SNRs, ML(E) is the best. The HML(Z) method gives small confidence regions, but taken coverage probabilities into account, we conclude that of decision-based methods, EM(Z) produces the most reliable confidence regions, with HEM(Z) also showing good performance. We note that the bootstrap procedure does require extra computing time, but it typically takes less than a minute to compute a confidence region on an ordinary PC.

4.3.6 Computational Costs

Some computational cost estimates are shown in Figure 4.5, which give the distribution of the number of iterations over 100 replications of the ML (direct optimization), EM, and HEM algorithms. One iteration for each method takes approximately

SNR	ML(E)	ML(Y)	EM(Y)	HML(Y)	HEM(Y)	ML(Z)	EM(Z)	HML(Z)	HEM(Z)
Model M1, grid deployment									
2	1.0000 96	2.3195 76	0.2692 47	0.6120 62	0.1977 49	0.3041 94	0.3204 97	0.0415 77	0.2411 90
5	0.0029 93	2.1307 72	0.1614 78	0.7897 84	0.0510 96	0.0491 97	0.0070 96	0.0212 98	0.0019 95
10	0.0003 93	2.1063 87	0.0497 97	0.6587 82	0.0010 98	0.0089 91	0.0037 96	0.0459 99	0.0004 98
Model M2, grid deployment									
2	0.8738 97	1.9541 81	0.1667 62	0.0583 54	0.0365 73	0.3745 97	0.1768 95	0.0074 46	0.1506 95
5	0.0016 97	0.8758 90	0.0106 91	0.0202 92	0.0014 87	0.0086 91	0.0066 94	0.0013 88	0.0015 90
10	0.0003 93	0.3162 99	0.0031 88	0.0061 96	0.0003 94	0.0134 87	0.0049 94	0.0003 88	0.0003 96
Model M1, random deployment									
2	1.0294 92	2.0788 61	0.2313 22	0.5898 50	0.1495 28	0.7190 95	0.3516 91	0.2851 55	0.2850 96
5	0.0027 94	2.0278 65	0.2754 50	0.6529 63	0.1773 66	0.1928 93	0.0294 77	0.2949 82	0.0159 91
10	0.0005 94	1.9758 78	0.1605 85	0.6260 59	0.0401 97	0.0735 91	0.0077 91	0.2707 85	0.0007 95
Model M2, random deployment									
2	0.8657 95	1.8346 55	0.2374 59	0.1478 26	0.0653 58	0.6836 96	0.2319 94	0.1445 63	0.2329 97
5	0.0019 92	1.0153 91	0.0378 93	0.2354 55	0.0024 86	0.0828 76	0.0128 90	0.1850 81	0.0027 91
10	0.0004 91	0.6465 97	0.0031 93	0.1237 78	0.0004 94	0.0482 91	0.0053 95	0.1035 92	0.0005 97

Table 4.5: Average area (top entry) and number (bottom entry) of 95% confidence regions (both out of 100 replications) containing the true target location.

the same amount of time, so comparing the number of iterations directly provides a reasonable estimate of relative computational costs. Comparisons are shown for a grid deployment with $\eta = 0.1$ and $\text{SNR} = 5$. Note that the EM algorithms were stopped at 300 iterations if they did not achieve convergence using a 10^{-4} tolerance. On average, the LVDF algorithms converge faster than their ODF counterparts; however, it takes the optimization about 1/10 of the iterations to converge on average, compared to the EM versions (recall that the M-step requires a numerical optimization; the number of iterations shown for EM is the sum of the optimization iterations at each M-step and the EM iterations). Given the significantly higher accuracy of the EM algorithms, this represents the usual trade-off between computational complexity and accuracy. The hybrid EM algorithms converge faster than their EM counterparts as one would expect.

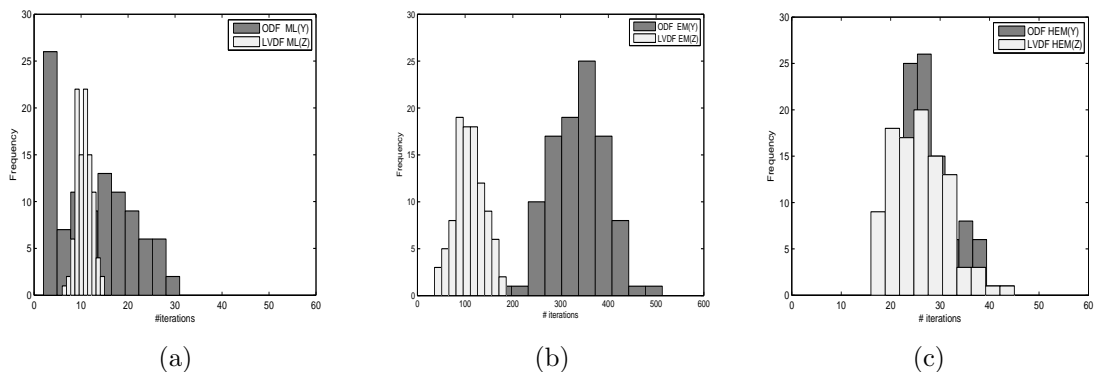


Figure 4.5: Distribution of iterations to convergence. (a) ML algorithms; (b) EM algorithms; (c) HEM algorithms.

4.4 Single Target Tracking

In this section, we examine the performance of the proposed localization algorithms for tracking a single target moving through the monitoring region R . The problem of tracking targets over time is itself a significantly harder problem which

we will address in the next chapter.

The setting we investigate here as follows. Sensors record energy readings at time slots $t = 1, 2, \dots$ and make decisions at each time slot. In our previous chapter, temporal decision fusion was introduced and shown to lead to significant improvements in terms of the detection probability for both stationary and moving targets. Temporal decision fusion combines decisions over time using an exponentially weighted moving average scheme. The same idea can be extended to fuse information about the location of the target over time. Specifically, we have $\tilde{v}_0 = \hat{v}_0$, and

$$(4.23) \quad \tilde{v}_t = \lambda \hat{v}_t + (1 - \lambda) \tilde{v}_{t-1}, \quad t = 1, 2, \dots$$

where \hat{v}_t denotes the location estimate obtained from measurements at time t . A more detailed discussion on the use of exponentially weighted moving average and the choice of λ can be found in [54]. An analogous scheme can be used for the signals' magnitude. In addition, we use \tilde{v}_{t-1} as a starting value for localization at time t , instead of the centroid of positive decisions at time t .

The first scenario examined using temporal decision fusion is of a target moving from west to east through the middle of the monitoring region $R = [0, 1]^2$. A 20×20 WSN is deployed on a grid and the target's speed is set to one hop per time slot; hence, it takes the target 20 time periods to cross R . The remaining parameters are set to $S_0 = 2$, $\lambda = 0.5$ and $\eta = 0.1$. The trajectories estimated by the various algorithms for a moderately noisy environment with SNR=2 under model M1 are shown in Figure 4.6, along with average error as a function of time. As expected from the previous results, the LVDF algorithms track the target particularly well, whereas the ODF one exhibit the worst performance. It is worth noting that ML(E) suffers more from errors at the beginning of the monitoring period. For larger SNRs (results not shown), ML(E) catches up and eventually outperforms LVDF, whereas

the ODF algorithms continue to exhibit an inferior performance. Similar findings (not shown) have been obtained for estimating the (constant) signal of the moving target, as well as for random sensor deployments.

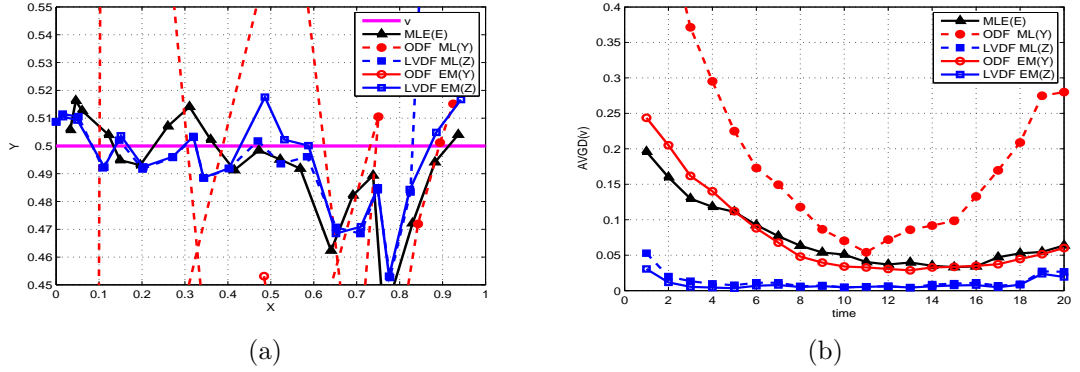


Figure 4.6: Tracking a moving target: (a) Estimated target trajectory for a single realization; (b) Average distance from the true trajectory (over 100 replications).

In the second scenario, the target is positioned at $v = (0.25, 0.25)$ and does not move. However, the signal's amplitude evolves over time according to the model $S_0(t) = 2/(1 + 0.01(t - 10)^2)$, with $t = 1, 2, \dots, 20$, and $\sigma = 0.4$ held constant, which results in SNR going from 2.5 at time 0 to 5 at time 10 and down to 2.5 again by time 20. Other settings are the same as in the first scenario (grid deployment, $\eta = 0.1$, signal model M1, $\lambda = 0.5$). The performance of the algorithms is summarized in Figure 4.7. Once again, the LVDF gives the best results at all points in time, followed by the gold standard and the ODF algorithms. At higher SNRs, the ML(E) algorithm becomes the best method, a result consistent with previous ones.

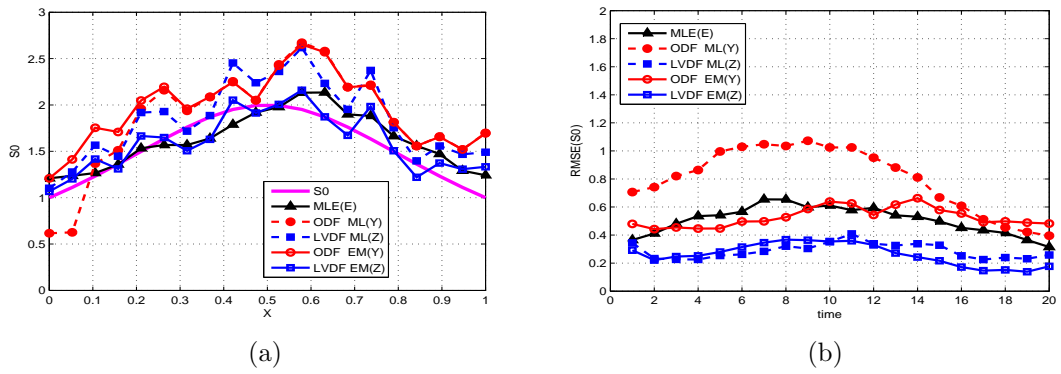


Figure 4.7: Tracking a stationary target with evolving signal: (a) Estimated signal amplitude for a single realization; (b) Average RMSE of temporally fused S_0 (over 100 replications).

CHAPTER V

Localization and Tracking of Multiple Targets with Wireless Sensor Networks

5.1 Introduction

In this chapter we introduce a framework for tracking multiple targets over time using binary decisions collected by a wireless sensor network, and apply the methodology to two case studies – an experiment involving tracking people and a project tracking zebras. As motivation, we start by describing these two case studies.

Our first case study is the Network Embedded Systems Technology (NEST) project developed at the University of California at Berkeley [17, 86]. The purpose of the NEST project was to develop both the necessary hardware and software platforms for a WSN to track targets traversing a monitored area. A prototype sensor system was deployed, where each sensor had an 8-meter sensing radius and a 10% false alarm probability (i.e., the probability of detecting the presence of a target, when none was present). In a particular testing experiment whose goal was to track one, two or three people crossing the monitored area, 144 sensors were used on a 12×12 grid pattern, spaced 5 meters apart. The sensors reported their decisions on the presence or absence of target(s) to one central node, called the *fusion center* in the WSN literature, which is responsible for making the final decision. All positive

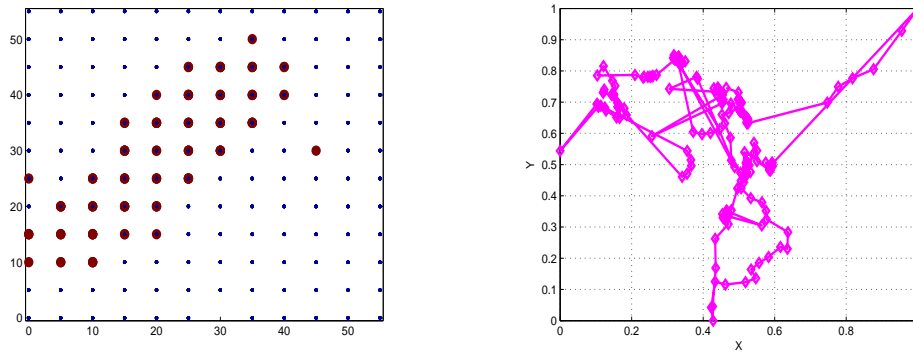


Figure 5.1: Left panel: The activation pattern of NEST sensors by a person traversing the monitored area. Right panel: The trajectory of a single zebra in the monitored area.

decisions from a trial involving a single person traversing the monitored area is shown in the left panel of Figure 5.1. Note that different positive decisions were made at different times. The goal is to infer the path of the target(s) through the monitored area, which requires detecting the presence and estimating the number of targets, locating them at a particular point in time, and tracking their positions over time.

Our second application involves data collected by the ZebraNet project¹, which tracked a zebra herd in its natural habitat at Sweetwaters Game Reserve near Nanyuki, Kenya [71, 126, 120]. In the initial deployment of ZebraNet, sensors with GPS capabilities were attached to the zebras rather than placed in the field. The trajectory of one zebra over the course of about a day is shown in the right panel of Figure 5.1. This example illustrates that in order to perform well in realistic environments, the tracking algorithms need to be able to handle complicated trajectories with abrupt changes in direction.

In previous chapters, we have already discussed a general structure of WSNs and the associated literature for detection and localization of a single target. Several results have been proposed in the literature to address the problem of localizing

¹ZebraNet is an inter-disciplinary joint animal monitoring study of the Departments of Electrical Engineering, Ecology and Evolutionary Biology, and Computer Science at Princeton University (<http://www.princeton.edu/~mrm/zebranet.html>).

multiple targets. For example, in [99] energy measurements emanated from multiple targets are used in a likelihood based approach to estimate their position, whereas in [77] a weighted least squares algorithm for localization, which achieves results similar to maximum likelihood but at a smaller computational cost. In both cases, the number of targets is assumed known. However, when the number of targets present in the field is unknown, the localization of multiple targets becomes more challenging.

Tracking of multiple targets over time is even harder problem, because a good algorithm needs to account for multiple targets moving in complicated, possibly intersecting patterns, for the appearance/disappearance of targets in the monitored area over time, and also, in some applications, results must be produced in real time (for a recent review see [70]). Some authors focus on tracking a single target [76, 51, 119], whereas others consider multiple targets [41, 112]. State space models are an obvious tool for tracking purposes and particle filtering algorithms have been proposed in a number of papers [67, 73, 58, 111, 112, 70]. Their main drawback is their computational complexity, which only reduces in the case of a linear Gaussian model due to the availability of the Kalman filter solution. In some of these models, restrictive assumptions are required, such as that one target can generate at most one sensor measurement during a time period, or that a sensor can receive a signal from one target only. In some applications, saving energy is an overarching consideration for the WSN operators and hence activation/deactivation schedules for sensors becomes important. A number of papers have addressed this issue within the context of target tracking [116, 18, 125, 76, 19, 107]. Finally, a recent development involving mobile fusion centers that follow the target has been studied in [18] and [76].

Our focus here is on tracking from binary decisions (in the NEST project nothing

else is available). Multi-target tracking techniques based on binary sensor readings have been examined in [86] and [119]. In particular, [86] developed a fusion algorithm which first converts binary detections into finer position estimates using spatial correlations, and then uses an MCMC algorithm to track the targets. This algorithm does not require prior information about the number of targets or their labels. The algorithm has been tested to track human subjects moving through an outdoor field in real-time mode [17]. [119] developed the virtual measurement (VM) approach, which aims to define a mapping between the space of binary sensor decisions and the space of target states, which is similar in spirit to an expectation-maximization algorithm (EM). This technique is less computationally intensive than particle filtering, but requires the assumption of no sensor failures and no false detections.

The goal of this work is to build on the target localization methodology developed in Chapter IV (see also [55]) to address the multi-target tracking problem in the context of our motivating applications. To achieve this goal, we extend our likelihood based framework for localization of a single target to localizing multiple targets and to tracking over time, and incorporate information about the targets' speed and acceleration. The developed framework appropriately handles multiple targets that appear or disappear over time, and does not assume any prior information on the number of targets. Finally, although the focus is on binary measurements, the proposed framework with minor adjustments can handle continuous measurements or a mixture of binary and continuous measurements, as discussed in Section 5.3.4.

The remainder of the chapter is organized as follows. Section 5.2 introduces the general setup of the multi-target localization and tracking problems and describes signal models and data types. Section 5.3 presents the theoretical basis for our localization algorithms, and Section 5.4 presents our approach to tracking target locations

over time. Section 5.5 provides a brief performance evaluation of the proposed algorithms via simulations, and Section 5.6 applies the algorithms to data from the NEST and ZebraNet projects.

5.2 Problem Formulation

Again consider a WSN comprised of N identical sensors deployed at locations s_i , $i = 1, 2, \dots, N$ over a two-dimensional monitoring region R . Suppose that p targets move in R over time, assuming positions $x_j(t)$, $j = 1, \dots, p$ at times $t = t_1, t_2, \dots$. Each target j emits a signal (e.g. infrared/temperature/acoustical) of strength $S_0^{(j)}(t)$ at the target location. The signal attenuates with distance from the target according to a decreasing function C , and thus a sensor located at s_i receives signal from target j given by

$$S_i^{(j)}(t) = S_0^{(j)}(t)C_{\eta_j}(\delta_i(x_j(t))),$$

where $\delta_i(x_j(t)) = \|s_i - x_j(t)\|$ is the distance from the target to sensor i , and η_j is a potentially time varying scaling parameter which represents the effective target size.

At time t , each sensor obtains an *energy* reading comprised of all p individual signals and corrupted by random noise:

$$(5.1) \quad E_i(t) = \sum_{j=1}^p S_i^{(j)}(t) + \epsilon_i(t), \quad i = 1, \dots, N,$$

where errors $\epsilon_i(t)$ are assumed to be independent in time and space with mean zero and variance $\sigma^2(t)$. The collected energy readings $E_i(t)$ are either directly transmitted to the fusion center or converted to binary decisions $Y_i(t) = I(E_i(t) \geq \tau_i)$, using a pre-specified threshold τ_i , which is related to the individual sensor's false alarm probability. Here we assume that all sensors are identical and $\tau_i \equiv \tau$.

In most circumstances, location estimates based on energy readings (*value fusion*) tend to be significantly more accurate, while those based on binary decisions (*decision*

fusion) are more power-efficient. The Local Vote Decision Fusion (LVDF) algorithm introduced in Chapter III significantly improves the accuracy of decision fusion, while sacrificing little in terms of energy consumption. It has also been shown that location estimates for a single target based on LVDF are in general competitive and outperform even value fusion in low signal-to-noise environments.

In tracking applications, our ultimate goal is to estimate the following parameter vector,

(5.2)

$$\theta(t) = (p(t), x_1(t), \dots, x_p(t), S_0^{(1)}(t), \dots, S_0^{(p)}(t), \eta_1(t), \dots, \eta_p(t), \sigma(t)), \quad t = 1, 2, \dots$$

based either on energy readings ($E_i(t)$) or binary decisions ($Y_i(t)$ or corrected $Z_i(t)$). Note that we assume that the sensor measurements are obtained on a *synchronized* schedule, at discrete points in time, which is usually the case in practice.

5.3 Methods and Algorithms

The necessary building block for a tracking algorithm is localization of targets at a given point in time. In this section, we generalize the algorithms that estimate target locations at a given point in time based on energy/binary data developed in a previous chapter from the case of one target to the case of multiple targets. To ease the presentation, we suppress the dependence on t in this section, but in general all parameter values and data depend on t . First, we discuss localization with the number of targets assumed known, and present a method for selecting the number of targets in 5.3.5.

5.3.1 Localization from Energy Readings

In the presence of Gaussian mean zero, variance σ^2 background noise, the log-likelihood of the energies at a fixed point in time is given by

$$(5.3) \quad \ell_E(\theta) = -\frac{N}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^N \left[E_i - \sum_{j=1}^p S_0^{(j)} C_{\eta_j}(\delta_i(x_j)) \right]^2.$$

With other parameters fixed, maximizing over $S_0^{(j)}$ is equivalent to solving the following system of linear equations:

$$(5.4) \quad \sum_{l=1}^p S_0^{(l)} \sum_{i=1}^N C_{\eta_l}(\delta_i(x_l)) C_{\eta_j}(\delta_i(x_j)) = \sum_{i=1}^N E_i C_{\eta_j}(\delta_i(x_j)), \quad j = 1 \dots p.$$

Maximizing with respect to the noise variance σ^2 can be done in closed form, if all other parameters are fixed, as

$$(5.5) \quad \sigma^2 = \frac{1}{N} \sum_{i=1}^N \left[E_i - \sum_{j=1}^p S_0^{(j)} C_{\eta_j}(\delta_i(x_j)) \right]^2.$$

We obtain the remaining parameters by numerical maximization of (5.3), with (5.4) and (5.5) plugged in.

5.3.2 Localization from Binary Decisions

Notice that the initial decisions $Y_i \in \{0, 1\}$, are independent Bernoulli random variables with probability of success given by $\mathbb{P}(Y_i = 1) \equiv \alpha_i(\theta) = 1 - F(A_i(\theta))$, where $F(\cdot)$ is the cumulative distribution function of ϵ_i/σ (not necessarily Gaussian) and $A_i(\theta)$ represents the standardized excess energy level given by

$$A_i(\theta) = \frac{\tau - \sum_{j=1}^p S_0^{(j)} C_{\eta_j}(\delta_i(x_j))}{\sigma}.$$

The log-likelihood function of $\{Y_i\}$ can be computed as

$$(5.6) \quad \ell_Y(\theta) = \sum_{i=1}^N [Y_i \log \alpha_i(\theta) + (1 - Y_i) \log(1 - \alpha_i(\theta))] .$$

Since there is no closed form solution for any of the parameters, either direct numerical maximization of the log-likelihood function (5.6) should be used, or an EM algorithm. We have explored both options both for localization and for tracking; the EM algorithms are often more accurate but are also substantially computationally more expensive. The EM algorithm proceeds as follows: under the Gaussian assumption on the noise, the M-step is defined by equations (5.3), (5.4), and (5.5).

The E-step equations are given by

$$(5.7) \quad \mathbb{E}[E_i|\vec{Y}] = \hat{E}_i = \sum_{j=1}^p S_0^{(j)} C_{\eta_j}(\delta_i(x_j)) + \frac{\sigma}{\sqrt{2\pi}} \exp\left(-\frac{A_i(\theta)^2}{2}\right) B_i(\theta, Y_i) ,$$

$$(5.8) \quad \mathbb{E}[E_i^2|\vec{Y}] = \hat{E}_i^2 = \sum_{j=1}^p S_0^{(j)} C_{\eta_j}(\delta_i(x_j))(\tau - \hat{E}_i) - \hat{E}_i\tau ,$$

where

$$B_i(\theta, Y_i) = \frac{Y_i - 1}{\Phi\left(-\frac{A_i(\theta)^2}{2}\right)} + \frac{Y_i}{1 - \Phi\left(-\frac{A_i(\theta)^2}{2}\right)} .$$

As usual, the E-step and the M-step are alternated until convergence.

5.3.3 Localization from LVDF Decisions

Since corrected decisions Z_i are not independent, we employ a *pseudo-likelihood* estimation approach [10]. In order to simplify calculations, it is further assumed that the success probabilities of initial decisions are approximately the same within the neighborhood, i.e. for $j \in U(i)$, $\mathbb{P}(Y_j = 1) = \mathbb{P}(Y_i = 1)$. The pseudo-loglikelihood function at a fixed time point for the corrected decisions Z_i is given by

$$(5.9) \quad \ell_Z(\theta) = \sum_{i=1}^N [Z_i \log \beta_i(\theta) + (1 - Z_i) \log(1 - \beta_i(\theta))] ,$$

where $\beta_i(\theta) = \mathbb{P}(Z_i = 1)$ is approximated by

$$\beta_i(\theta) \approx \sum_{k=[M/2]}^M \binom{M}{k} \alpha_i(\theta)^k (1 - \alpha_i(\theta))^{M-k} .$$

Again, we can either maximize the likelihood directly or apply the EM algorithm. The M-step is the same as that used for the initial decisions Y_i . The E-step requires calculating the first and second conditional moments $\mathbb{E}[E_i|\vec{Z}]$, $\mathbb{E}[E_i^2|\vec{Z}]$, which can be written as $\mathbb{E}[E_i|\vec{Z}] = \sum_{k=0,1} \mathbb{E}(E_i|Y_i = k)\mathbb{P}(Y_i = k|Z)$, and an analogous formula holds for the second moment. The moments conditional on Y were calculated in (5.7) and (5.8), and $\mathbb{P}(Y_i = k|\vec{Z})$ can be approximately computed using the Bayes rule. For example, for $k = 1$ we have

$$(5.10) \quad \mathbb{P}(Y_i = 1|\vec{Z}) = \alpha_i \prod_{j:i \in U(j)} \left(\frac{\tilde{\beta}_{ji}}{\beta_j} \right)^{Z_j} \left(\frac{1 - \tilde{\beta}_{ji}}{1 - \beta_j} \right)^{1 - Z_j},$$

where

$$(5.11) \quad \tilde{\beta}_{ji} = \mathbb{P}\left(\sum_{k \in U(j), k \neq i} Y_k \geq \frac{M}{2} - 1 \right) \approx \sum_{q=[M/2-1]}^{M-1} \binom{M-1}{q} \alpha_j^q (1 - \alpha_j)^{M-1-q}.$$

5.3.4 Hybrid Maximum Likelihood Estimates

In some situations, a mixture of energy readings and binary decisions may be transmitted to the fusion center. One natural protocol is to have sensors that make positive decisions transmit the energies, while sensors that make negative decisions only transmit their negative decisions (or, equivalently, nothing at all). This strategy leads to lower communication costs compared to transmitting all energies, and more accurate targets location estimates compared to methods based entirely on binary decisions. In this case, it is straightforward to extend the EM algorithm for binary decisions. We refer to this extension as hybrid expectation maximization (HEM) algorithm in Chapter IV. The only modification needed is to replace conditional moments for the available energies with the observed energy readings. It is also possible to directly maximize the likelihood of the observed data; this approach does not perform as well as the hybrid EM, so we do not pursue it here.

5.3.5 Estimating the Number of Targets

Once we generalized the localization algorithms from one to many targets, we have to resolve an additional problem of estimating the number of targets. We propose to use the Bayes Information Criterion (BIC), a common way to select a “model complexity” parameter such as the number of targets within a likelihood framework. To pick p , we maximize

$$(5.12) \quad \text{BIC} = -2\ell_p(\hat{\theta}) + 2(4p + 1) \log N,$$

where $4p + 1$ gives the total number of parameters that need to be estimated for a model with p targets (signal amplitude, two coordinates, and the attenuation parameter for each target, and the noise variance σ^2), and $\hat{\theta}$ is the maximum likelihood estimate of the parameters assuming p targets are present.

5.4 Multiple Target Tracking

There are many ways to build a complete model and algorithm for tracking, and the choice depends on the applications. Tracking can be performed in an offline or online fashion. In the former case, one collects all the measurements (energies or binary decisions) at all the time point t_1, \dots, t_{N_T} , and then estimates all the parameters at all time points simultaneously. If no assumptions are made on how things evolve over time, this results in a prohibitively large number of parameters and computational expense. More importantly, in many applications one wants to have up-to-date information instantly, which requires online estimation, namely estimating parameters at time t as soon as the data for that time point becomes available. Even though in principle one could use the new information to update estimates at earlier time points, in practice computational constraints typically lead one to simply maximize the likelihood for one current time point t_n . However,

one can use the information from previous time points, for example, the previous estimated location and signal of the target at time t_{n-1} could provide starting values for localization at time t_n ; other ways of incorporating information about the past are discussed below.

An important modeling consideration is whether to model a dependency structure in the noise over time. In our context, the noise is primarily receiver noise, so there is no reason to assume dependence. If the noise is likely to be dependent over time, appropriate time series models can be incorporated into our framework; this direction is outside the scope of the current paper. Throughout the paper, we assume that the noise is independent in time as well as in space.

Even though our framework in principle allows for estimating all the parameters separately, it is important to consider the application at hand and make reasonable assumptions about which parameters are likely to change over time and which are likely to remain constant. Incorporating the application context in this way will lead to both more accurate estimation and reduced computational cost. For example, consider the following three different cases for signal amplitude:

1. Signal amplitudes $S_0^{(j)}(t)$ are changing over time and are different for each target;
2. Signal amplitudes $S_0^{(j)}(t) \equiv S_0^{(j)}$ are constant over time, but different for each target;
3. Signal amplitudes $S_0^{(j)}(t) \equiv S_0$ are the same for all targets and constant over time.

Whether the signal changes over time depends a lot on the type of sensor used. The NEST project uses infrared sensors to detect humans, and thus we can reasonably assume that the signal amplitudes are the same for all targets and constant over

time. Similarly, we assume that the signal attenuation parameter η does not depend on time and is the same for all targets. Further, we assume the variance $\sigma^2(t) \equiv \sigma^2$ remains constant over time, since there is no reason to believe otherwise in our application.

In this case the formulas described in Section 5.3 are simplified, and in online tracking, current estimates of global parameters can be updated at every time step by incorporating new data. For example, the common signal amplitude can be updated at each time step t_n as follows:

$$(5.13) \quad \hat{S}_0(t_n) = \frac{\sum_{i=1}^N \sum_{k=1}^n E_i(t_k) \sum_{j=1}^p C_{\hat{\eta}_j}(\delta_i(\hat{x}_j(t_k)))}{\sum_{i=1}^N \sum_{k=1}^n \sum_{j=1}^p C_{\hat{\eta}_j}^2(\delta_i(\hat{x}_j(t_k)))}.$$

Estimates obtained via proposed methods can be affected by additional error due to a possible model misspecification, or high level of noise present in sensor measurements. One approach here would be to combine the results for target location and signal magnitude over time using an exponentially weighted moving average scheme (as discussed in Chapter IV), or smooth estimated data using, for example, splines as basis functions.

In many applications, targets follow fairly regular trajectories, which suggests that some kind of trajectory smoothing could be beneficial in estimating it. In order to incorporate information about the target's previous position and guarantee some degree of smoothness in the trajectory, we utilize a penalized likelihood approach, which in general can be written as

$$(5.14) \quad \ell(\{\theta(t), t \in [0, t_{n+1}]\}) - \sum_{j=1}^p \lambda_j \int_0^{t_{n+1}} [\ddot{x}_j(t)]^2 dt - \sum_{j=1}^p \rho_j \int_0^{t_{n+1}} [\ddot{S}_0^{(j)}(t)]^2 dt,$$

where $\ddot{x}_j(t)$ denotes the acceleration of the target and $\ddot{S}_0^{(j)}(t)$ the second derivative of the amplitude of the signal. The second term is not needed if we assume the signal remains constant over time.

Assuming independent errors over time, the log-likelihood of the observations can be decomposed into a sum. Thus, for online tracking, we approximate $\ell(\{\theta(t), t \in [0, t_{n+1}]\})$ with $\sum_{q=1}^n \ell(\hat{\theta}(t_q)) + \ell(\theta(t_{n+1}))$, and only use the last term. Similarly, if we approximate the integral in (5.14) by second-order differences, only the two previous time points affect the penalty terms that involve $\theta(t_{n+1})$. Thus, given the parameter estimates for times up to t_n , and assuming constant signal, we estimate parameters at time point t_{n+1} by maximizing

$$(5.15) \quad \ell(\tilde{\theta}(t_{n+1})) - \sum_{j=1}^p \lambda_j \left[\left(\frac{x_j^{(1)}(t_{n+1}) - \hat{x}_j^{(1)}(t_n)}{t_{n+1} - t_n} - \frac{\hat{x}_j^{(1)}(t_n) - \hat{x}_j^{(1)}(t_{n-1})}{t_n - t_{n-1}} \right)^2 + \left(\frac{x_j^{(2)}(t_{n+1}) - \hat{x}_j^{(2)}(t_n)}{t_{n+1} - t_n} - \frac{\hat{x}_j^{(2)}(t_n) - \hat{x}_j^{(2)}(t_{n-1})}{t_n - t_{n-1}} \right)^2 \right] \frac{(t_{n+1} - t_n)}{(t_n - t_{n-1})^2},$$

where we write $x = (x^{(1)}, x^{(2)})$ for the two planar coordinates of x , and, with our assumptions, $\tilde{\theta}(t_{n+1}) = (x_1(t), \dots, x_p(t), S_0, \eta, \sigma^2)$. The estimation procedure iterates over different parameters; we use estimates that average over time for the global parameters (such as (5.13) for S_0), and estimate the coordinates $x_j(t)$ for the current time point only with the global estimates plugged in.

Using the smoothing penalty introduces the additional challenge of choosing the smoothing parameters λ_j . At the same time, penalized maximum likelihood provides the user with flexibility to enforce smoothness on trajectory estimates as needed and to incorporate prior knowledge about the expected trajectories to improve estimation. For example, one may expect that a larger degree of smoothing would be helpful in the NEST project (people deliberately walking through a field) than in the ZebraNet project (zebras grazing in a natural environment). The choices of smoothing parameters for these datasets are discussed in Section 5.6. It may also be possible to develop appropriate criteria for automatic choice of λ_j , but that is beyond the scope of this paper.

Finally, to adapt to the situations when targets appear/disappear over time we propose the following algorithm. At each time point, the number of targets present is estimated by using the BIC as described in Section 5.3.5 in conjunction with penalized maximum likelihood. The locations $\hat{p}(t_{n+1})$ of the identified targets at t_{n+1} are matched to those estimated at the previous time slot ($\hat{p}(t_n)$). If $\hat{p}(t_{n+1}) < \hat{p}(t_n)$, tracking of unmatched targets is discontinued; otherwise, new targets start to be tracked.

5.5 Performance Evaluation

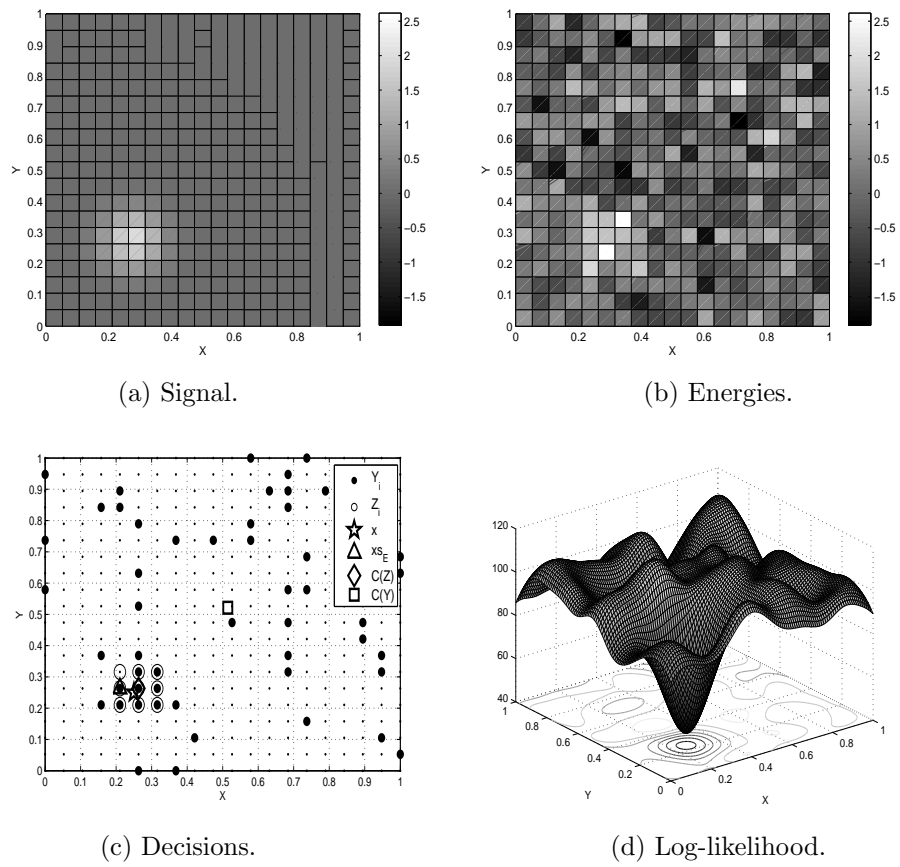
We briefly examine the performance of the proposed methods and algorithms on simulated data under several specific challenging scenarios that may occur in tracking multiple targets. The simulation setting is as follows: a wireless sensor network comprised of 400 sensors is deployed on a 20×20 grid over the unit square. The signal received by a sensor at location s_i from target j is modeled as $S_i^{(j)}(t) = S_0^{(j)}(t) \exp(-\delta_i(x_j(t))^2/\eta_j^2)$, where $S_0^{(j)}(t)$ denotes the amplitude, $x_j(t)$ the location of target j at time t and η_j its effective size. In accordance with our earlier discussion, we set all $\eta_j \equiv \eta = 0.1$ and $S_0^{(j)} \equiv S_0 = 2$. Further, the signal is assumed to be contaminated by Gaussian noise with zero mean and standard deviation σ , the value of which is determined by the signal-to-noise ratio $\text{SNR} = S_0/\sigma$. The individual sensor false alarm probability was set to 0.1, which in turn determines the decision threshold τ , and the overall system false alarm was set to 0.1 as well (the probability of the fusion center making a false detection – see [54] for more details on how to control the system’s false alarm for each method). We compared the performance of EM algorithms based on pure binary decisions for the original decisions $\{Y_i\}$ and LVDF-corrected ones $\{Z_i\}$ (referred to as EM(Y) and EM(Z), respectively), hybrid

algorithms based on energies corresponding to positive decisions only (HEM(Y) and HEM(Z)), and the maximum likelihood estimates based on complete energy measurements (ML(E)) as a benchmark, since they are expected to be the most accurate. The reported results are based on 100 realizations of the noise where targets were detected by *all* the detection algorithms corresponding to the methods under consideration. For simplicity of comparisons, $\lambda = 0$ is used in simulations, and η is assumed known rather than estimated.

5.5.1 Choosing Starting Values for the Localization of Multiple Targets

Since many of the employed algorithms are iterative in nature, they prove sensitive to starting values. As an illustration, we first generated a signal emitted by a single target located at $x = [0.25, 0.25]$ with a signal amplitude $S_0 = 2$ and effective target size $\eta = 0.1$ (see Figure 5.2(a)). Further, we simulated energies at SNR set to 3 (see Figure 5.2(b)) and computed the corresponding initial and corrected decisions (Figure 5.2(c)). The log-likelihood function computed for energies achieves a global minimum near the true target location $\hat{x}_E = [0.25, 0.26]$, but has several local minima (Figure 5.2(d)). There are several options how one can choose a starting value for log-likelihood optimization: location of the sensor with maximum energy reading $x_{sE} = [0.21, 0.26]$, the centroid of the sensor locations corresponding to positive initial decisions $C(Y) = [0.51, 0.52]$, or the centroid of the sensor locations corresponding to positive corrected decisions $C(Z) = [0.26, 0.26]$. The last option clearly produces starting values closest to the global minimum, and hence to the true target location.

Now, consider more than one target ($p > 1$) present in the monitored area. The choice of the starting values becomes even more important and also more complicated. When it remains unclear how to choose the sensor locations of multiple energy

Figure 5.2: Target located at $x = [0.25, 0.25]$.

peaks, one could divide sensor locations corresponding to positive initial or corrected decisions into multiple clusters, and then select the centroids of these clusters as initial target locations.

There are a number of clustering techniques that can be used here. A comprehensive investigation suggests that using the centroids of positive decisions after clustering them using the K-means [38] and normalized cuts [101] perform well. It should be noted that the clustering method used affects the performance of the methods; for example, hierarchical clustering methods did not provide good starting values.

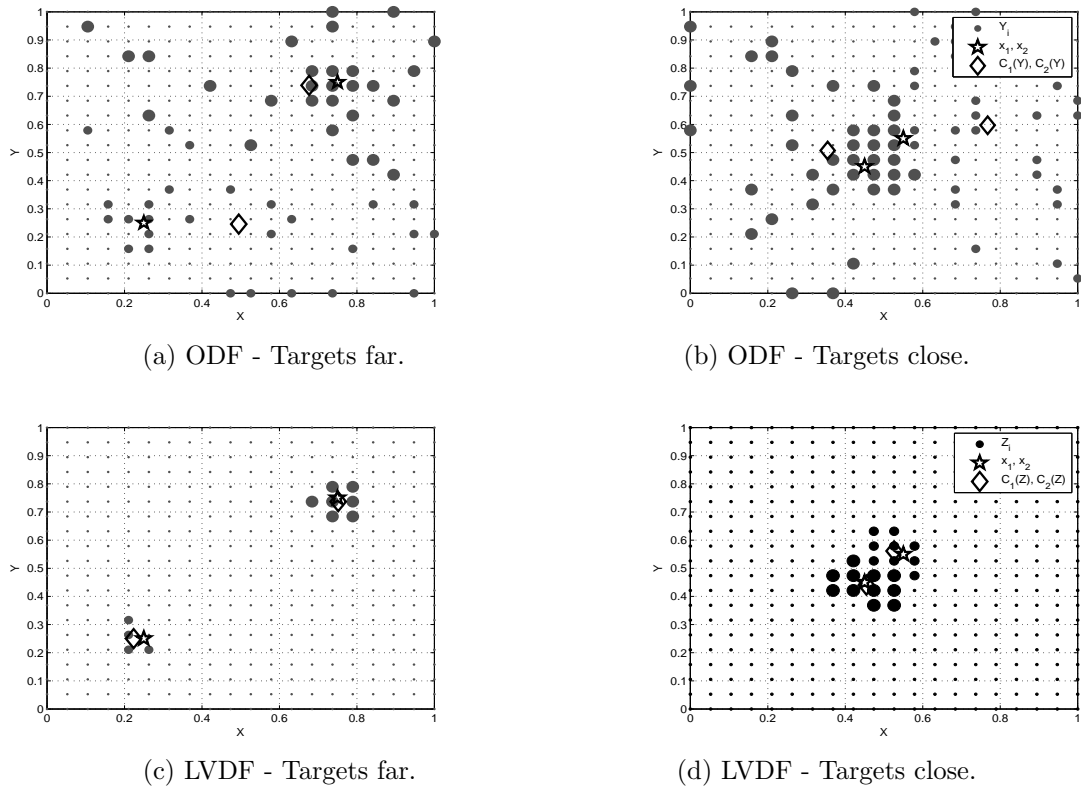


Figure 5.3: Clustering results of initial (upper panels) and corrected (lower panel) decisions.

Figure 5.3 shows the results of K-means clustering applied to sensor locations corresponding to the initial decisions and corrected decisions for two targets being far apart and close together respectively. We examine two scenarios involving two tar-

gets; the "far apart" locations are $x_1 = (0.25, 0.25)$ and $x_2 = (0.75, 0.75)$, and "close together" locations are $x_1 = (0.45, 0.45)$ and $x_2 = (0.55, 0.55)$. The corresponding energies, initial and corrected decisions for these two scenarios were obtained for WSN of 20×20 sensors with signal-to-noise ratio of 3. True target locations and the centroids of the clusters are sketched out in the Figure 5.3 as stars and diamonds respectively. There is very little difference between the quality of the clusters when targets are far apart, but as targets get closer clustering becomes an ambiguous and difficult task especially for ODF decisions. One can see that LVDF decisions produce more localized clusters, and hence provide better starting values.

Instead of averaging locations corresponding to all positive decisions within one cluster, one can compute the largest connected component of the graph of these decisions and average locations which belong to the component. In that case, both ODF and LVDF would be well localized, since taking two largest connected components of the ODF graph is an operation quite similar to LVDF, in the sense that it also removes distant false positives. However, LVDF can be performed locally within the network, whereas the largest connected components can only be determined by the fusion center; hence, performing LVDF on the spot will produce significant communications cost savings compared to transmitting all ODF decisions to the fusion center.

Further, to keep faithful to the algorithm, we apply this procedure to the original decisions Y for $EM(Y)$ and $HEM(Y)$, and to corrected decisions Z for $EM(Z)$ and $HEM(Z)$. For $ML(E)$, we use the centroids of LVDF decisions to provide the starting values, since those are generally more accurate.

5.5.2 Identifying the Location and Estimating the Number of Multiple Targets

One challenge in tracking comes from the presence of multiple targets with intersecting trajectories, especially when the number of targets is not assumed to be known. Hence, we first examine the algorithms' comparative ability to identify the locations and correct number of targets at a fixed point in time, contrasting performance results for targets far apart and close together with the same setup as discussed above.

First, we compare the localization accuracy of the various algorithms assuming that the number of targets is known. Table 5.1 gives the average distance from the estimated to the true target locations, which is computed by matching the estimated locations to true locations using the Hungarian algorithm [60].

Results in Table 5.1 show that overall, the LVDF methods clearly outperform the methods based on the original decisions. In general, the ML(E) and the HEM(Z) estimates are the most accurate, with ML(E) doing slightly better for higher SNRs and far apart targets, and HEM(Z) doing better for close targets and low SNR. In general, for the lowest SNR of 3 the LVDF methods are the most competitive, and when the targets are close together, the LVDF methods outperform ML(E) at all SNR levels, with the HEM(Z) exhibiting the best performance; ML(Z) and EM(Z) are slightly worse than HEM(Z) and fairly similar to each other.

Next, we compare the performance of the various algorithms when the number of targets is unknown, in the same simulation setting. The results on the estimated number of targets are given in Table 5.2. If the estimated number of targets is different from the true number of targets, some targets (either true or estimated ones) will be left unmatched. We do not include any penalty for this, since the choice of penalty would heavily affect the performance measure and should thus be

Table 5.1: Average distances between estimated and true target locations for two static targets, assuming $p = 2$ known.

Targets close							
SNR	ML(E)	ML(Y)	EM(Y)	HEM(Y)	ML(Z)	EM(Z)	HEM(Z)
3	0.0877	0.3152	0.1301	0.1163	0.0618	0.0606	0.0532
5	0.0461	0.3277	0.0672	0.0684	0.0382	0.0399	0.0294
10	0.0750	0.3067	0.0445	0.0411	0.0461	0.0312	0.0121
Targets far							
SNR	ML(E)	ML(Y)	EM(Y)	HEM(Y)	ML(Z)	EM(Z)	HEM(Z)
3	0.0262	0.1411	0.0343	0.0266	0.0255	0.0257	0.0225
5	0.0103	0.0781	0.0166	0.0116	0.0197	0.0199	0.0114
10	0.0051	0.0343	0.0136	0.0052	0.0164	0.0165	0.0052

left to the user; instead, we report separately the results on estimating the number of targets itself. Results on the average distance from true locations in Table 5.3 are consistent with results in Table 5.1 and in fact the estimated number of targets provides better discrimination between different algorithms in this case.

Table 5.2: Average estimate of the number of targets ($p = 2$).

Targets close							
SNR	ML(E)	ML(Y)	EM(Y)	HEM(Y)	ML(Z)	EM(Z)	HEM(Z)
3	1.79	1.04	1.00	2.55	1.73	1.06	1.73
5	2.05	1.12	1.00	2.60	1.98	1.30	2.09
10	2.13	1.34	1.00	2.66	2.02	1.74	2.23
Targets far							
SNR	ML(E)	ML(Y)	EM(Y)	HEM(Y)	ML(Z)	EM(Z)	HEM(Z)
3	2.01	2.11	1.49	2.67	2.03	1.96	2.00
5	2.02	2.10	1.96	2.56	2.08	2.01	2.10
10	2.01	1.98	2.00	2.60	2.08	2.03	2.03

Table 5.3: Average distances between estimated and true target locations for two static targets, assuming $p = 2$ unknown.

Targets close							
<i>SNR</i>	ML(E)	ML(Y)	EM(Y)	HEM(Y)	ML(Z)	EM(Z)	HEM(Z)
3.0	0.0440	0.0522	0.0520	0.1327	0.0538	0.0569	0.0423
5.0	0.0236	0.0548	0.0589	0.0994	0.0331	0.0539	0.0273
10.0	0.0146	0.0484	0.0606	0.0842	0.0285	0.0382	0.0193
Targets far							
<i>SNR</i>	ML(E)	ML(Y)	EM(Y)	HEM(Y)	ML(Z)	EM(Z)	HEM(Z)
3.0	0.0191	0.1054	0.0349	0.0971	0.0277	0.0268	0.0239
5.0	0.0121	0.0909	0.0168	0.0757	0.0230	0.0204	0.0169
10.0	0.0063	0.0407	0.0135	0.0770	0.0193	0.0188	0.0078

When the two targets are far apart, almost all methods prove fairly accurate, and ML(E) and all Z based methods are more accurate than Y based ones. On the

other hand, when targets are close, only the ML(E), ML(Z) and HEM(Z) methods performed well at all SNR values. Similar conclusions hold for a more complicated configuration of four targets (see Figure 5.4 and Table 5.4). To reduce the number of comparisons, from this point on we only report results for three algorithms that provide the best performance in each category: ML(E) (energies), ML(Z) (binary), and HEM(Z) (hybrid).

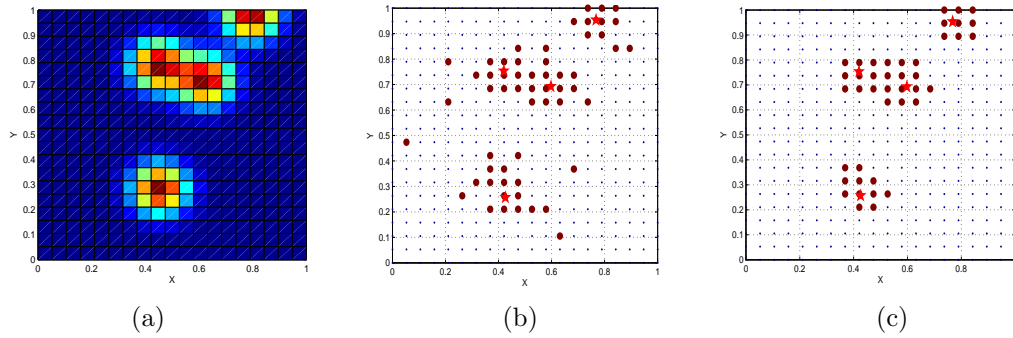


Figure 5.4: Four targets with $\eta = 0.1$ and $S_0 = 2$ located at $x_1 = (0.42, 0.26)$, $x_2 = (0.59, 0.69)$, $x_3 = (0.42, 0.75)$, $x_4 = (0.77, 0.95)$ in the monitored area R ; SNR=3. (a) Signal emitted by targets, (b) initial and (c) corrected decisions.

Table 5.4: The number of targets is assumed ($p = 4$) unknown.

Average Estimate of the Number of Targets.					
SNR	ML(E)	ML(Y)	HEM(Y)	ML(Z)	HEM(Z)
3.0	3.97	3.28	4.25	4.08	3.91
5.0	4.02	3.85	4.27	4.14	4.02
10.0	4.02	3.94	4.10	4.12	4.01
Average Minimal Distance.					
SNR	ML(E)	ML(Y)	HEM(Y)	ML(Z)	HEM(Z)
3.0	0.0222	0.0353	0.0469	0.0361	0.0283
5.0	0.0130	0.0306	0.0312	0.0288	0.0140
10.0	0.0072	0.0202	0.0130	0.0258	0.0069

5.5.3 Tracking of Multiple Targets

We now turn our attention to tracking multiple targets over time, again focusing on the difficulties encountered in applications. Suppose that information about one of the targets becomes lost due to sensor failure. Such a scenario is depicted in Figure 5.5(a), where three identical targets ($S_0 = 2$, $\eta = 0.1$) follow parallel linear

trajectories. At time slots $t = 7, 8$ (out of a total of 11) the information about the second target is lost, and the target 'reappears' at $t = 9$. Another difficulty is illustrated in Figures 5.5(b) and 5.5(c), where two targets travel very close to each other for a period of time, and the third target is also briefly lost. These issues make it challenging to estimate the number and location of the targets correctly. Figure 5.5 shows that our algorithm behaves as expected: when the signal is lost, it stops tracking the target in question and then starts tracking it as a new target once the signal is recovered. We are intentionally not enforcing any matching between newly appearing targets and targets that had disappeared earlier, but such matching could be easily implemented if needed. When two targets come close together, the two noise realizations show that the target labels are assigned arbitrarily once the targets separate, since in our scenario all targets are indistinguishable. If the targets have different signal amplitudes, they will be tracked correctly (results not shown). Note that in all cases, the number of targets is estimated correctly in these realizations. Finally, we show results on estimating the number of targets for the case of two

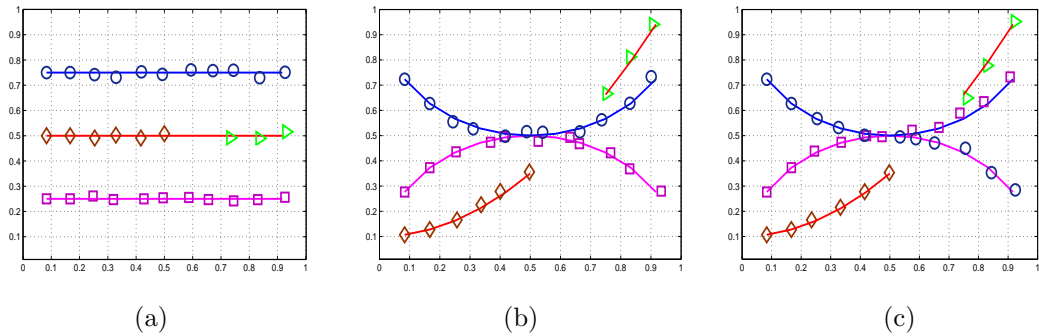


Figure 5.5: True trajectories (solid lines) and positions estimated by $ML(Z)$ at each time point for three targets with $SNR = 5$. (a) The signal from the second target is briefly lost; (b) Two targets come close together and the third target briefly loses signal; (c) Another noise realization for (b).

identical targets ($S_0 = 2$, $\eta = 0.1$) following crossing linear trajectories over twenty time slots. The trajectories intersect at $t = 10$, as shown in Figure 5.6(a). The

number of targets estimated over time by the three methods and averaged over 100 replications of the noise for SNR=5 is shown in Figure 5.6(b). When the targets are in exactly the same spot, ML(Z) tends to estimate them as one target, which is not surprising considering it uses binary data only. On the other hand, the other two methods that have access to energy measurements yield correct estimates even when the trajectories cross. These results are consistent with previously obtained ones; as targets move closer, the ML(Z) method experiences more difficulties in estimating their number correctly. Finally, we compare the tracking accuracy of our algorithms

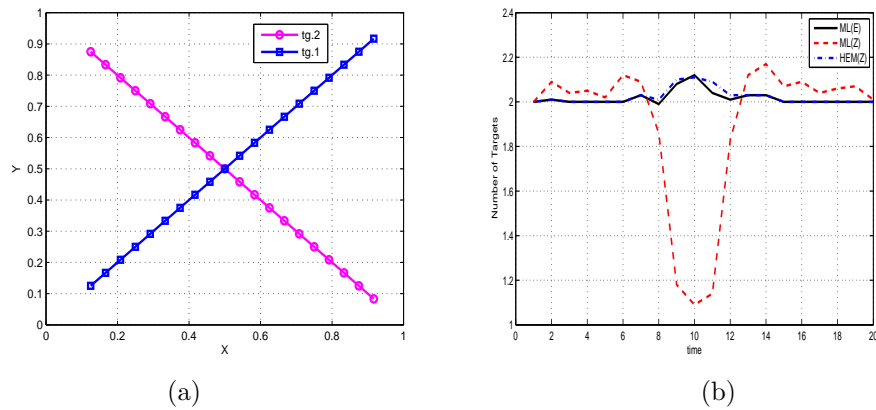


Figure 5.6: (a) True trajectories; (b) Average estimated number of targets as a function of time.

in the scenario of Figure 5.6 to their performance when they are provided with the correct, fixed number of targets. The results in Table 5.5 show that the accuracy is essentially the same, and even slightly better for the adaptive method. This is due to the fact that when the method can really only find one target but is forced to estimate two, it is likely to pick some arbitrary second location driven by noise.

Table 5.5: Distances from the true locations averaged over time and 100 noise replications for fixed and estimated number of targets, for two targets with intersecting linear trajectories.

SNR	Fixed ($p = 2$)			Adaptive		
	ML(E)	ML(Z)	HEM(Z)	ML(E)	ML(Z)	HEM(Z)
3	0.0329	0.0449	0.0376	0.0190	0.0292	0.0244
5	0.0185	0.0307	0.0195	0.0114	0.0224	0.0129
10	0.0086	0.0244	0.0089	0.0062	0.0193	0.0065

5.6 Applications

5.6.1 The NEST project

We start with a brief description of the data collection setup. The network consists of 144 sensors placed approximately 5 meters apart on a grid pattern. Each sensor has a sensing radius of 8 meters, a probability of detection 80% and that of raising a false alarm of 10%. Given the nature of the sensors (infrared), it is assumed that the model for the underlying signal is given by $S_i(t) = S_0/(1 + (\delta_i(x_j(t))/\eta))^3$. A network detection was declared if at least three sensors recorded positive decisions. The data available come in the form of positive decisions for each sensor. The test experiments involved 1, 2 or 3 people crossing the monitored area following linear intersecting trajectories, available from video recordings. The ML(Z) algorithm is suitable for the available data (there are no energy measurements available) and was applied with a neighborhood of $M = 9$ sensors.

In Figure 5.7, the ‘true’ (recovered from video recordings) and estimated trajectories are shown, for the three scenarios involving different numbers of targets. Table 5.6 gives averaged over time distances from the true target trajectories, for adaptive and fixed number of targets, and for smoothing vs no smoothing ($\lambda = 0$). Again, the adaptive version of ML(Z) outperforms the one with the number of targets fixed a priori, with the most apparent improvement in accuracy for the scenario with 3 people, since not all of them crossed the area simultaneously.

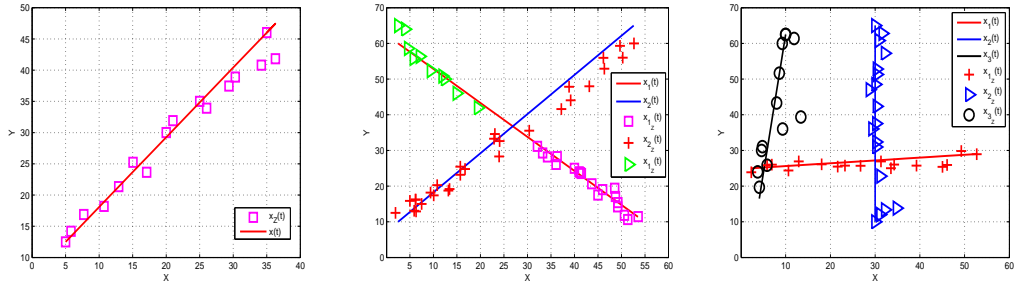
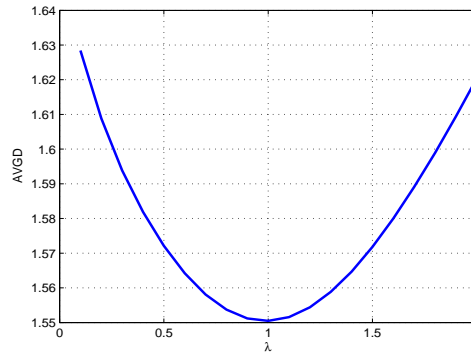


Figure 5.7: Estimated and true trajectories for one, two, and three NEST targets.

Table 5.6: Average distances from the true trajectories and estimated SNR.

Target	Fixed p , $\lambda = 0$			Adaptive, $\lambda = 0$		
	Sc.1	Sc.2	Sc.3	Sc.1	Sc.2	Sc.3
1	1.6553	5.9369	14.8523	1.6551	5.8832	2.9071
2	n/a	9.4583	11.0036	n/a	7.0816	1.8923
3	n/a	n/a	18.2545	n/a	n/a	4.0922
SNR	2.0961	1.7785	1.3643	2.0968	1.9652	2.0634
Target	Fixed p , $\lambda = 1$			Adaptive, $\lambda = 1$		
	Sc.1	Sc.2	Sc.3	Sc.1	Sc.2	Sc.3
1	1.5505	5.7325	17.5941	1.5505	5.2864	1.5156
2	n/a	8.5750	10.0829	n/a	6.8767	1.9553
3	n/a	n/a	19.3467	n/a	n/a	2.3720
SNR	2.0909	1.7663	1.3514	2.0909	2.0541	2.0374

Finally, various values of the smoothing parameter λ were tested and the most appropriate value $\lambda = 1$ was determined through a simulation study (see Figure 5.8). Since the estimates obtained from corrected decisions are already quite accurate without smoothing, the differences between smoothed and non-smoothed estimates are fairly small.

Figure 5.8: Averaged distance from the true trajectory of a single moving target (Sc.1) as a function of λ .

5.6.2 The ZebraNet Project

Unlike the highly controlled trajectories of people crossing the field in the NEST project, the trajectories of zebras in their natural habitat are highly irregular. The data were collected during from two ZebraNet deployments: a two-day deployment in January 2004 and a two-month deployment in the summer of 2005. During both deployments the sensors were equipped with GPS location devices and were actually fitted as collars on zebras. Due to hardware failures and severe weather conditions first experiment (2004) lasted for a few days and the data were collected for only one zebra for one day period each 8 minutes.

During the second deployment (2005), the data were collected for a two-month period from four zebras, selected for their varying behavioral patterns. Specifically, we have data on a bachelor male (id.6), actively searching for a mate, a female leader of the herd (id.10), a passive female with a characteristic of a very small home range (id.14), and another female zebra (id.8). The last zebra was reported to have had trouble with the collar position. The zebras' locations and a time stamp were recorded every 8 minutes for approximately 10 days, but due to hardware problems there are many missing values in the data, so we only use the time frame when the movements of all four zebras were recorded, which is just over 24 hours long. In general, the ZebraNet project found that placing sensing collars on zebras did not work very well, because some zebras managed to remove or lose the collars and there were other frequent hardware failures. Thus, it seems reasonable to consider a stationary sensor network of the NEST type for future deployments in this project instead.

In order to test the proposed algorithms on realistic trajectories, the following simulated sensor experiment was designed. A 20×20 sensor grid was simulated on

the unit square, and the true locations of the zebras available from the ZebraNet data were then mapped to this monitored region. The original monitored region is roughly $5 \times 5km$, so the simulated grid corresponds to sensors roughly $250m$ apart. The emitted signals were generated according to the model used in previous simulations, with the most challenging of the previously considered settings (SNR = 3). One random realization of the noise was used to generate the energies, following the noise model used in simulations.

Figure 5.9 shows the profile of the true animal trajectories for the first and the second deployments, respectively. In Figure 5.9(b) it is interesting to note that the trajectories of the zebras with ids 6, 8, and 10 intersect at some points in time, while zebra 14 remains isolated. The tracking results using the HEM(Z) algorithm are shown in Figure 5.10 and Figure 5.11 for one and the four zebras. They indicate that all the zebras are fairly well tracked at almost all points in time. Nevertheless, the more active zebras (id 6 and 10) prove the hardest to track.

Table 5.7 compares the performance of the three main algorithms when the number of targets is fixed in advance (at the true $p = 4$) to estimating it adaptively, with and without smoothing. For smoothing, an optimal λ was picked from the set $\{0, 100, 500, 1000, 5000\}$ for each zebra. We can see that, consistent with our previous results, adaptively estimating the number of targets yields significantly improved tracking. Further, smoothing the trajectories yields some improvement; however, the complex nature of the underlying trajectories, coupled with the sparse sampling over time, limit the gains from smoothing.

More detailed information on the estimated number of targets is provided in Table 5.8, which shows the percentage of time points where the number of targets was estimated as $2, \dots, 6$. Recall that the SNR is set to 3, and these results are

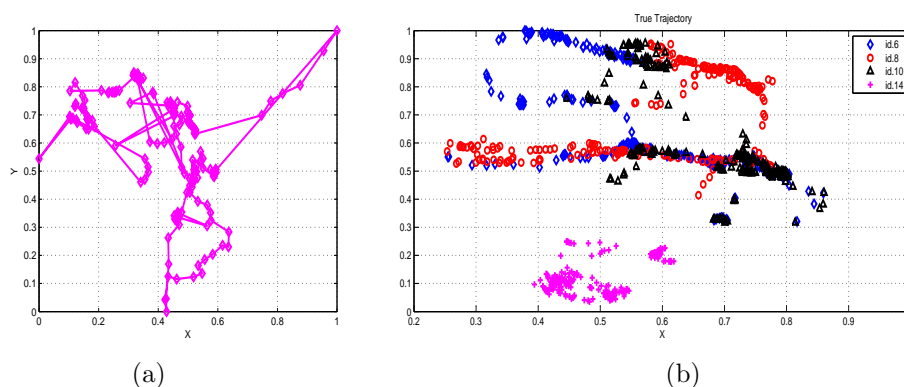


Figure 5.9: The recorded locations of the single zebra (a) and the four zebras (b) scaled and plotted on the unit square.

consistent with earlier simulation results: when the two zebras are very close together, the number of targets is likely to be estimated as 3 rather than 4, and particularly so by $ML(Z)$ which only has binary information available; extra targets are also sometimes picked up due to high noise levels, but they tend to be quickly dropped. Overall, it seems that these algorithms would be appropriate for tracking animals in natural environments, and their tracking performance can be further improved if additional discriminating information about the targets is available.

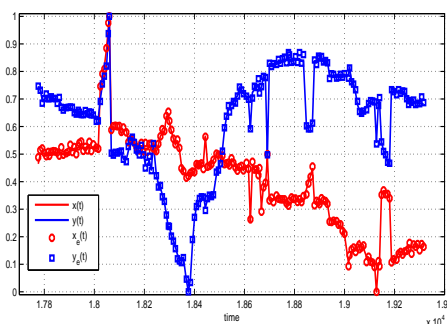


Figure 5.10: True and estimated by $HEM(Z)$ coordinates $x(t)$ and $y(t)$ for the zebra.

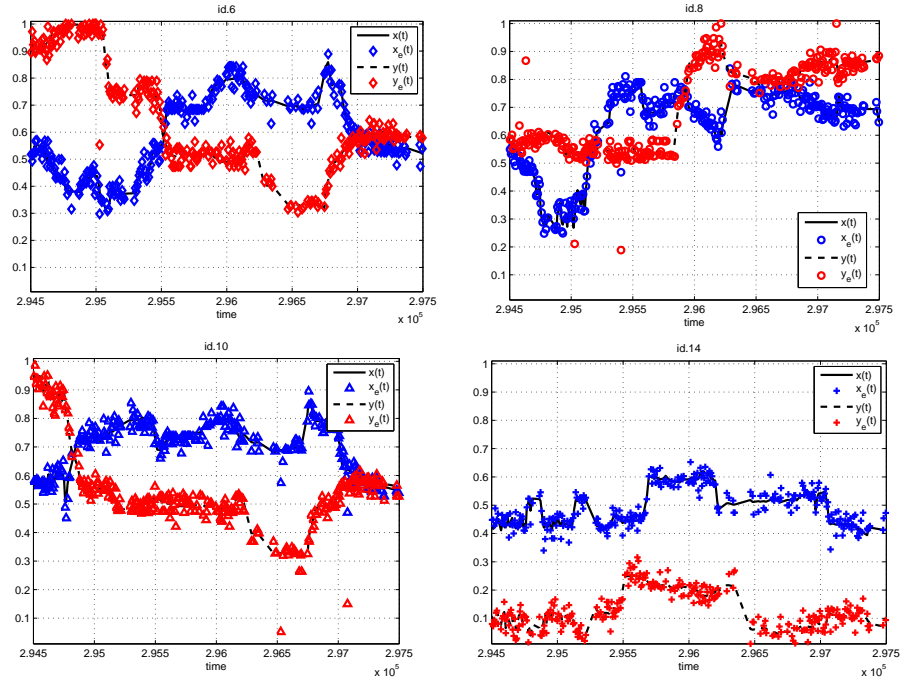


Figure 5.11: True and estimated by HEM(Z) coordinates $x(t)$ and $y(t)$ for the four zebras.

Table 5.7: Average distance from the true zebra trajectories. One unit of distance is approximately 5 km.

id	Fixed $p = 4$, $\lambda = 0$			Adaptive, $\lambda = 0$		
	ML(E)	ML(Z)	HEM(Z)	ML(E)	ML(Z)	HEM(Z)
6	0.0943	0.1140	0.0965	0.0270	0.0359	0.0298
8	0.0632	0.0655	0.0694	0.0252	0.0291	0.0314
10	0.0976	0.1200	0.0983	0.0243	0.0330	0.0302
14	0.0459	0.0452	0.0511	0.0230	0.0288	0.0300
id	Fixed $p = 4$, optimal λ			Adaptive, optimal λ		
	ML(E)	ML(Z)	HEM(Z)	ML(E)	ML(Z)	HEM(Z)
6	0.0805	0.1016	0.0888	0.0269	0.0288	0.0298
8	0.0621	0.0604	0.0622	0.0223	0.0264	0.0282
10	0.0849	0.0969	0.0841	0.0236	0.0219	0.0283
14	0.0378	0.0389	0.0437	0.0206	0.0280	0.0269

Table 5.8: The distribution of the estimated number of targets for zebra tracking (%).

	2	3	4	5	6
ML(E)	0.30	21.60	59.17	18.05	0.89
ML(Z)	10.95	54.44	26.33	7.69	0.59
HEM(Z)	0.30	18.93	57.69	21.01	2.07

CHAPTER VI

Local Data Fusion Framework for Classification in Wireless Sensor Networks and Beyond

6.1 Introduction

The goal of this chapter is to develop a general data fusion framework for classification that utilizes several ideas from LVDF. The framework can be viewed as local data fusion in WSN, where each sensor obtains energy measurements about one or several features of the environment, combines them with those of neighboring ones, and makes a decision about the presence or absence of the phenomenon of interest.

We will restrict our attention to linear classification rules. Thus, the proposed framework can be regarded as an extension of LDA (for review see [35], [37], [30], and [44]) and unlike other LDA extensions that use the optimal scoring [43], locally linear embedding [72], or locally weighted naive Bayes [36, 115], our new rule is still linear. The proposed set-up can also be viewed as a linear local fusion algorithm that takes into consideration the underlying (e.g., spatial) structure. It involves a collection of local classifiers that can consult their neighbors (in some suitably defined metric) and take their measurements into account. Fusion of classifiers obtained from multiple classification techniques has been addressed in [23]. However, in our framework we deal with the fusion of classification decisions obtained from multiple

local observations. We assume that each classifier knows the probability that its neighbors are observing the same class label as they are, which can be modeled through some local covariance model, and derive optimal weights (in terms of the probability of correct classification) for fusing data from such classifiers.

Obvious applications of the proposed framework are classifying spatial data and also fusing data from multiple sources with varying degrees of reliability. As motivation, we introduce the following two applications of proposed framework that will be further discussed in Section 6.4 of this chapter.

Our first application includes synthesized binary textures. Texture is the term used for repeated spatial patterns with local variations. Texture characteristics are widely used for data segmentation, classification and synthesis in many applications including computer vision [63], optics [94], multimedia [32],[122], [25],surveillance [62], and military applications [48], just to name a few. The modeling of texture is a common component of image analysis. The largest class of stochastic models commonly used for textures is Markov Random Fields (MRFs). The main disadvantage of MRFs is that there is no explicit form for the joint probability of the random variables describing the model. Special subclasses of MRF models that allow an explicit expression for the joint probability through conditional probabilities that correspond to the spatial dependence in the data have been developed by Abend [2], Pickard [91, 92, 93], Besag [10, 9], Qian and Titterington [96, 95], and others. In this study, we use Partially Ordered Markov Models (POMMs) developed by Davidson and Cressie[24] to generate several types of binary textures and evaluate performance of our fusion framework.

Our second application involves a data set that contains handwritten digits automatically scanned from envelopes by the U.S. Postal Service and then size and slant

normalized by the neural network group at AT&T research labs [22]. Each observation is a 16×16 grayscale image of an isolated digit (0-9). From the whole data set we choose the digits four (4) and nine (9), whose classification is rather challenging. This data set contains 852 and 821 images for four and nine, respectively.

The rest of the chapter is organized as follows. In Section 6.2 we describe our local fusion framework and its main properties, and in Section 6.3 we discuss several extensions of proposed framework. In Section 6.4 we evaluate the performance of our framework with application to synthesized binary textures and normalized handwritten digits.

6.2 Methods and Algorithms

6.2.1 Problem Formulation

In a standard two-class classification problem, we are given a set of training data $\{(X_i, Y_i)\}$, $i = 1, \dots, N$ with the input vector $X_i \in \mathbb{R}^d$ and a response $Y_i \in \{0, 1\}$, which is the true class label for object i . For the training data it is usually assumed that the observations $\{(X_i, Y_i)\}$ are an independent and identically distributed sample from an unknown joint distribution, and that given class labels Y_i the input features X_i are conditionally independent and identically distributed. The main objective is to find a classification rule based on the training data in order to be able to predict a class label Y for a new object given the vector of its features X . The optimal Bayes classification rule is based on a posterior conditional distribution of the object being in class $Y = c$ given X , $\mathbb{P}(Y = c|X)$, and is defined as:

$$(6.1) \quad \hat{Y} = \operatorname{argmax}_{c \in \{0,1\}} \mathbb{P}(Y = c|X).$$

The probability of correct classification $P_x(X)$ for a single observation is given by:

$$(6.2) \quad P_x(X) = \pi_1 \mathbb{P}(\hat{Y}(X) = 1|Y = 1) + \pi_0 \mathbb{P}(\hat{Y}(X) = 0|Y = 0).$$

6.2.2 The Linear Discriminant Rule

Classical Linear Discriminant Analysis (LDA) is based on an assumption that the conditional distribution functions of X given $Y = 1$ and $Y = 0$, denoted $f_1(X)$ and $f_0(X)$, are multivariate normal with common covariance matrix Σ and means μ_1 and μ_0 for class 1 and class 0, respectively:

$$\begin{aligned} f_1(X) &= \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp(-0.5(X - \mu_1)'\Sigma^{-1}(X - \mu_1)), \\ f_0(X) &= \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp(-0.5(X - \mu_0)'\Sigma^{-1}(X - \mu_0)). \end{aligned}$$

In theory, an optimal classification rule (6.1) for LDA can be written as follows:

$$(6.3) \quad \hat{Y}(d_x(X)) = \begin{cases} 1, & d_x(X) \geq \log(\pi_0/\pi_1) \\ 0, & d_x(X) < \log(\pi_0/\pi_1), \end{cases}$$

where $\pi_1 = \mathbb{P}(Y = 1)$ and $\pi_0 = \mathbb{P}(Y = 0)$ are the prior class probabilities and $d_x(X)$ is a linear function that projects the d -dimensional vector X to the one-dimensional space of maximum class separability, i.e.:

$$(6.4) \quad d_x(X) = \frac{\log(f_{X|Y=1}(X))}{\log(f_{X|Y=0}(X))} = (\mu_1 - \mu_0)'\Sigma^{-1} \left(X - \frac{1}{2}(\mu_1 + \mu_0) \right).$$

The Bayes risk, i.e., the probability of correct classification $P_x(X)$ under the Bayes rule is:

$$\begin{aligned} P_x(X) &= \pi_1 \mathbb{P}(d_x(X) \geq \log(\pi_0/\pi_1) | Y = 1) + \pi_0 \mathbb{P}(d_x(X) < \log(\pi_0/\pi_1) | Y = 0) \\ (6.5) \quad &= \pi_1 \bar{\Phi} \left(\frac{\log(\pi_0/\pi_1) - d(\mu_1)}{D(\mu_1, \mu_0)} \right) + \pi_0 \Phi \left(\frac{\log(\pi_0/\pi_1) - d(\mu_0)}{D(\mu_1, \mu_0)} \right), \end{aligned}$$

where $D(\mu_1, \mu_0)$ is the Mahalanobis distance between the classes:

$$D(\mu_1, \mu_0) = |(\mu_1 - \mu_0)'\Sigma^{-1}(\mu_1 - \mu_0)|^{1/2}.$$

The unknown parameters μ_0 , μ_1 , Σ , and π_0 and π_1 are estimated from training data as:

$$(6.6) \quad \begin{aligned} \hat{\pi}_c &= N_c/N, \\ \hat{\mu}_c &= \sum_{Y_i=c} X_i/N_c, \\ \hat{\Sigma} &= \sum_{c \in \{0,1\}} \sum_{Y_i=c} (X_i - \hat{\mu}_c)(X_i - \hat{\mu}_c)' / (N - 2), \end{aligned}$$

where N_c ($c \in \{0, 1\}$) is the number of class- c observations.

Given estimated values of μ_1 , μ_0 , Σ , π_0 , π_1 , and a vector of input features X , one can easily assign the corresponding class label Y using (6.4) and evaluate the probability of correct classification using (6.5). Compared to other more complicated classification techniques, LDA has the advantages of linearity, shift invariance, and low computational cost.

6.2.3 A Local Data Fusion for Classification

It should be noted that LDA makes no use of any structure in the training data. If there is in fact some local structure in the data, accounting for it will improve classification performance. We propose a Local Data Fusion (LDF) algorithm according to which we combine the input features X with features in the training data set in the neighborhood of the object of interest, denoted $U(X)$.

Let Z define a fused feature vector:

$$(6.7) \quad Z = w_0 X + \sum_{h \in U(X)} w_h X_h.$$

The idea is to use the fused feature Z instead of X to predict the underlying class label Y . The weights w_0, w_h , $h \in U(X)$ are to be constructed by maximizing the probability of correct classification P_z :

$$P_z = \max_{\{w_0, w_h, h \in U(X)\}} \left\{ \pi_1 \mathbb{P}(\hat{Y}(Z) = 1 | Y = 1) + \pi_0 \mathbb{P}(\hat{Y}(Z) = 0 | Y = 0) \right\}.$$

The optimal Bayes classification rule for Z can be constructed using the ratio of the conditional distribution functions $f_1(Z)$ and $f_0(Z)$ of Z given class $Y = 1$ and $Y = 0$, respectively:

$$\hat{Y}(Z) = \begin{cases} 1, & \text{if } \frac{f_1(Z)}{f_0(Z)} \geq \frac{\pi_0}{\pi_1} \\ 0, & \text{if } \frac{f_1(Z)}{f_0(Z)} < \frac{\pi_0}{\pi_1}, \end{cases}$$

To compute $f_1(Z)$ and $f_0(Z)$, we first consider the simplest local class dependency structure:

$$(6.8) \quad \begin{aligned} \mathbb{P}(Y_h = 1 | Y = 1) &= p, \\ \mathbb{P}(Y_h = 0 | Y = 0) &= p, \end{aligned}$$

i.e., we assume a constant probability of having the same class labels for all neighbors. According to this assumption, we assign weight w_0 to the input vector X and w_1 to all remaining observations in the neighborhood $U(X)$. We define a random variable Q as the number of objects in $U(X)$ with the same class label as Y . Assuming also that all Y_h ($h \in U(X)$) are conditionally independent given Y , variable Q follows a binomial distribution with parameters p and $M = |U(X)|$, conditional on Y . Hence, the conditional density $f_1(Z)$ is computed as follows:

$$\begin{aligned} f_1(Z) &\equiv f(Z|Y = 1) = \mathbb{E}_Q \{f(Z|Q, Y = 1)\} = \\ &= \sum_{q=0}^M f(Z|Q = q, Y = 1) \mathbb{P}(Q = q|Y = 1) = \\ &= \sum_{q=0}^M \phi(Z; \mu_{z_1}(q), \Sigma_z) \binom{M}{q} p^q (1-p)^{M-q}, \end{aligned}$$

where $\phi(Z; \mu_{z_1}(q), \Sigma_z)$ is the multivariate normal density with mean $\mu_{z_1}(q)$ and co-

variance matrix Σ_z . Similarly, $f_0(Z)$ is given by,

$$\begin{aligned} f_0(Z) &\equiv f(Z|Y=0) = \mathbb{E}_Q \{f(Z|Q, Y=1)\} = \\ &= \sum_{q=0}^M \phi(Z; \mu_{z_0}(q), \Sigma_z) \binom{M}{q} p^q (1-p)^{M-q}. \end{aligned}$$

Values of $\mu_{z_1}(q) \equiv \mu_z(Q=q, Y=1)$, $\mu_{z_0}(q) \equiv \mu_z(Q=q, Y=0)$ and Σ_z can be calculated as:

$$\begin{aligned} \mu_{z_1}(q) &= w_0 \mu_1 + w_1 (q \mu_1 + (M-q) \mu_0), \\ \mu_{z_0}(q) &= w_0 \mu_0 + w_1 (q \mu_0 + (M-q) \mu_1), \\ \Sigma_z &= (w_0^2 + M w_1^2) \Sigma. \end{aligned} \tag{6.9}$$

The weights are only defined up to a scaling constant, so we need to impose a constraint. One convenient option is to have $\Sigma_z = \Sigma$, setting

$$w_0^2 + M w_1^2 = 1.$$

In general, one can also account for more complex neighborhood structures by grouping neighbors within $U(X)$ into k layers, $L_j(X) \in U(X)$, $j \in \{1, \dots, k\}$, so that within each layer $L_j(X)$ objects have a constant probability of having the same class label Y , i.e.

$$\mathbb{P}(Y_l = 1|Y = 1) = \mathbb{P}(Y_l = 0|Y = 0) = p_j, l \in L_j(X).$$

In the extreme case of $k = |U(X)|$, all neighbors have different probabilities of agreement.

In this data structure, we assign different weights w_j , $j \in \{1, \dots, k\}$ to the input characteristics in different layers. We define a random vector $Q = (Q_0, Q_1, \dots, Q_k)$ where each Q_j , $j = 1, \dots, k$ represents the number of objects in layer $L_j(X)$ that

have the same class label value as Y . Note that Q_0 corresponds to the input X itself, so $Q_0 = 1$ with probability $p_0 = \mathbb{P}(Y = Y) = 1$. Given the class label Y , class labels in different layers and within each layer can preserve some degree of dependence; however, here we adopt a pseudo-likelihood formulation [10] and assume that all neighbors Y_h are conditionally independent. Letting M_j in layer $L_j(X)$, the distribution of Q can be defined as:

$$(6.10) \quad \begin{aligned} P(Q = (q_0, q_1, \dots, q_k)) &= \prod_{j=1}^k \mathbb{P}(Q_j = q_j), \text{ where} \\ P(Q_j = q_j) &= \binom{M_j}{q_j} p_j^{q_j} (1 - p_j)^{M_j - q_j}, \quad j = 1, \dots, k. \end{aligned}$$

The conditional distribution functions of Z given $Y = 1$ and $Y = 0$, i.e., $f_1(Z)$ and $f_0(Z)$, in the presence of multiple layers are calculated as follows:

$$(6.11) \quad \begin{aligned} f_1(Z) &= \sum_{q=(q_0, \dots, q_k) \in \Omega_Q} \phi(Z; \mu_{z_1}(q), \Sigma_z) \prod_{j=1}^k \mathbb{P}(Q_j = q_j), \\ f_0(Z) &= \sum_{q=(q_0, \dots, q_k) \in \Omega_Q} \phi(Z; \mu_{z_0}(q), \Sigma_z) \prod_{j=1}^k \mathbb{P}(Q_j = q_j), \end{aligned}$$

where the values of the conditional means $\mu_{z_1}(q) \equiv \mu_{z|Y=1}(q = (q_0, \dots, q_k), w_0, \dots, w_k)$, $\mu_{z_0}(q) \equiv \mu_{z|Y=0}(q = (q_0, \dots, q_k), w_0, \dots, w_k)$ are computed as:

$$(6.12) \quad \begin{aligned} \mu_{z_1}(q) &= \sum_{j=0}^k w_j (q_j \mu_1 + (M_j - q_j) \mu_0), \\ \mu_{z_0}(q) &= \sum_{j=0}^k w_j (q_j \mu_0 + (M_j - q_j) \mu_1). \end{aligned}$$

and we again impose the constraint that forces $\Sigma_z = \Sigma$, i.e.,

$$\sum_{j=0}^k w_j^2 M_j = 1.$$

Recall that the Bayes rule is given by the ratio $R(Z) = \frac{f_1(Z)}{f_0(Z)}$, which can be written as follows:

$$(6.13) \quad \mathbf{B}(Z) = \frac{\mathbb{E}_Q \{ \phi(Z; \mu_{z_1}(Q), \Sigma_z) \}}{\mathbb{E}_Q \{ \phi(Z; \mu_{z_0}(Q), \Sigma_z) \}} = \frac{\sum_{q \in \Omega_Q} \phi(Z; \mu_{z_1}(q), \Sigma_z) \prod_{j=1}^k \mathbb{P}(Q_j = q_j)}{\sum_{q \in \Omega_Q} \phi(Z; \mu_{z_0}(q), \Sigma_z) \prod_{j=1}^k \mathbb{P}(Q_j = q_j)}.$$

For Q having more than one possible value, $R(Z)$ is a complicated, non-linear function of Z , and we cannot compute these expectations explicitly. However, we can compute a simpler classification function $d_z(Z)$ as:

$$(6.14) \quad d_z(Z) = \sum_{q \in \Omega_Q} (\mu_{z_1}(q) - \mu_{z_0}(q))' \Sigma^{-1} \left(Z - \frac{1}{2}(\mu_{z_1}(q) + \mu_{z_0}(q)) \right) \prod_{j=1}^k \mathbb{P}(Q_j = q_j).$$

Proposition VI.1. *Classification function $d_z(Z)$ is a linear function of Z for all values of μ_1 , μ_0 , Σ , and p .*

Proof. Given

$$\begin{aligned} \mu_{z_1}(q) - \mu_{z_0}(q) &= \sum_{j=0}^k w_j (2q_j - M_j) (\mu_1 - \mu_0), \\ \mu_{z_1}(q) + \mu_{z_0}(q) &= \sum_{j=0}^k w_j M_j (\mu_1 + \mu_0), \end{aligned}$$

the expression for $d_z(Z)$ defined by (6.14) is simplified to:

$$d_z(Z) = \sum_{j=0}^k w_j M_j (2p_j - 1) (\mu_1 - \mu_0)' \Sigma^{-1} \left(Z - \frac{1}{2} \sum_{j=0}^k w_j M_j (\mu_1 + \mu_0) \right),$$

which is linear in Z . □

Introducing additional notation,

$$S_1 = \sum_{j=0}^k w_j M_j, \quad S_2 = \sum_{j=0}^k w_j M_j (2p_j - 1), \quad S_3 = \sum_{j=0}^k w_j (2q_j - M_j),$$

we can reduce the expression for $d_z(Z)$ to the form of (6.4):

$$d_z(Z) = S_2 (\mu_1 - \mu_0)' \Sigma^{-1} \left(Z - \frac{S_1}{2} (\mu_1 + \mu_0) \right).$$

Recall that

$$\phi(Z; \mu_{z_1}(Q), \Sigma_z) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp \left\{ -0.5 (Z_i - \mu_{z_1}(Q))' \Sigma_z^{-1} (Z - \mu_{z_1}(Q)) \right\},$$

so that equation (6.14) can be also written as:

$$\begin{aligned}
d_z(Z) &= \sum_{q \in \Omega_q} (\mu_{z_1}(q) - \mu_{z_0}(q))' \Sigma^{-1} \left(Z - \frac{1}{2}(\mu_{z_1}(q) + \mu_{z_0}(q)) \right) \prod_{j=1}^k \mathbb{P}(Q_j = q_j) \\
&= \mathbb{E}_Q \left\{ \log \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} - 0.5(Z - \mu_{z_1}(Q))' \Sigma^{-1} (Z - \mu_{z_1}(Q)) \right\} \\
&\quad - \mathbb{E}_Q \left\{ \log \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} - 0.5(Z - \mu_{z_0}(Q))' \Sigma^{-1} (Z - \mu_{z_0}(Q)) \right\}.
\end{aligned}$$

Hence, the expression 6.14 coincides with:

$$(6.15) \quad d_z(Z) = \mathbb{E}_Q \{ \log \phi(Z; \mu_{z_1}(Q), \Sigma_z) \} - \mathbb{E}_Q \{ \log \phi(Z; \mu_{z_0}(Q), \Sigma_z) \}.$$

Given Z , the class label \hat{Y} is assigned according to:

$$(6.16) \quad \hat{Y}(Z) = \begin{cases} 1, & d_z(Z) \geq \log(\pi_0/\pi_1) \\ 0, & d_z(Z) < \log(\pi_0/\pi_1), \end{cases}$$

The corresponding probability of correct classification in this case can be calculated explicitly:

$$\begin{aligned}
P_z &= \max_{\{w_0, \dots, w_k\}} \left\{ \pi_1 \mathbb{E}_Q \bar{\Phi} \left(\frac{\log \pi_0/\pi_1 - d_z(\mu_{z_1}(Q))}{|S_2| D(\mu_1, \mu_0)} \right) \right. \\
&\quad \left. + \pi_0 \mathbb{E}_Q \Phi \left(\frac{\log \pi_0/\pi_1 - d_z(\mu_{z_0}(Q))}{|S_2| D(\mu_1, \mu_0)} \right) \right\} \\
(6.17) \quad &= \max_{\{w_0, \dots, w_k\}} \left\{ \pi_1 \sum_{q \in \Omega_Q} \bar{\Phi} \left(\frac{\log \pi_0/\pi_1 - d_z(\mu_{z_1}(q))}{|S_2| D(\mu_1, \mu_0)} \right) \prod_{j=1}^k \mathbb{P}(Q_j = q_j) \right. \\
&\quad \left. + \pi_0 \sum_{q \in \Omega_Q} \Phi \left(\frac{\log \pi_0/\pi_1 - d_z(\mu_{z_0}(q))}{|S_2| D(\mu_1, \mu_0)} \right) \prod_{j=1}^k \mathbb{P}(Q_j = q_j) \right\}.
\end{aligned}$$

The next two propositions derive the main properties of the function P_z .

Proposition VI.2. *The probability of correct classification P_z based on fused features Z is mean shift invariant.*

Proof. We need to show that if μ_0, μ_1 are replaced by $\mu_0 + c, \mu_1 + c$, respectively, P_z will stay the same.

Recall that P_z depends on μ_1 and μ_0 only through the functions $D_M(\mu_1, \mu_0)$, $d_z(\mu_{z_1}(q))$, and $d_z(\mu_{z_0}(q))$. Notice that $D(\mu_1, \mu_0)$ is a function of difference in means ($\mu_1 - \mu_0$), and therefore is shift invariant. Also, $d_z(\mu_{z_1}(q))$, and $d_z(\mu_{z_0}(q))$ can be expressed in terms of the previously defined sums S_2 and S_3 :

$$\begin{aligned} d_z(\mu_{z_1}(q)) &= S_2(\mu_1 - \mu_0)' \Sigma^{-1} \left(\mu_{z_1}(q) - \frac{1}{2} S_1(\mu_1 + \mu_0) \right) = \\ &= 0.5 S_2 S_3 D^2(\mu_1, \mu_0), \\ d_z(\mu_{z_0}(q)) &= -0.5 S_2 S_3 D^2(\mu_1, \mu_0). \end{aligned}$$

Since neither S_2 or S_3 depend on μ_0 , μ_1 , both $d_z(\mu_{z_1}(q))$ and $d_z(\mu_{z_0}(q))$ only depend on μ_0 , μ_1 through $D(\mu_1, \mu_0)$, and therefore are shift invariant.

This completes the proof of the Proposition. \square

Proposition VI.3. *The probability of correct classification P_z based on fused features Z is greater or equal to the probability of correct classification for LDA for all values of μ_1 , μ_0 , Σ , and p .*

Proof. Recall that P_z is maximized over $\{w_0, \dots, w_1\}$, and therefore P_z is greater or equal to the inner expression in (6.18) with any weights w_0, \dots, w_1 satisfying the constraint $\sum_{j=0}^k w_j^2 M_j = 1$; for instance, $w_0 = 1$ and $w_1 = w_2 = \dots = w_k = 0$. With these weights, $S_2 = 1$, $d_z(Z) = d_x(X)$, $\mu_{z_1}(q) = \mu_1$, and $\mu_{z_0}(q) = \mu_0$, so that

$$\begin{aligned} P_z &\geq \{w_0 = 1, w_1 = 0, \dots, w_k = 0\} \left\{ \pi_1 \mathbb{E}_Q \bar{\Phi} \left(\frac{\log \pi_0 / \pi_1 - d_z(\mu_{z_1}(Q))}{|S_2| D(\mu_1, \mu_0)} \right) \right. \\ &\quad \left. + \pi_0 \mathbb{E}_Q \Phi \left(\frac{\log \pi_0 / \pi_1 - d_z(\mu_{z_0}(Q))}{|S_2| D(\mu_1, \mu_0)} \right) \right\} \\ &= \pi_1 \bar{\Phi} \left(\frac{\log(\pi_0 / \pi_1) - d_x(\mu_1)}{D(\mu_1, \mu_0)} \right) + \pi_0 \Phi \left(\frac{\log(\pi_0 / \pi_1) - d_x(\mu_0)}{D(\mu_1, \mu_0)} \right) = P_x. \end{aligned}$$

\square

Corollary VI.4. *The probability of correct classification P_z based on fused vector Z achieves its minimum, $P_z \equiv P_x$:*

$$\pi_1 \bar{\Phi} \left(\frac{\log(\pi_0/\pi_1) - 0.5D^2(\mu_1, \mu_0)}{D(\mu_1, \mu_0)} \right) + \pi_0 \Phi \left(\frac{\log(\pi_0/\pi_1) + 0.5D^2(\mu_1, \mu_0)}{D(\mu_1, \mu_0)} \right),$$

when $p_1 = \dots = p_k = 0.5$ with $w_0 = 1$, $w_1 = \dots = w_k = 0$.

Numerically, it can be shown that the probability of correct classification P_z achieves its maximum:

$$\begin{aligned} \max(P_z) &= \pi_1 \bar{\Phi} \left(\frac{\log(\pi_0/\pi_1) - 0.5 \sum_{j=0}^k M_j D^2(\mu_1, \mu_0)}{\sqrt{\sum_{j=0}^k M_j D(\mu_1, \mu_0)}} \right) \\ &+ \pi_0 \Phi \left(\frac{\log(\pi_0/\pi_1) + 0.5 \sum_{j=0}^k M_j D^2(\mu_1, \mu_0)}{\sqrt{\sum_{j=0}^k M_j D(\mu_1, \mu_0)}} \right), \end{aligned}$$

when $p_j = 1$ or 0 with

$$w_j = \begin{cases} \frac{1}{\sqrt{\sum_{j=0}^k M_j}}, & \text{if } p_j = 1 \\ -\frac{1}{\sqrt{\sum_{j=0}^k M_j}}, & \text{if } p_j = 0. \end{cases}$$

For illustration purposes, we explored the one-layer neighborhood structure and plotted the probability of correct classification P_z (Figure 6.1(b)) and corresponding optimal weights (Figure 6.1(a)) as a function of p . Figure 6.1 shows that for $p \in [0, 1/2)$, the weight w_1 is negative (see (a)) and P_z is decreasing in p . Analogously, when $p \in [1/2, 1]$, the weight w_1 is positive and P_z is an increasing function of p . These properties match the intuition that the more likely your neighbors are to be of the same class, the more it helps to fuse (when $p > 1/2$). The whole process is symmetric around $p = 1/2$.

Corollary VI.5. *For special cases when $p_j = 1$ or 0 for all $j = 1, \dots, k$, the linear decision rule defined by the d_z (6.14) and the Bayes rule defined by the R_z (6.13) are equivalent.*

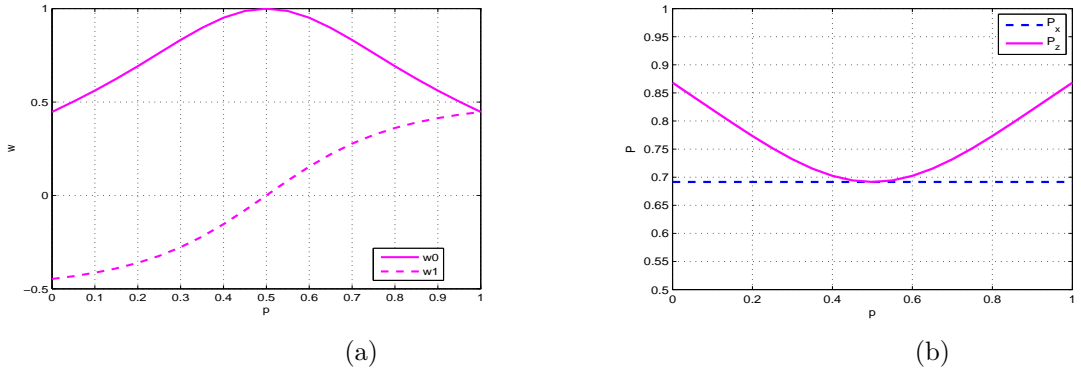


Figure 6.1: (a) Weights and (b) the probability of correct classification as a function of p . ($d = 1$, $\pi_0/\pi_1 = 1$, $\sigma = 1$, $\mu_1 = 1$, $\mu_0 = 0$)

Proof. Recall that by construction we have:

$$\begin{aligned} \log(R(Z)) &= \log \mathbb{E}_Q \{ \phi(\mu_{z_1}(Q), \Sigma_z, Z) \} - \log \mathbb{E}_Q \{ \phi(\mu_{z_0}(Q), \Sigma_z, Z) \}, \\ d_z(Z) &= \mathbb{E}_Q \{ \log \phi(\mu_{z_1}(Q), \Sigma_z, Z) \} - \mathbb{E}_Q \{ \log \phi(\mu_{z_0}(Q), \Sigma_z, Z) \}. \end{aligned}$$

When $p_j = 1$, the probability $\mathbb{P}(Q_j = q_j) = \binom{M_j}{q_j} p_j^{q_j} (1 - p_j)^{M_j - q_j} \neq 0$ only when $q_j = M_j$. Similarly, when $p_j = 0$, $\mathbb{P}(Q_j = q_j) \neq 0$ for $q_j = 0$. In this case, the sample space of Q is reduced to a single term Q_1 which consists of zeros and M_j depending on the values of p_j 's. So the expectations in the formulas for $d_z(Z)$ and $R(Z)$ are taken with respect to a point mass distribution $P(Q = Q_1) = 1$. It is clear that under these conditions $\log(R(Z)) = d_z(Z)$, and consequently $\hat{Y}(R(Z)) = \hat{Y}(d_z(Z))$. \square

Next Corollary shows that in the one-dimensional case, the proposed linear rule $d_z(Z)$ coincides with the optimal Bayes rule $R(Z)$.

Corollary VI.6. *In the one-dimensional case, when $\pi_0 = \pi_1$ and $R(Z)$ is a monotone function, the classification rules defined by functions $R(Z)$ and $d_z(Z)$ produce the same class label \hat{Y} .*

Proof. In order to assign class label \hat{Y} to the object with features X we compare the classification function $d_z(Z)$ to $\log(\pi_0/\pi_1)$, and classification function $R(Z)$ to $\frac{\pi_0}{\pi_1}$. When $\pi_0 = \pi_1$, the boundary conditions for $d_z(Z)$ and $R(Z)$ are defined, respectively, as $d_z(Z) = 0$ and $R(Z) = 1$. If $R(Z)$ is a monotone function of Z , then there is only one solution to the equation $R(Z) = \frac{\pi_0}{\pi_1}$, or $R(Z) = 1$, if $\frac{\pi_0}{\pi_1} = 1$. By solving the equation $d_z(\alpha_z) = 0$, we obtain the decision boundary α_z :

$$\alpha_z = \frac{\mu_1 + \mu_0}{2} \sum_{j=0}^k w_j M_j.$$

Next we will establish that if $\frac{\pi_0}{\pi_1} = 1$, then the solution to $d_z(Z) = 0$ ($Z = \alpha_z$) is also the solution to the equation $R(Z) = 1$.

When $d = 1$, the function $R(Z)$ for each q depend on $Z - \mu_{z_1}(q)$ in the numerator and $Z - \mu_{z_0}(q)$ in the denominator. We next calculate these values for α_z .

$$\begin{aligned} \alpha_z - \mu_{z_1}(q) &= -0.5(\mu_1 - \mu_0) \sum_{j=0}^k w_j (2q_j - M_j), \\ \alpha_z - \mu_{z_0}(q) &= 0.5(\mu_1 - \mu_0) \sum_{j=0}^k w_j (2q_j - M_j). \end{aligned}$$

By plugging them into the formula of $R(Z)$ and using the symmetry of the normal density function, we get that $R(\alpha_z) = 1$, i.e. $R(\alpha_z) = \frac{\pi_0}{\pi_1}$. \square

6.3 Extensions

The most straightforward extension of the proposed framework is to assume the conditional probabilities of neighbors Y_h ($h \in U(X)$) having the same class labels as Y to be different for $Y = 1$ and $Y = 0$, i.e., to assume that

$$\begin{aligned} \mathbb{P}(Y_h = 1|Y = 1) &= p_1, \\ \mathbb{P}(Y_h = 0|Y = 0) &= p_0. \end{aligned}$$

The conditional distribution functions $f_1(Z)$ and $f_0(Z)$ are then computed as follows:

$$\begin{aligned}
f_1(Z) &= \mathbb{E}_{Q_1} f(Z|Q_1, Y = 1) = \\
&= \sum_{q_1=0}^M \phi(Z; \mu_{z_1}(q_1), \Sigma_z) \binom{M}{q_1} p_1^{q_1} (1 - p_1)^{M-q_1}, \\
f_0(Z) &= \mathbb{E}_{Q_0} f(Z|Q_0, Y_i = 1) = \\
&= \sum_{q_0=0}^M \phi(Z; \mu_{z_0}(q_0), \Sigma_z) \binom{M}{q_0} p_0^{q_0} (1 - p_0)^{M-q_0},
\end{aligned}$$

where Q_1 and Q_0 follow the binomial distribution with parameters p_1 and M , and p_0 and M , respectively.

Values for $\mu_{z_1}(q_1) \equiv \mu_z(Q_1 = q_1, Y = 1)$, $\mu_{z_0}(q_0) \equiv \mu_z(Q_0 = q_0, Y = 0)$ and Σ_z can be calculated as:

$$\begin{aligned}
\mu_{z_1}(q_1) &= w_0 \mu_1 + w_1 (q_1 \mu_1 + (M - q_1) \mu_0), \\
\mu_{z_0}(q_0) &= w_0 \mu_0 + w_1 (q_0 \mu_0 + (M - q_0) \mu_1), \\
\Sigma_z &= w_0^2 + M w_1^2 \Sigma,
\end{aligned}
\tag{6.18}$$

and again impose the constraint $w_0^2 + M w_1^2 = 1$. The corresponding classification rule can be computed then as:

$$\begin{aligned}
d_z(Z) &= AZ' - \frac{1}{2}B, \text{ where} \\
A &= (w_0 + w_1 M(p_1 + p_0 - 1))(\mu_1 - \mu_0)' \Sigma^{-1}, \\
B &= \mu'_{z_1}(M p_1) \Sigma^{-1} \mu_{z_1}(M p_1) - \mu'_{z_1}(M p_1) \Sigma^{-1} \mu_{z_1}(M p_1) \\
&\quad + w_1^2 M(p_1(1 - p_1) - p_0(1 - p_0)) D^2(\mu_1, \mu_0).
\end{aligned}$$

This form of $d_z(Z)$ is more complicated, but remains linear in Z . Hence, $d_z(Z)$ is a univariate normal random variable and the corresponding probability of correct classification can be computed explicitly.

Other natural extensions of the proposed local fusion framework include incorporating more complicated neighborhood structures, and classification of multi-class classification problems.

6.4 Performance Evaluation

In this section, we compare the performance of the proposed LDF framework to classical LDA applied to three different types of synthesized binary textures and to a real data set of handwritten digits.

6.4.1 Classification of binary textures

There are several algorithms available in the literature for texture modeling. Here we use an algorithm that is based on partially ordered Markov models (POMMs) introduced by Davidson and Cressie in [24]. According to this algorithm, first, we have to choose the neighborhood dependency structure and corresponding conditional probability. Following the original paper [24], we define

$$U(Y_{ij}) = \{Y_{i-1,j}, Y_{i-1,j-1}, Y_{i,j-1}, Y_{i+1,j-1}\},$$

and a Bernoulli distribution for the conditional probability,

$$\mathbb{P}(Y_{ij}|U(Y_{ij})) = q^{Y_{ij}}(1-q)^{1-Y_{ij}}, \text{ where}$$

$$q = e^{T_{ij}}/(1 + e^{T_{ij}}), \text{ and}$$

$$T_{ij} = \alpha + \beta Y_{i-1,j} + \gamma Y_{i-1,j-1} + \delta Y_{i,j-1} + \epsilon Y_{i+1,j-1}.$$

Using different values for the parameters $\alpha, \beta, \gamma, \delta$, and ϵ (Figure 6.2), we generate three types of binary textures $\{Y_{ij}\}$ (Figure 6.3). Note that each binary texture is a 64×64 pixel black-and-white image with the following proportions of the number of black ($Y_{ij} = 0$) pixels π_0 to the number of white ($Y_{ij} = 1$) pixels π_1 for each type of textures:

- Type 1: $\pi_0 = 0.6, \pi_1 = 0.4$;
- Type 2: $\pi_0 = 0.52, \pi_1 = 0.48$;
- Type 3: $\pi_0 = 0.51, \pi_1 = 0.49$.

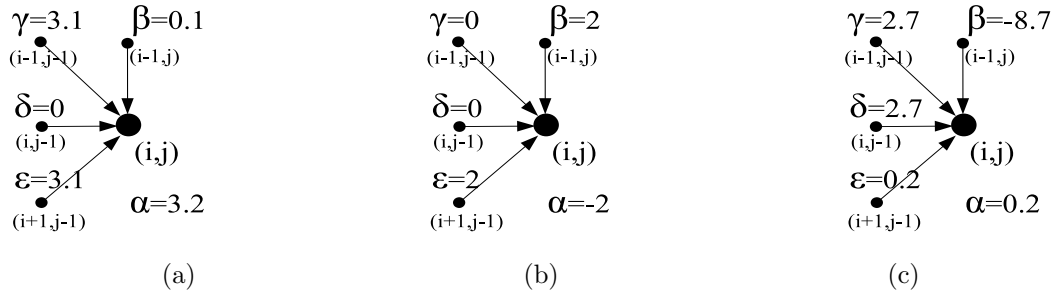


Figure 6.2: Parameters $\{\alpha, \beta, \gamma, \delta, \epsilon\}$ corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.

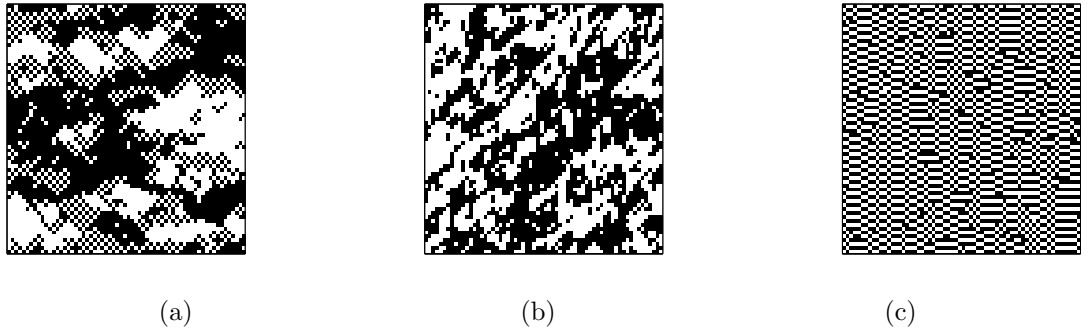


Figure 6.3: True labels Y_{ij} corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.

Given $\{Y_{ij}\}$, we simulate i.i.d. realizations of $\{X_{ij}\}$ from a normal distribution with means $\mu_0 = 0$ and $\mu_1 = 1$ for $Y_{ij} = 0$ and $Y_{ij} = 1$, respectively. We set the variance $\sigma^2 = 0.5$, so that the signal-to-noise ratio, defined as $SNR = \frac{\mu_1 - \mu_0}{\sigma^2}$, equals 2. The described procedure corresponds to a scenario when the original texture image was contaminated or distorted at a given SNR (Figure 6.4).

To recover the original texture image, or $\{Y_{ij}\}$ that in our setting are treated as true labels, we first apply LDA with true means μ_1 and μ_0 to predict $\{Y_{ij}\}$ from the

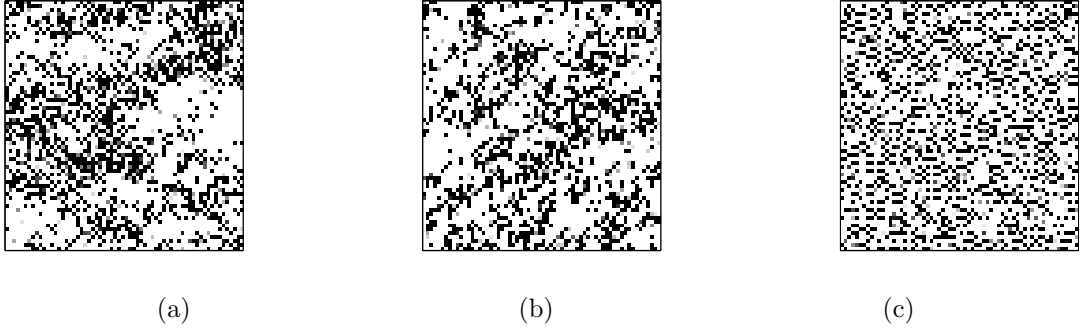


Figure 6.4: Simulated X_{ij} corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.

feature values $\{X_{ij}\}$ and then plot the predicted labels on Figure 6.5. As one can easily notice, there is still a severe distortion compared to the original textures. The corresponding rates of correct classification P_x are also relatively low (84.1%–84.4%).

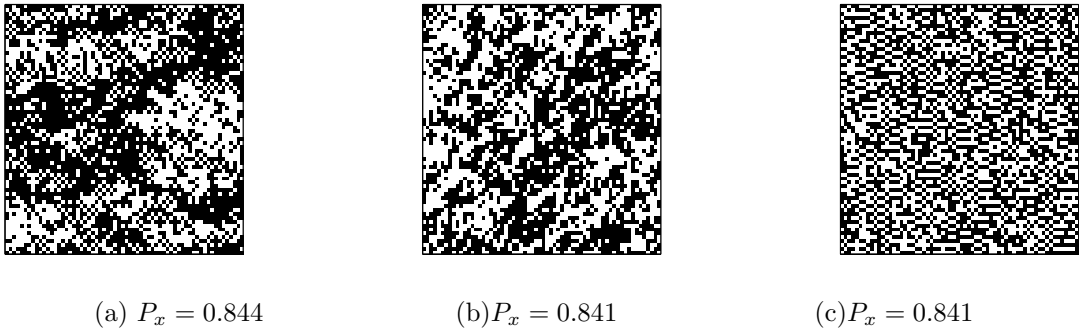
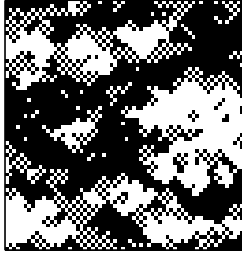


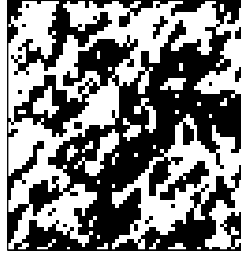
Figure 6.5: Labels predicted via LDA for binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.

Next, we apply LDF framework to obtain fused feature values $\{Z_{ij}\}$ and predict class labels $\{Y_{ij}\}$ using the proposed linear classification rule for fused data. The results are shown on Figure 6.6; the rates of correct classification P_z range from 90% (for Type 2) to 94.2% (for Type 3), which illustrates the superior performance of the algorithm. Note since the proportions of the number of black pixels π_0 to the number of white pixels π_1 are very similar for all types of textures, we expect that the rates of correct classification P_z are close to the optimal values, according to

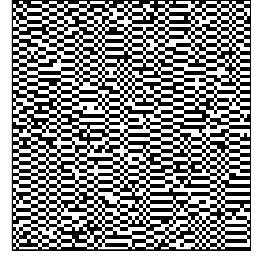
Corollary VI.6.



(a) $P_z = 0.931$



(b) $P_z = 0.90$



(c) $P_z = 0.943$

Figure 6.6: Labels predicted via LDA-LF corresponding to binary textures of (a) Type 1, (b) Type 2, and (c) Type 3.

The average difference, over 100 replications, in the rates of correct classification P_z and P_x as a function of SNR is plotted for different types of textures in Figure 6.7(a). It can be seen that the largest gains correspond to low (1) and medium (3) SNRs; for large SNR values the gains become small for all types of textures, because the relative difference in the class means becomes large and the probability of correct classification for both LDA and our fusion framework approaches one. The largest gains are obtain for Type 3 texture. This is due a highly regular pattern in the vertical direction, which is picked up by the corresponding weights in the fusion framework.

Notice that the choice of the right fusion neighborhood is in general a very important task. In the comparisons reported above a "universal" neighborhood which accounts for vertical (V), horizontal(H), and two diagonal directions (left upper corner and right lower corner(D1), and left lower corner and right upper corner(D2)) with two neighbors for each direction is used, as illustrated in Figure 6.7(b). In our setup, it represents a neighborhood structure with four layers with two objects in each one, i.e., $M_j = 2, j = 1, \dots, 4$. The corresponding probabilities of class label

agreement p_j (up to second decimal digit) and the weights w_j (up to the third decimal digit) are in Table 6.1. The results show that neighbor contributions are greater when the corresponding probability p_j is further from $1/2$, as expected.

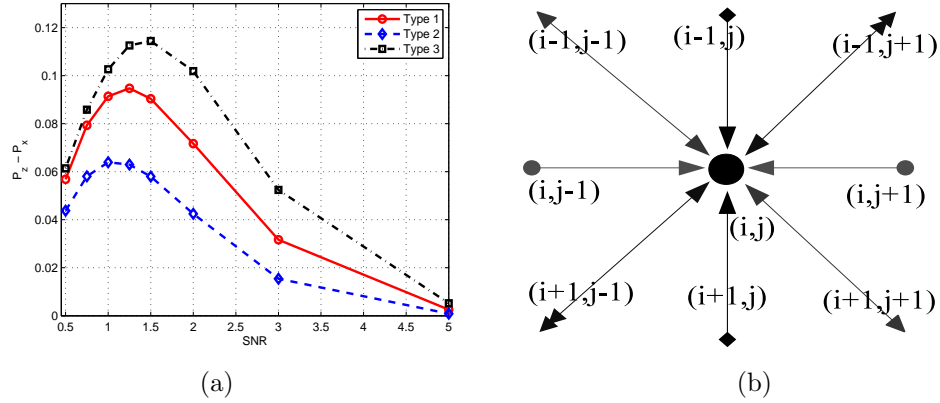


Figure 6.7: (a) Estimated average difference ($P_z - P_x$) as a function of SNR . (b) Universal neighborhood 4×2 .

	LDA	V	H	D1	D2
	Type 1				
p	1	0.6	0.6	0.83	0.83
w	0.974	0.029	0.028	0.108	0.108
	Type 2				
p	1	0.74	0.68	0.73	0.61
w	0.986	0.071	0.052	0.069	0.031
	Type 3				
p	1	0.03	0.52	0.48	0.49
w	0.947	-0.226	0.004	-0.004	-0.002

Table 6.1: Probabilities of the within direction class similarity p_j and optimal weights w_j .

In Table 6.2, for each texture type we compare the probabilities of having the same labels within the neighborhood p and the average rate of correct classification P_z for several types of the neighborhood structures. Note that the first column represents the rate of correct classification obtained via LDA, i.e., P_x . The results in Table 6.2 suggest using the neighborhood structures with layers where the probabilities of having the same class label as the central object deviate significantly from 0.5.







	LDA						
M	0	2	2	2,2	4	4,4	8
Type 1							
p	0.5	0.6	0.6	0.6,0.6	0.6	0.6, 0.83	0.72
P_z	0.844	0.849	0.849	0.854	0.854	0.931	0.913
Type 2							
p	0.5	0.74	0.68	0.74,0.68	0.71	0.71, 0.67	0.69
P_z	0.841	0.869	0.857	0.882	0.880	0.900	0.899
Type 3							
p	0.5	0.03	0.52	0.03,0.52	0.27	0.27,0.49	0.38
P_z	0.841	0.943	0.842	0.943	0.885	0.885	0.866

Table 6.2: Probabilities of the within neighborhood class similarity p and estimated probabilities of correct classification P_z .

6.4.2 Handwritten Digit Recognition

Next, we apply the proposed fusion framework to a real data set that contains normalized handwritten digit images: 852 images of the digit four (4), and 821 images of the digit nine (9). Each observation corresponds to a 16×16 gray-scale image, i.e., the feature vector is 256-dimensional.

The results summarized in Table 6.3 contain training classification rates and test classification rates for LDA and LDF using four neighbors ($M = 4$). To test the performance of the method with different sample sizes and particularly in the "large p small n " scenario, we perform a random split of the data, keep 50% (837 images), 20% (335 images) or 10% (168 images) for training. and the rest for testing. The correct classification rates are then averaged over 100 such random splits. On each split, dimensions corresponding to the background for all images and dimensions where the difference in class means did not exceed 0.1 were excluded from consideration. On average, about a half of all dimensions were used for analysis.

Table 6.3 shows that the main gains occur when $p \gg n$, which is the case of most interest. When only 10% of the data is used for training, LDF provides about 12% improvement in the correct classification rate.

% training		50%	20%	10%
LDA	train	0.9926	0.9986	1.0000
	test	0.9743	0.9553	0.7578
LDF	train	0.9926	0.9960	0.9869
	test	0.9794	0.9712	0.8793

Table 6.3: Training and test rates of correct classification for handwritten data.

The work presented in this chapter is still in progress, but the obtained results are very promising, especially for spatially correlated data.

CHAPTER VII

Conclusions and Future Research

The main contributions of our work are the application of wireless sensor networks to detection, localization, and tracking of single and multiple targets. In Chapter III, we proposed a new decision fusion algorithm for target detection which makes no assumptions about the signal model, and therefore can be used for detection of single or multiple targets. Numerical results show that the proposed scheme achieves significantly higher detection rates under a variety of settings, including sensor deployment mechanisms, signal-to-noise ratio, target size, etc. In Chapter IV, we examined the problem of estimating the location and signal amplitude of a single target. We developed several localization algorithms based on corrected decisions that provide highly accurate estimates, and in the presence of high levels of noise consistently outperform the energy based ML algorithm. Through the sequence of simulations we also showed that LVDF based methods are robust to signal model and noise distribution misspecification. Further, in Chapter V we introduced an LVDF based framework for localization and tracking multiple targets. Our multi-target tracking approach is based on a penalized maximum likelihood framework, and allows for sensor failures, targets appearing and disappearing over time, and complex intersecting target trajectories. We show that the proposed framework provides the

most robust performance in noisy environments, and gives good tracking results in applications – an experiment involving tracking people (NEST) and a project tracking zebras (ZebraNet).

In Chapter II, we addressed the problem of an efficient wireless sensor network design under random deployment. We proposed a general design framework which can be extended to include a performance measure of the ultimate network task (such as accuracy of target localization or field estimation) into the constraints. The precise nature of the optimization problem will be determined by the application, but the general principle of optimizing the network cost subject to operational constraints is applicable to a wide variety of WSN problems.

One example of an interesting WSN problem for future work is to develop an adaptive sensor scheduling algorithm which would control how often different sensors acquire measurements over time. In applications like herd monitoring, one could use a strategy which adjusts sensors sampling rate according to the average speed of animals, such as turning off most sensors during the night when the animals are not moving. Similar strategies can be considered in surveillance applications. In WSN design, our cost-efficient framework can be extended to networks with several types of sensors (heterogeneous networks) and to networks where each sensor can collect measurements about more than one environmental characteristic.

Our linear data fusion classification framework developed in Chapter VI opens a whole new direction of research in classification of correlated observations. It allows naturally incorporating spatial and time varying components into classification, and also fusing data from multiple sources with varying degrees of reliability. This framework can be extended to multi-class problems and then to more complicated underlying classification algorithms.

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] A.A. Abdel-Samad and A.H. Tewfik. Search strategies for radar target localization. In *Proceedings of International Conference on Image Processing*, volume 3, pages 862–866, October 1999.
- [2] K. Abend, T. Harley, and L. Kanal. Classification of binary random patterns. *IEEE Transactions on Information Theory*, 11(4):538 – 544, 1965.
- [3] J. Ahn and B. Krishnamachari. Modeling search costs in wireless sensor networks. Technical report, University of Southern California, Department of Electrical Engineering-Systems, 2007.
- [4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.
- [5] I.F. Akyildiz, D. Pompili, and T. Melodia. Underwater acoustic sensor networks:research challenges. *Ad Hoc Networks Journal (Elsevier)*, 18:257–279, March 2005.
- [6] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.
- [7] S.A. Aldosari and J.M.F. Moura. Saddlepoint approximation for sensor network optimization. In *The Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages iv/741–iv/744, March 2005.
- [8] J. Bahi, A. Makhoul, and A. Mostefaoui. Localization and coverage for high density sensor networks. In *Proceedings of the 5th Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 295–300, 2007.
- [9] J. Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, September 1975.
- [10] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society: Series B*, 48(3):259–302, 1986.
- [11] Christian Bettstetter. On the minimum node degree and connectivity of a wireless multi-hop network. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 80–91, 2002.
- [12] D Blatt and A.O.III Hero. Energy-based sensor network source localization via projection onto convex sets. *IEEE Transactions on Signal Processing*, 54(9):3614–3619, September 2006.
- [13] R. Cardell-Oliver, K. Smettem, M. Kranz, and K. Mayer. A reactive soil moisture sensor network:design and field evaluation. *International journal of Distributed Sensor Networks*, 1(2):149–163, 2005.
- [14] Z. Chair and P.K. Varshney. Optimal data fusion in multiple sensor detection systems. *IEEE Transactions on Aerospace and Electronic Systems*, 22(1):98–101, 1986.

- [15] B. Chen and P.K. Varshney. A Bayesian sampling approach to decision fusion using hierarchical models. *IEEE Transactions on Signal Processing*, 50(8):1809–1818, August 2002.
- [16] 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*, 50(8):1843–1854, August 2002.
- [17] Phoebus Chen, Songhwai Oh, Michael Manzo, B. Sinopoli, C. Sharp, K. Whitehouse, G. Tolle, J. Jeong, P. Dutta, J. Hui, S. Shaffert, S. Kim, J. Taneja, B. Zhu, T. Roosta, M. Howard, D. Culler, and S. Sastry. Experiments in instrumenting wireless sensor networks for real-time surveillance. In *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, pages 3128 – 3133, May 2006.
- [18] T. Chen, W. Liao, M. Huang, and H. Tsai. Dynamic object tracking in wireless sensor networks. In *Proceedings of 13th IEEE International Conference on Networks, 2005*, volume 1, page 6, November 2005.
- [19] W.-P. Chen, J.C. Hou, and Lui Sha. Dynamic clustering for acoustic target tracking in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 3(3):258 – 271, July–August 2004.
- [20] T. Clouqueur, V. Phipatanasuphorn, K.K. Saluja, and P. Ramanathan. Sensor deployment strategy for detection of targets traversing a region. *Mobile Networks and Applications*, 28(8):453–461, 2003.
- [21] T. Clouqueur, P. Ramanathan, K.K. Saluja, and K.C. Wang. Value-fusion versus decision-fusion for fault-tolerance in collaborative target detection in sensor networks. In *Proceedings of 4th Annual Conference on Information Fusion*, pages TuC2/25–TuC2/30, 2001.
- [22] Y. Le Cun, B. Boser, J. S. Denker, R. E. Howard, W. Hubbard, L. D. Jackel, and D. Henderson. Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems 2*, pages 396–404, 1990.
- [23] R. A. Dara, M. S. Kamel, and N. Wanas. Data dependency in multiple classifier systems. *Pattern Recognition*, 42(7):1260–1273, July 2009.
- [24] J.L. Davidson, Noel Cressie, and X. Hua. Texture synthesis and pattern recognition for partially ordered Markov models. *Pattern Recognition*, 32(9):1475–1505, September 1999.
- [25] J. DeBonet and P. Viola. A non-parametric multi-scale statistical model for natural images. *Advances in Neural Information Processing Systems*, 9, 1997.
- [26] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39(1):1–38, 1977.
- [27] A. W. Van der Vaart. *Asymptotic Statistics*. Cambridge Univ. Press, 1998.
- [28] P. Doukhan. *Mixing: Properties and Examples*. Springer–Verlag, 1994.
- [29] M. Duarte and Y.H. Hu. Distance-based decision fusion in a distributed wireless sensor network. *Telecommunication Systems*, 26(2-4):339–350, June 2004.
- [30] R.O. Duda, P.E. Hart, and D.H. Stork. *Pattern Classification*. Wiley, 2000.
- [31] R.J. Efron, B. Tibshirani. *An Introduction to the Bootstrap*. CRC Press, 1994.
- [32] Kie B. Eom. Synthesis of color textures for multimedia applications. *Multimedia Tools Applications*, 12(1):81–98, 2000.
- [33] E.B. Ermis and V. Saligrama. Detection and localization in sensor networks using distributed FDR. In *Proceedings of Conference on Information Sciences and Systems*, 2006.

- [34] D. Estrin. Wireless sensing systems: from eco-systems to human-systems. In *Feedback and Dynamics in Nature Workshop held at the Grace Hopper Celebration of Women in Computing Conference*, 2006.
- [35] Ronald Aylmer Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.
- [36] Eibe Frank, Mark Hall, and Bernhard Pfahringer. Locally weighted naive bayes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 249–256. Morgan Kaufmann, 2003.
- [37] J.H. Friedman. Regularized discriminant analysis. *Journal of the American Statistical Association*, 84:165–175, 1989.
- [38] A. Gordon. *Classification*. Chapman and Hall, 1980.
- [39] P. Gupta and P.R. Kumar. Critical power for asymptotic connectivity in wireless networks. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 1, 1998.
- [40] X. Guyon. *Random Fields on a Network: Modeling, Statistics, and Applications*. Springer-Verlag, New York, 1995.
- [41] Li Haihao, Liu Mei, Shen Yi, and Qiao Deli. Research on task allocation technique for multi-target tracking in wireless sensor network. In *Proceedings of International Conference on Mechatronics and Automation*, pages 360 – 365, August 2007.
- [42] P. Hall. *The Theory of Coverage Processes*. New York: Wiley, 1988.
- [43] Trevor Hastie, Robert Tibshirani, and Andreas Buja. Flexible discriminant analysis by optimal scoring. *Journal of the American Statistical Association*, 89:1255–1270, 1994.
- [44] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Springer Series in Statistics)*. Springer New York, 2001.
- [45] M. Hefeeda and M. Bagheri. Efficient k-coverage algorithms for wireless sensor networks. Technical report, School of Computing Science, Simon Fraser University, January 2007.
- [46] Wen Hu, Van Nghia Tran, Nirupama Bulusu, Chun Tung Chou, Sanjay Jha, and Andrew Taylor. The design and evaluation of a hybrid sensor network for cane-toad monitoring. In *Proceedings of the 4th international symposium on Information processing in sensor networks (IPSN 2005)*, page 71. IEEE Press, 2005.
- [47] C.-F. Huang and Y.-C. Tseng. The coverage problem in wireless sensor networks. *Mobile Networks and Applications*, 10(4):519 – 528, August 2005.
- [48] Lewis F. Jardme. *Digital Image Processing as an Aid to Camouflage Design and Assessment*. PhD thesis, School of Electrical Engineering and Science, Royal Military College of Science, Shrivvenham, 1989.
- [49] X. Ji and H. Zha. Sensor positioning in wireless ad-hoc sensor networks with multidimensional scaling. In *Infocom*, 2004.
- [50] M. Kam, Q. Zhu, and W.S. Gray. Optimal data fusion of correlated local decisions in multiple sensor detection systems. *IEEE Transactions on Aerospace and Electronic Systems*, 28(3):916–920, 1992.
- [51] S. Kamath, E. Meisner, and V. Isler. Triangulation based multi target tracking with mobile sensor networks. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3283–3288, April 2007.

- [52] L.M. Kaplan, Q. Le, and P. Molnar. Maximum-likelihood methods for bearings-only target localization. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 3001–3016, May 2001.
- [53] N. Katenka, E. Levina, and G. Michailidis. A cost-efficient approach to wireless sensor network design. Technical report, Technical report 474, Department of Statistics, University of Michigan., November 2007. Under revision.
- [54] N. Katenka, E. Levina, and G. Michailidis. Local vote decision fusion for target detection in wireless sensor networks. *IEEE Transactions on Signal Processing*, 56(1):329–338, January 2008.
- [55] N. Katenka, E. Levina, and G. Michailidis. Robust target localization from binary decisions in wireless sensor networks. *Technometrics: A journal of statistics for the physical, chemical and engineering sciences*, 50(4):448–461, November 2008.
- [56] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Wireless sensor networks for structural health monitoring. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 427–428, New York, NY, USA, 2006. ACM Press.
- [57] L. Klein. A Boolean algebra approach to multiple sensor voting fusion. *IEEE Transactions on Aerospace and Electronic Systems*, 29(2):317–327, April 1993.
- [58] C. Kreucher, K. Kastella, and A. Hero. Multitarget tracking using the joint multitarget probability density. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1396–1414, October 2005.
- [59] L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, and M. Yarvis. Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the north sea. In *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pages 64–75, New York, NY, USA, 2005. ACM Press.
- [60] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.
- [61] S. Kumar, T.H. Lai, and J. Balogh. On k-Coverage in a mostly sleeping sensor network. In *Proceedings of the 10th annual international conference on Mobile Computing and Networking*, pages 144–158, 2004.
- [62] A. Leone and C. Distanto. Shadow detection for moving objects based on texture analysis. *Pattern Recognition*, 40(4):1222–1233, April 2007.
- [63] E. Levina and P.J. Bickel. Texture synthesis and non-parametric resampling of random fields. *Annals of Statistics*, 34(4):1751–1773, 2006.
- [64] D. Li, K.D. Wong, Y.H. Hu, and A.M. Sayeed. Detection, classification, and tracking of targets. *IEEE Signal Processing Magazine*, 19(3):17–29, March 2002.
- [65] Q. Li, J. Aslam, and D. Rus. Distributed energy-conserving routing protocols for sensor network. In *Proceedings on 37th Hawaii International Conference on System Science*, January 2003.
- [66] X. Li and F. Fritz Prinz. Embedded fiber bragg grating sensors in polymer structures fabricated by layered manufacturing. *Journal of Manufacturing Process*, 5(1):78–86, 2003.
- [67] Yao Li and P.M. Djuric. Particle filtering for target tracking with mobile sensors. In *Proceedings of 2007 IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages II–1101 – II–1104, April 2007.

- [68] B. Liu and D. Towsley. A study of the coverage of large-scale sensor networks. In *Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, October 2004.
- [69] C. Liu, K. Wu, Y. Xiao, and B. Sun. Random coverage with guaranteed connectivity: joint scheduling for wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 17(6):562–575, June 2006.
- [70] J. Liu, M. Chu, and J.E. Reich. Multitarget tracking in distributed sensor networks. *IEEE Signal Processing Magazine*, 24(3):36–46, May 2007.
- [71] T. Liu, C. Sadler, P. Zhang, and M. Martonosi. Implementing software on resource-constrained mobile sensors: Experiences with impala and zebranet. In *Proceedings of 2nd International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 256–269, June 2004.
- [72] M. Loog and D. de Ridder. Local discriminant analysis. In *Proceedings on 18th International Conference on Pattern Recognition (ICPR 2006)*, volume 3, pages 328–331, 2006.
- [73] Hui Ma and Brian W.-H. Ng. Distributive jpdaf for multi-target tracking in wireless sensor networks. In *Proceedings of 2006 IEEE Region 10 Conference, TENCON 2006*, pages 1–4, November 2006.
- [74] M. Magdon-Ismael, F. Sivrikaya, and B. Yener. Joint problem of power optimal connectivity and coverage in wireless sensor networks. *Wireless Networks*, 13(4):537 – 550, August 2007.
- [75] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97, New York, NY, USA, 2002. ACM Press.
- [76] Baljeet Malhotra and Alex Aravind. Path-adaptive on-site tracking in wireless sensor networks. *IEICE - Transactions on Information and Systems*, E89-D(2):536–545, 2006.
- [77] C. Meesookho, U. Mitra, and S. Narayanan. On energy-based acoustic source localization for sensor networks. *IEEE Transactions on Signal Processing*, 56(1):365 – 377, January 2008.
- [78] C. Meesookho and S. Narayanan. Distributed range difference based target localization in sensor network. In *Proceedings of the 39th Asilomar Conference on Signals, Systems and Computers*, pages 205–209, November 2005.
- [79] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proceedings of the 12th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1380–1387, 2001.
- [80] V. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff. A minimum cost heterogeneous sensor network with a lifetime constraint. *IEEE Transactions on Mobile Computing*, 4(1):4–15, January–February 2005.
- [81] W. Mo, D. Qiao, and Z. Wang. Mostly sleeping wireless sensor networks: Connectivity, k-Coverage and α -Lifetime. In *Proceedings of the 43rd Annual Allerton Conference on Communication, Control, and Computing*, September 2005.
- [82] R. Niu and P.K. Varshney. Target location estimation in wireless sensor networks using binary data. In *Conference on Information Sciences and Systems*, March 2004.
- [83] R. Niu and P.K. Varshney. Performance evaluation of decision fusion in wireless sensor networks. In *Proceedings of 40th Annual Conf. Info. Sciences Systems*, March 2006.

- [84] R. Niu, P.K. Varshney, and Q. Cheng. Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks. *International journal on Information Fusion*, 53, 2004.
- [85] M. Noel, P. Joshi, and T. Jannett. Improved maximum likelihood estimation of target position in wireless sensor networks using particle swarm optimization. In *Proceedings of 3rd International Conference on Information Technology: New Generations*, April 2006.
- [86] S. Oh, P. Chen, M. Manzo, and S. Sastry. Instrumenting wireless sensor networks for real-time surveillance. In *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, May 2006.
- [87] P. Padhy, K. Martinez, A. Riddoch, H. L. R. Ong, and J. K. Hart. Glacial environment monitoring using sensor networks. In *Proceedings of Real-World Wireless Sensor Networks*. ACM Press, 2005.
- [88] M. Penrose. *Random Geometric Graphs*. Oxford: University Press, 2003.
- [89] M.D. Penrose and J.E. Yukich. Central limit theorems for some graphs in computational geometry. *Annals of Applied Probability*, 11:1005–1041, 2001.
- [90] M.D. Penrose and J.E. Yukich. Limit theory for random sequential packing and deposition. *Annals of Applied Probability*, 12:272–301, 2002.
- [91] D. K. Pickard. A curious binary lattice process. *Journal of Applied Probability*, 14:717 – 731, 1977.
- [92] D. K. Pickard. Unilateral ising models. *Journal of Advances in Applied Probability*, 10:58 – 64, 1978.
- [93] D. K. Pickard. Unilateral markov fields. *Journal of Advances in Applied Probability*, 12:655 – 671, 1980.
- [94] J. Portilla, R. Navarro, O. Nestares, and A. Taberero. Texture synthesis-by-analysis based on a multiscale early-vision model. *Optical Engineering*, 35(8), 1996.
- [95] W. Qian and D. M. Titterton. Multidimensional markov chain models for image textures. *Journal of the Royal Statistical Society: Series B*, 53:661–674, 1991.
- [96] W. Qian and D. M. Titterton. Pixel labelling for three-dimensional scenes based on markov mesh models. *Signal Processing*, 22(3):313–328, 1991.
- [97] S. Ren, Q. Li, H. Wang, X. Chen, and X. Zhang. Design and analysis of sensing scheduling algorithms under partial coverage for object detection in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 18(3):334–350, March 2007.
- [98] S. Shakkottai, R. Srikant, and N. Shroff. Unreliable sensor grids: Coverage, connectivity, and diameter. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1073–1083, 2003.
- [99] X. Sheng and Y.H. Hu. Energy based acoustic source localization. In *Proceedings of 3rd International Workshop on Information Processing in Sensor Networks*, volume 2634, pages 286–300, April 2003.
- [100] J.-P. Sheu and H.-F. Lin. Probabilistic coverage preserving protocol with energy efficiency in wireless sensor networks. In *Proceedings of the Wireless Communications and Networking Conference*, pages 2631–2636, 2007.
- [101] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.

- [102] R. Steele, C. Secombe, and W. Brookes. Using wireless sensor networks for aged care: the patient's perspective. In *Proceedings of Pervasive Health Conference*, pages 1–10, 2006.
- [103] T. Sun, L.J. Chen, C.C. Han, and M. Gerla. Reliable sensor networks for planet exploration. In *Proceedings of IEEE International Conference on Networking, Sensing and Control*, 2005.
- [104] Y. Sung, L. Tong, and A. Swami. Asymptotic locally optimal detector for large scale sensor network under the poisson regime. *IEEE Transactions on Signal Processing*, 53(6):2005–2017, June 2005.
- [105] A.G. Tartakovsky and X.-R. Li. Sequential testing of multiple hypotheses in distributed systems. In *Proceedings of 3rd International Conference on Information Fusion*, volume 2, pages 10–13, July 2000.
- [106] A.G. Tartakovsky and V. Veeravalli. Change point detection in multichannel and distributed systems with applications. *Applications of Sequential Methodologies*, pages 339–370, 2004.
- [107] Jing Teng, H. Snoussi, and C. Richard. Prediction-based proactive cluster target tracking protocol for binary sensor networks. In *Proceedings of 2006 IEEE International Symposium on Signal Processing and Information Technology*, pages 234 – 239, December 2007.
- [108] R. Tenney and N. Sandell. Detection with distributed sensors. *IEEE Transactions on Aerospace and Electronic Systems*, 17(3):501–510, 1981.
- [109] D. Tian and N. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *Proceedings of the ACM Workshop on Wireless Sensor Networks and Applications*, October 2002.
- [110] H. Varian. *Intermediate Microeconomics*. W.W. Norton and Co., 1999.
- [111] Mahesh Vemula, Monica F Bugallo, and Petar M. Djuric. Particle filtering-based target tracking in binary sensor networks using adaptive thresholds. In *Proceedings of 2nd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pages 17 – 20, December 2007.
- [112] T. Vercauteren, Dong Guo, and Xiaodong Wang. Joint multiple target tracking and classification in collaborative sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):714 – 723, April 2005.
- [113] R. Viswanathan and P.K. Varshney. Distributed detection with multiple sensors: Part I-Fundamentals. *Proceedings IEEE*, 85(1):54–63, January 1997.
- [114] P.-J. Wan and C.-W. Yi. Asymptotic critical transmission radius and critical neighbor number for k-Connectivity in wireless ad hoc networks. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 1–8, 2004.
- [115] Bin Wang and Harry Zhang. Probability based metrics for locally weighted naive bayes. In *Proceedings of the 20th conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence (CAI 2007)*, pages 180–191, Berlin, Heidelberg, 2007. Springer-Verlag.
- [116] H. Wang, K. Yao, and D. Estrin. Information-theoretic approaches for sensor selection and placement in sensor networks for target localization and tracking. *Journal of Communication and Networks*, 7(4):438–448, December 2005.
- [117] W. Wang, V. Srinivasan, B. Wang, and K. Chua. Coverage for target localization in wireless sensor networks. In *Proceedings on Information Processing in Sensor Networks*, pages 118–125, April 2006.

- [118] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill. Integrated coverage and connectivity configuration in wireless sensor networks. In *Proceedings of the 1st International Conference on Embedded networked sensor systems*, pages 28–39, 2003.
- [119] Xuezhi Wang and B. Moran. Multi-target tracking using virtual measurement of binary sensor networks. In *Proceedings of 9th International Conference on Information Fusion, ICIF '06*, pages 1 – 8, July 2006.
- [120] Y. Wang, S. Jain, M. Martonosi, and K. Fall. Erasure coding based routing for opportunistic networks. In *Proceedings of the ACM SIGCOMM Workshop on Delay Tolerant Networking and related topics (WDTN)*, August 2005.
- [121] Y.-C. Wang, C.-C. Hu, and Y.-C Tseng. Efficient deployment algorithms for ensuring coverage and connectivity of wireless sensor networks. In *Proceedings of the 1st International Conference on Wireless Internet*, pages 114–121, July 2005.
- [122] Goerges Winkenbach and David H. Salesin. Rendering parametric surfaces in pen and ink. In *Proceedings of the 23rd International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 469–476, August 1996.
- [123] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, New York, NY, USA, November 2004. ACM Press.
- [124] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 13–24, 2004.
- [125] L. Yang, C. Feng, J.V. Rozenblit, and H. Qiao. Adaptive tracking in distributed wireless sensor networks. In *Proceedings of 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems*, June 2006.
- [126] P. Zhang, C. Sadler, S. Lyon, and M. Martonosi. Hardware design experiences in zebranet. In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2004.
- [127] M. Zhu, S. Ding, R. Brooks, Q. Wu, N. Rao, and S. Lyengar. Fusion of threshold rules for target detection in sensor networks. *ACM Transactions on Sensors Networks*, 2005. Submitted.
- [128] Y. Zou and K. Chakrabarty. Target localization based on energy considerations in distributed sensor networks. *IEEE Transactions on Signal Processing*, 5:51–58, February 2003.