Sensor Management for Large-Scale Multisensor-Multitarget Tracking

SENSOR MANAGEMENT FOR LARGE-SCALE MULTISENSOR-MULTITARGET TRACKING

By

R. THARMARASA, B.Sc.Eng., M.A.Sc.

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree of
Doctor of Philosophy

McMaster University

DOCTOR OF PHILOSOPHY (2008) MCMASTER UNIVERSITY

(Electrical and Computer Engineering) Hamilton, Ontario, Canada

TITLE: **Sensor Management for Large-Scale Multisensor-Multitarget Tracking**

AUTHOR: R. Tharmarasa

B.Sc.Eng.

University of Moratuwa

Sri Lanka, 2001

M.A.Sc.

McMaster University

Canada, 2003

SUPERVISOR: Dr. T. Kirubarajan

NUMBER OF PAGES: xix, 164

Abstract

In this thesis we consider the problem of managing an array of sensors in order to track multiple targets in the presence of clutter in centralized, distributed and decentralized architectures. As a result of recent technological advances, a large number of sensors can be deployed and used for multitarget tracking purposes. Even though a large number of sensors are available, due to frequency, power and other physical limitations, only a few of them can be used at any one time. The problem is then to select sensor subsets that should be used by fusion centers at each measurement time step in order to optimize the tracking performance subject to their operational constraints.

In general, sensor management is performed based on the predicted tracking performance at the future time steps. In this thesis, the Posterior Cramér-Rao lower bound (PCRLB), which provides a measure of the optimal achievable accuracy of target state estimation, is used as the performance measure. We derive the multitarget PCRLB and show the existence of a multitarget information reduction matrix (IRM), which can be calculated off-line in most cases. First, the sensor subset selection problem for centralized architecture is considered for two different scenarios: (i) fixed and known number of targets; (ii) varying number of targets. Then, in the

distributed architecture, in addition to assigning sensor subsets to local fusion centers (LFCs), the transmission frequencies and powers of active sensors need to be assigned. In this thesis, we assume that the transmission power of the sensors will be software controllable within certain lower and upper limits. Finally, we consider the decentralized architecture in which there is no central fusion center (CFC), each fusion center (FC) communicates only with the neighboring FCs, and communications are restricted. In this case, each FC has to decide which sensors should to be used by itself at each measurement time step by considering which sensors may be used by neighboring FCs.

We give the optimal formulations for all of the above problems. Finding the optimal solutions to the above problems in real time is very hard in large scale scenarios. We present algorithms to find suboptimal solutions in real time. Simulation results illustrate the performance of the algorithms, both in terms of their real-time capability for large scale problems and the resulting estimation accuracy.

To my parents, who sacrificed so much for my well-being,

and my wife, who supported and encouraged me during my studies

Acknowledgements

I would like to thank all my friends, family members and others who encouraged and supported me during the research leading to this thesis.

First, I would like to express my sincere gratitude to my supervisor, Dr. T. Kirubarajan, for giving me an opportunity to work on a very interesting topic and for providing continuous guidance, advice and support throughout the course of this research work. I also thank Dr. A. Sinha, Dr. M. Hernandez and Dr. J. Peng, for their valuable comments and suggestions. I would also like to thank my committee members, Dr. T. Davidson and Dr. S. Sirouspour, for their feedback on the ideas covered in this thesis.

I thank my fellow graduate students in the ECE department, in particular all my friends in the ETFLab, for their help and support. Further, my sincere thanks are due to the ECE department staff, especially Cheryl Gies and Helen Jachna.

Finally, I would like to thank my family for supporting and encouraging me to pursue this degree.

Contents

List of Tables

List of Figures

Nomenclature

Acronyms

ACO	ant colony optimization
ASW	anti-submarine warfare
CFC	central fusion center
EKF	extended Kalman filter
FC	fusion center
FIM	Fisher information matrix
GA	genetic algorithm
ILS	iterated least squares
IRM	information reduction matrix
JPDA	joint probabilistic data association
LFC	local fusion center
MHT	multiple hypothesis tracking
NN	nearest neighbor
PCRLB	posterior Cramér-Rao lower bound
PDA	probabilistic data association
pdf	probability density function

RMSE	root mean square error
SIR	sampling importance resampling
SN	strongest neighbor
SNR	signal-to-noise ratio
UAV	unmanned aerial vehicle

Mathematical notations

a_k	association vector
$B_{\max}$	initial sensor energy
B^i_{rem}	remaining energy of sensor i
C^f_i	cost of i-th sensor of FC f
C_l	lower limit of sensor change interval
C_u	upper limit of sensor change interval
$\mathbb{E}(\cdot)$	expectation operator
f_k	nonlinear state transition model at time k
F	number of frequency channels
F_k	state transition matrix at time k
$\hat{F}_k$	Jacobian of f_k
g	gate size
h_k	nonlinear measurement model at time k
H_k	linear measurement matrix at time k
J	Fisher information matrix
J_X	prior information matrix
J_z	measurement information matrix
L_u	tolerable estimation uncertainty
M	number of fusion centers
M_L	tolerable missing probability
n	number of active sensors
n_j	number of sensors that can be handled by j-th FC
N	number of available sensors
N_f	number of available sensors for FC f

$\mathcal{N}$	Gaussian distribution
p_l	lower limit of transmitting power
p_u	upper limit of transmitting power
P	covariance of estimate
P_D	probability of detection
P_{new}	number of particles representing new target states
P_t	transmitting power
Q	information reduction matrix
r	dimensionality of state vector
S	indicator function of sensor status
T	number of targets
T_L	sensor lead time
T_N	maximum number of new targets
T_s	sampling time interval
$\mathcal{U}[\cdot]$	uniform distribution
V	surveillance region's hyper-volume
V_L	tolerable variance of the estimation error
$\hat{x}$	estimated target state
x_k	target state at time k
x_k^t	target state of t-th target at time k
X_k	stacked target states of multiple targets at time k
y_k	state information vector at time k
Y_k	information matrix at time k
z_k	measurement vector at time k
$z_k(j,i)$	j-th measurement at the i-th sensor at time k

$z_k(i)$	measurements of i-th sensor at time k
Z_k	sequence of measurements up to time k
$\delta(\cdot)$	Dirac delta function
Γ_k	covariance matrix of process noise at time k
λ	spatial density of the false alarms
λ_b	spatial density of the new targets
μ_{FA}	probability mass function of number of false alarms
μ_{NT}	probability mass function of number of new targets
ν_k	process noise at time k
ω_k	measurement noise at time k
σ	standard deviation of measurement error
Σ_k	covariance matrix of measurement error at time k
υ	false alarm distribution
$[\cdot]'$	transpose
$[\cdot]_t$	t-th block diagonal matrix
Δ_α^β	second-order partial derivative operator
∂	partial derivative operator

Chapter 1

Introduction

1.1 Motivation and Contribution of the Thesis

1.1.1 Sensor management

Sensor management has become a crucial part of tracking because of technological advances in recent years. The use of a large number of sensors, which can be deployed all over the surveillance region, in tracking applications has become feasible because of the availability of cheap sensors (e.g., a large number of sonobuoys used in anti-submarine warfare (ASW)). In the context of target tracking, major issues in sensor management are optimal path selection [30, 62, 68], optimal sensor placement [32, 31, 45, 55] and optimal sensor selection [38, 46, 49, 70].

In optimal path selection, the objective is to find the optimal and safe (if there is any threat) path of a moving sensor, such as unmanned aerial vehicle (UAV), under the given constraints, such as, for example, maximum velocity and turn rate, to monitor the surveillance region [30]. In the optimal sensor placement, the objective is to decide where, when and how many sensors must be placed such that the target

state estimation errors remain below a certain threshold level [32]. In the optimal sensor selection, we have to decide which of the already deployed sensors must be used at each measurement time step in order to maximize the tracking performance [13]. In this thesis, we consider the problem of optimal sensor selection in centralized, distributed and decentralized tracking systems for multitarget tracking purposes.

In the literature, sensor subset selection problems are considered under limited cases. In most of the papers, only single target tracking or fixed and well separated targets are considered [6, 38, 46, 59, 81], while some others considered only the area coverage problem [7, 18, 34], where objective is to cover each point in the surveillance region by certain number of sensors. Note that in the area coverage problem, only the detection probabilities of the targets are considered, not the tracking accuracies. In addition, only small scale problems, where complete enumeration is possible to find the optimal solutions, are considered in many of the literature [16, 38, 49]. In this thesis, we consider multiple, possibly time varying, number of targets, which may be closely spaced, in large scale problems. We give the mathematical formulations, which are missing in most papers, of sensor selection problems, and the solution techniques to find suboptimal solutions in real time.

1.1.2 Posterior Cramér-Rao lower bound

In general, sensor management decisions, such as deploying new sensors or activating existing ones must be made with a lead time that is needed to carry out the necessary operations. Hence, sensors must be managed in order to control the estimated/predicted future performance of the trackers. The Posterior Cramér-Rao lower bound (PCRLB) gives a measure of the achievable optimum performance and, importantly, this bound can be calculated predictively [32, 72, 73, 83]. Furthermore,

the PCRLB is independent of the filtering algorithm employed, and is therefore not constrained by the idiosyncrasies of any particular filtering methodology. Hence, in this thesis we use the PCRLB as the measure of tracking performance.

Only recently have expressions for multitarget PCRLBs been presented [42]. In [42] the PCRLBs were given under three different measurement/association assumptions, and the necessary simulation techniques are computationally expensive. In this thesis we show that, in the case of unknown associations and a maximum of one target generated measurement per target per time step, the measurement origin uncertainty can be expressed as an information reduction matrix (IRM) [33, 83]. The calculation of this IRM is computationally more expensive than calculation of the single target IRM. However, we present some approximations that help to reduce the computational load. In addition, the calculation time of the PCRLB can be further reduced by calculating the measurement contributions (to the PCRLB) of each sensor in parallel.

In the single target case, the IRM depends on the measurement noise covariance, false alarm rate, field of view of the sensor and probability of detection. If all of the dependent variables are constant, the IRM can be calculated off-line [83]. However, if the probability of detection is range (sensor-to-target) dependent, the IRM is also range dependent [32]. The relationship between the range and the IRM can be calculated off-line in a table format. The linear interpolation can then be used to approximate the IRM for a particular range [32]. However, in the multitarget case, the IRM is a function of not only the range of each target from the sensor but also the distances between the targets in the measurement space. Hence, the number of dependent variables of the IRM increases exponentially with the number of targets.

The off-line calculation of the IRM is viable only in small scale (in terms of the number of targets) problems because of memory limitations. However, even if there are many targets in the surveillance region, most of the time, only a small number of targets will be close to one another. Hence off-line calculation of the IRM may still be viable in most scenarios.

1.1.3 Centralized tracking

In this thesis, first we consider the sensor selection problem for centralized tracking system, in which all the deployed sensors are connected to a central fusion center (CFC) [4, 8]. In order to get maximum information, we want to use as many sensors as possible at each time step. Note that the information from any sensor is always positive as long as we know the accuracy of its measurements. However, because of physical limitations (e.g., limited frequency channels, limited capacity of processor) only a subset of sensors can be used at any one time [33]. Then, the problem is to select a subset of sensors that gives the optimal tracking performance.

In general once a sensor subset is activated, it remains active for some more time until the next sensor subset becomes active. That is, measurements are received from a sensor subset for a number of consecutive sensor revisit times. Although a particular sensor subset may be optimal at the time of initial activation, due to target motion, it may not remain optimal throughout the entire activation period. In order to get a sensor subset that remains optimal during the entire activation interval and not just at the time of activation, the overall tracking performance over the entire activation period is optimized.

In this thesis, two different scenarios are considered under the centralized architecture. In the first scenario, we assume that the total number of targets in the

surveillance region is known and fixed. The problem is then to select the sensors that allow one to best alleviate the measurement association ambiguities and therefore provide accurate target state estimates. Next we consider a scenario in which the number of targets in the surveillance region is unknown and time varying. The problem is then to select sensors to both accurately track existing targets and quickly detect new ones. This problem can be formulated as a bi-criterion optimization: one objective is to provide accurate state estimates of existing targets and the other is to maximize the probability of detecting new targets.

We present strategies for selecting the optimal sensor subset (n out of N) based on the multitarget PCRLB. If the total number of possible combinations $\binom{N}{n}$ is small, then we can evaluate each and every combination and select the best one. However, in large scale problems, finding the optimal sensor subset by complete enumeration is not viable. We present a search technique to obtain a suboptimal solution in real time. Local searches are typically sensitive to the initial guess, and an intelligent choice for the initial guess is proposed.

1.1.4 Distributed tracking

Next we extend the work of centralized architecture to distributed architecture, which has few local fusion centers (LFCs) and one CFC [8, 29]. In distributed architecture, each LFC uses a certain number of sensors at each time steps to get the information about the surveillance region. LFCs update the estimates of their local tracks using the measurements that are received from their local sensors via wireless channel and then report their estimates to CFC, which will combine all of these information to form the global tracks. Then, the problem is to decide which of the available sensors must be used by each LFC, and decide what frequency channel and transmission

power (if controllable) should be used by each active sensor.

In the literature, it is generally assumed that the LFCs are not fixed, i.e., the locations or the number of LFCs can be changed [39, 51]. This might be true if all the deployed sensors can act as a LFC and/or the LFCs are moving platforms. However, this is not always possible. Hence, in this thesis we consider the scenario where the LFCs are different from the sensors. After the deployment of all the sensors and the LFCs, the locations and the number of them are fixed.

In the literature, in order to reduce the computational load, sensors are clustered based on target or LFC locations and each cluster is handled separately [12, 79]. However, target based clustering is not possible for two reasons: first, the number of targets may be different from the number of LFCs; second, targets may be closely spaced. In those papers, it is assumed that a particular number of sensors is required to track each target and one sensor can detect only one target [6, 54, 59]. This is not a valid assumption in practice. If we have a good estimate for one target at the current time, we could track that target even with one sensor at the next measurement time. In addition to that, if there are many targets in the coverage region of a sensor, then that sensor can detect all of them with corresponding detection probabilities. LFC based clustering is also not optimal if many targets are in one LFC's field of view while no or very few targets are in the fields of view of other LFCs. Hence, in this thesis we do not use any clustering for distributed architecture.

In general, sensors and fusion centers communicate through wireless frequency channels. Hence, the bandwidth limitation, which limits the communication, is an important issue [77]. In centralized tracking, we do not need to consider about the co-channel interference since all the sensors are connected to one fusion center and each sensor uses a different frequency channel. The advantage of distributed tracking

is the reusability of the frequency channels. However, there is a trade-off between reusability and reachability. The transmission power of the sensors can be software controllable with a minimum and maximum limit [14]. This will enable us to save the power as well as to increase the reachability of each sensor. However, in the literature there has been no consideration of the variable transmission power and the frequency reusability issues.

In this thesis, we consider the following scenario: the sensors and the LFCs are already deployed and their positions are known; each LFC can use a certain maximum number of sensors because of the physical limitation [11]; only a limited number of frequency channels are available; transmission power of each sensor is software controllable within certain lower and upper limits; in order to extract the signal, signal-to-noise ratio (SNR) at each LFC for each frequency must be greater than or equal to a known threshold level; LFCs communicate with CFC at each measurement time steps; active sensors are changed at every measurement step. In order to eliminate the redundancy, we force an additional constraint that one sensor can be used by at most one LFC.

First, we show that the PCRLB for the above architecture is same as the one for the centralized architecture. Then, we give the optimal formulation for the sensor management for the above scenario based on the PCRLB. This is an NP-hard multi-objective mixed-integer optimization problem, and very hard to find the optimal solution in real time. We propose an algorithm to find a suboptimal solution in real time by decomposing the original problem into subproblems. Subproblems are easier to solve using algorithms like CPLEX solver [17].

1.1.5 Decentralized tracking

Finally, we extend the above two works to decentralized architecture [4]. In the distributed architecture, CFC can perform sensor selection for all the local fusion centers by considering whole surveillance region information and then instruct the LFC to use the selected sensors. Even though this is the optimal way to select the sensors, this may not always be feasible due to computational and communication constraints. Hence, we consider the decentralized architecture, which does not have any CFC and each fusion center (FC) can communicate only with its neighbors [4, 50]. Since each FC communicates only with its neighbors, no FC has the global picture of all the targets and the sensors in the surveillance region. Hence, each FC must select its own sensors by considering which sensors are/may be used by its neighbors.

The main challenge in decentralized tracking is to decide what type of track information to communicate between the FCs so that communication requirement is low, fusion can be made easily and the result will be close to optimal. The possible options are measurements, tracks and tracklets [4, 24]. Even though communicating the measurements is the optimal way, it requires a lot of communication. Communicating tracks requires less communication compared to measurements. However, in this architecture a track information from a FC can reach another fusion center via multiple FCs directly or indirectly, and avoiding multiple counting of that information is difficult. Tracklet, which is a track for a target based on only the most recent measurements from its local sensors since the data from the tracker was last communicated for that target, is a good choice for solving the above problem [24]. In addition, tracklets require less communication and easy to fuse. Hence, in this thesis we assume that tracklets are communicated among the fusion centers.

In this thesis, we consider two different scenarios under decentralized tracking. In

the first scenario, we assumed that communication and sensor selection are synchronized over the FCs. That is, all the fusion centers change their sensors at the same time, send their tracklets just before sensor selection, and their sensor change intervals are equal and known. In this case, each fusion center has to decide its sensors by considering which sensors may be used by neighbors. Note that no FC exactly knows which sensors will be used by its neighbors.

Even though avoiding clustering, as explained for distributed tracking, will help improve the tracking performance, there is a possibility for a sensor to be selected by more than one fusion center. Using one sensor by more than one fusion center will introduce redundancy as well as correlation between tracklets. Hence, in order to avoid this problem sensors are clustered based on FCs, i.e., a sensor can only be used by the nearest FC.

The optimal PCRLB for decentralized architecture is not available yet, and we derive a suboptimal PCRLB for this architecture, and then formulate the problem as a combinatorial optimization problem. We propose a sequential sensor selection technique, in which FCs are numbered and perform the sensor selection in the number order, to improve the performance. The advantage of sequential sensor selection is that the effect of any mistake made by earlier FCs can be reduced by later FCs. In addition clustering can also be avoided, since later FCs can avoid the sensors used by prior FCs. However, the disadvantage of sequential sensor selection is that it requires more communication and the lattor FCs must wait until others select their sensors.

In the second scenario, we assume that FCs may change their sensors at different time steps, and exact sensor change times of neighbors are not known, instead only an interval is known. It introduces additional uncertainties, such as when new sensor

subset will be used by neighbors and when tracklets will be received, and these uncertainties must be incorporated in the objective functions. We derive the objective functions for already detected targets and possible new ones by incorporating all uncertainties. In addition, sensors are typically disposable and last until their energy drains. Hence, sensors must be used efficiently in order to extend their life time for the duration of tracking [18, 58]. However, if a replacement is available for a particular sensor, then it can be used without considering its remaining energy. Then, we derive an object function based on the remaining energies of the sensors, and their neighboring sensors.

In most cases, we may need more precise estimates for some high-priority targets than the others based on target types and states [76]. Then, we introduce weights to each target in the cost function so that we can give more weights to the obtainable accuracies of the high priority targets. Also, accuracies of predicted target states, which are used in PCRLB calculation, diminish with time. Hence, we propose to use weights for each time step based on the uncertainties in predicted target states.

In this architecture, the track estimates of neighbors at any time are not exactly known, since only the tracklets are received from them. Incorrect predictions of neighbors' estimates will result in erroneous sensor selection. In order to know the exact estimates of the neighbors, we may need to request updated tracks from neighbors just before selecting the sensors. In addition, in the asynchronous case we can avoid the clustering easily since FC knows which sensors are used by its neighbors. However, if two FCs decided to change their sensors at the same time, then we need clustering or collaboration to avoid using one sensor by more than one FC. Collaboration always leads to better sensor selection, with additional communication and computation loads.

In both cases, finding the optimal solutions of the problems in real time is prohibitively expensive, and we propose an iterative local search technique to find a suboptimal solution in real time. Also, we compare the performance of the proposed technique with those of genetic algorithm (GA) [27, 36, 53] and ant colony optimization (ACO) algorithm [21, 69].

1.2 Organization of the Thesis

This thesis is structured as follows: Chapter 2 describes the general sensor management and target tracking problem. Chapter 3 explains the derivation of the PCRLB for multitarget tracking. Chapters 4, 5 and 6 provide algorithms for sensor subset selection for centralized, distributed and decentralized architectures, respectively. We conclude with some thoughts on future directions in Chapter 7.

1.3 Related Publications

1.3.1 Journal articles

1. R. Tharmarasa, T. Kirubarajan and A. Sinha, "Multitarget-Multisensor Management for Decentralized Sensor Networks", Submitted to *IEEE Transaction on Aerospace and Electronic Systems*, 2007.

2. R. Tharmarasa, T. Kirubarajan, J. Peng and A. Sinha, "Dynamic Sensor Management for Distributed Tracking", Submitted to *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2007.

3. R. Tharmarasa, T. Kirubarajan, M. L. Hernandez and A. Sinha, "PCRLB Based

Multisensor Array Management for Multitarget Tracking", *IEEE Transaction on Aerospace and Electronic Systems*, vol. 43, no. 2, pp. 539–555, April 2007.

4. R. Tharmarasa, T. Kirubarajan and M. L. Hernandez, "Large-Scale Optimal Sensor Array Management for Target Tracking", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 5, pp. 803–814, September 2007.

1.3.2 Conference publications

1. R. Tharmarasa and T. Kirubarajan, "Collaborative Sensor Management for Decentralized Asynchronous Sensor Networks", *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, vol. 6699, San Diego, CA, August 2007.

2. R. Tharmarasa, T. Kirubarajan, A. Sinha and M. L. Hernandez, "Multitarget-Multisensor Management for Decentralized Sensor Networks", *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, vol. 6236, Orlando, FL, April 2006.

3. R. Tharmarasa, T. Kirubarajan and J. Peng, "Dynamic Sensor Management for Distributed Tracking", *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, vol. 5913, San Diego, CA, August 2005.

4. R. Tharmarasa, T. Kirubarajan and M. L. Hernandez, "PCRLB Based Multisensor Array Management for Multitarget Tracking", *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, vol. 5428, Orlando, FL, April 2004.

Chapter 2

Multitarget Tracking and Sensor Management

2.1 Target Tracking

The process of determining the likely value of a quantity of interest from incomplete, inaccurate and uncertain observations is called estimation. The task of continuous estimation of the state of a moving object (also called target) with time is called tracking. The estimation of the current state of a dynamic system from noisy measurements is called filtering and estimating the future state using current measurements is called prediction. The tracking system should produce the measure of the accuracy of the state estimates in addition to the state estimates [5, 8].

A tracking system typically operates in the manner shown in Figure 2.1 with the objective of getting more accurate estimates by managing the sensors. Targets are tracked using the measurements received from the sensors that are managed by the sensor manager, which uses the past estimates of the tracks as the input. The

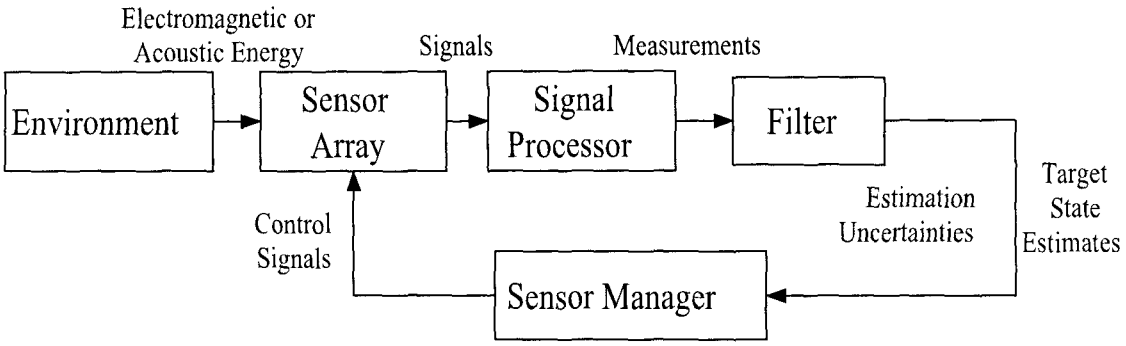


Figure 2.1: A tracking system with sensor management

following sections describe how target tracking is performed.

2.1.1 Models

In order to track a target, at least two models are required [5]:

1. System model: a model describing the evolution of the state with time.

2. Measurement model: a model relating the noisy measurements to the state.

The possible system models are:

- nonlinear system model

$$x_{k+1} = f_k(x_k) + \nu_k \tag{2.1}$$

- linear system model

$$x_{k+1} = F_k\, x_k + \nu_k \tag{2.2}$$

The possible measurement models are:

- nonlinear measurement model

$$z_k = h_k(x_k) + \omega_k \tag{2.3}$$

- linear measurement model

$$z_k = H_k\ x_k + \omega_k \tag{2.4}$$

where f_k and h_k are nonlinear functions, F_k and H_k are known matrices, x_k is the state of the target, z_k is the measurement vector, ν_k is the process noise and ω_k is the measurement noise at measurement time k. In this thesis, we assume that ν_k is Gaussian with zero mean and covariance Γ_k, and ω_k is Gaussian with zero mean and covariance Σ_k.

2.1.2 Filtering algorithms

In the Bayesian approach to dynamic state estimation, we want to construct the posterior probability density function (pdf) of the state given all the received measurements so far. Since this pdf contains all available statistical information, it may be said to be the complete solution to the estimation problem. In principle, an optimal estimate of the state may be obtained from the pdf. In recursive filtering the received measurements are processed sequentially rather than as a batch so that it is not necessary to store the complete measurement set nor to reprocess existing measurement if a new measurement becomes available. Such a filter consists of essentially two stages: prediction and update.

The prediction stage uses the system model to predict the state pdf forward from

one measurement time to the next. Suppose that the required pdf $p(x_k|Z_k)$ at measurement time k is available, where $Z_k = [z_1, z_2, \ldots, z_k]$. The prediction stage involves using the system model (2.1) to obtain the prior pdf of the state at measurement time $k+1$ and given by

$$p(x_{k+1}|Z_k) = \int p(x_{k+1}|x_k)p(x_k|Z_k)dx_k \tag{2.5}$$

The update stage uses the latest measurement to modify the prediction pdf. At next measurement time $k+1$, a measurement z_{k+1} becomes available and will be used to update the prior via Bayes' rule:

$$p(x_{k+1}|Z_{k+1}) = \frac{p(z_{k+1}|x_{k+1})p(x_{k+1}|Z_k)}{p(z_{k+1}|Z_k)} \tag{2.6}$$

In the above the likelihood function $p(z_{k+1}|x_{k+1})$ is defined by the measurement model (2.3).

The above recursive propagation of the posterior density is only a conceptual solution, and in general it cannot be determined analytically. Analytical solution exists only in a restrictive set of cases.

2.1.2.1 Kalman Filter

The Kalman filter assumes that the state and measurement models are linear and the initial state error and all the noises entering into the system are Gaussian. Under the above assumptions, if $p(x_k|Z_k)$ is Gaussian, it can be proved that $p(x_{k+1}|Z_{k+1})$ is also Gaussian, and can be parameterized by a mean and covariance [5].

If the state and measurement equations are given by (2.2) and (2.4) respectively, then the Kalman filter algorithm can be viewed as the following recursive relationship

[5]:

$$p(x_k|Z_k) = \mathcal{N}(x_k; \hat{x}_{k|k}, P_{k|k}) \tag{2.7}$$

$$p(x_{k+1}|Z_k) = \mathcal{N}(x_{k+1}; \hat{x}_{k+1|k}, P_{k+1|k}) \tag{2.8}$$

$$p(x_{k+1}|Z_{k+1}) = \mathcal{N}(x_{k+1}; \hat{x}_{k+1|k+1}, P_{k+1|k+1}) \tag{2.9}$$

where

$$\hat{x}_{k+1|k} = F_{k+1}\hat{x}_{k|k} \tag{2.10}$$

$$P_{k+1|k} = \Gamma_{k+1} + F_{k+1}P_{k|k}F_{k+1}^{'} \tag{2.11}$$

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1}(z_{k+1} - H_{k+1}\hat{x}_{k+1|k}) \tag{2.12}$$

$$P_{k+1|k+1} = P_{k+1|k} - K_{k+1}H_{k+1}P_{k+1|k} \tag{2.13}$$

with

$$S_{k+1} = H_{k+1}P_{k+1|k}H_{k+1}^{'} + \Sigma_{k+1} \tag{2.14}$$

$$K_{k+1} = P_{k+1|k}H_{k+1}^{'}S_{k+1}^{-1} \tag{2.15}$$

In the above, $\mathcal{N}(x; \hat{x}, P)$ is a Gaussian density with argument x, mean $\hat{x}$ and covariance P, and $[\cdot]'$ denotes the transpose.

This is the optimal solution to the tracking problem if the above assumptions hold [5]. The implication is that no algorithm can do better than a Kalman filter in this linear Gaussian environment.

In many situations of interest the assumptions made above do not hold. Hence the Kalman filter cannot be used as described above, and approximations are necessary.

2.1.2.2 Information Filter

Information filter is an alterative form of Kalman filter [5, 29]. Information state vector $y_{k|k-\iota}$ and information matrix $Y_{k|k-\iota}$ are defined as:

$$y_{k|k-\iota} = P^{-1}_{k|k-\iota}\hat{x}_{k|k-\iota} \tag{2.16}$$

$$Y_{k|k-\iota} = P^{-1}_{k|k-\iota} \tag{2.17}$$

The measurement information vector and corresponding matrix are defined as:

$$i_k = H'_k\Sigma^{-1}_k z_k \tag{2.18}$$

$$I_k = H'_k\Sigma^{-1}_k H_k \tag{2.19}$$

Then the estimate is given as:

$$y_{k|k} = y_{k|k-1} + i_k \tag{2.20}$$

$$Y_{k|k} = Y_{k|k-1} + I_k \tag{2.21}$$

The advantage of information filter is that measurements from multiple sensors can be filtered easily by summing their information matrices and vectors:

$$y_{k|k} = y_{k|k-1} + \sum_{j=1}^{n} i_{k,j} \tag{2.22}$$

$$Y_{k|k} = Y_{k|k-1} + \sum_{j=1}^{n} I_{k,j} \tag{2.23}$$

where n is the number of sensors.

2.1.2.3 Extended Kalman Filter (EKF)

If the state and measurement equations are given by (2.1) and (2.3) respectively, then a local linearization of the equations may be a sufficient description of the nonlinearity. Local linearizations of the above functions are

$$\hat{F}_k = \left.\frac{df_k(x)}{dx}\right|_{x=\hat{x}_{k-1|k-1}} \tag{2.24}$$

$$\hat{H}_k = \left.\frac{dh_k(x)}{dx}\right|_{x=\hat{x}_{k|k-1}} \tag{2.25}$$

The EKF is based on that $p(x_k|Z_k)$ is approximated by a Gaussian. Then all the equations of the Kalman filter can be used with this approximation and the linearized functions [5].

If the true density is substantially non-Gaussian, then a Gaussian can never describe it well. In such cases, particle filters will yield an improvement in performance in comparison to the EKF.

2.1.2.4 Particle Filtering

The Particle Filter [28] (see also [3] and [23]) provides a mechanism for representing the density $p(x_k|Z_k)$ of the state vector x_k at sampling time k as a set of random samples (particles) $\{x_k^{(i)} : i = 1, 2, \ldots, m\}$, with associated weights $\{w_k^{(i)} : i = 1, 2, \ldots, m\}$, where m is the number of particles. It then uses the principle of Importance Sampling to propagate and update these particles and their associated weights as and when new measurements become available [22].

We take the Importance Density to be the prior $p(x_k|x_{k-1})$ and use the method of Sampling Importance Resampling (SIR) to produce a sample of equally weighted

particles that approximate $p(x_k|Z_k)$, i.e.,

$$p(x_k|Z_k) \approx \frac{1}{m}\sum_{i=1}^{m}\delta(x_k - x_k{}^{(i)}) \tag{2.26}$$

where $\delta(.)$ is the Dirac delta function. The SIR method works as follows:

- **Prediction**: For each particle $x_{k-1}^{(i)}$, generate $\nu_{k-1}^{(i)}$ according to the known distribution of the transition noise and then a sample $x_{k|k-1}^{(i)}$ from the prior distribution $p(x_k|x_{k-1})$ can be obtained using the state propagation equation (2.1).

- **Weighting:** The information given by the observation can be utilized to find the importance weights. Each particle is given an importance weight $w_k^{(i)}$ using the formula

$$w_k^{(i)} = p(z_k|x_{k|k-1}^{(i)}) \tag{2.27}$$

- **Resampling:** The weighted samples will be resampled to eliminate those with low weights, multiply those with high weights and regenerate those with equal weights. The new m particles are sampled with replacement from $\{x_{k|k-1}^{(1)}, x_{k|k-1}^{(2)}, \ldots, x_{k|k-1}^{(m)}\}$ so that the probability of sampling particle i is proportional to w_k^i. Then new samples $\{x_k^{(1)}, x_k^{(2)}, \ldots, x_k^{(m)}\}$ will have equal weights $(1/m)$.

At each stage, the mean of the posterior distribution is used to determine an estimate $\hat{x}_k$ of the target state x_k, i.e.,

$$\hat{x}_k = \mathbb{E}[x_k|Z_k] \tag{2.28}$$

$$\approx \frac{1}{m}\sum_{i=1}^{m} x_k{}^{(i)} \tag{2.29}$$

The strength of the technique is that we are not restricted by the assumptions of linearity and Gaussian noise that are necessary in implementing Kalman filter based techniques.

2.2 Multitarget Tracking

Figure 2.2 illustrates the basic elements of a conventional multitarget tracking system.

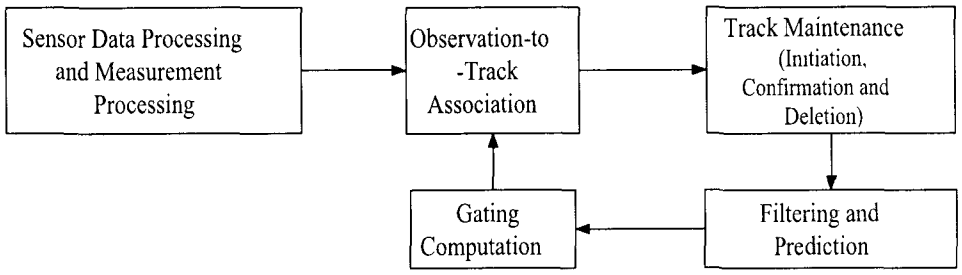


Figure 2.2: Basic elements of a conventional multitarget tracking system.

A signal processing unit converts the signals from the sensor to measurements, which is the input of the multiple target tracking system. In the tracking system, track is a symbolic representation of a target moving through the surveillance region and it is represented by a filter state. The incoming measurements are used for the track maintenance, which refers to the functions of track initiation, confirmation, and deletion [4, 8]. Measurements, which are not assigned to any of the existing tracks, can initiate new tentative tracks, and it becomes confirmed when the measurements included in that track satisfy the confirmation criteria. Similarly, a track that is not updated becomes degraded, and it will be deleted if not updated within some reasonable interval.

For existing tracks, a validation procedure is used to limits the region in the measurement space where the tracking system looks to find the measurement from

the targets. Measurements outside the validation region can be ignored, since they are very unlikely to have originated from the target of interest. It is possible to have more than one measurement in the validation region, and a more detailed association technique is used to determine the final pairings. After updating the tracks with associated measurements, tracks are predicted ahead to the arrival time for the next set of measurements. Gates are placed around these predicted positions and the processing cycle repeats.

If the true measurement conditioned on the past is normally (Gaussian) distributed with its probability density function given by

$$p(z_{k+1}|Z_k) = \mathcal{N}[z_{k+1}; \hat{z}_{k+1|k}, S_{k+1}] \tag{2.30}$$

where $\hat{z}_{k+1|k}$ is the predicted (mean) measurement at time $k+1$ and $S_k + 1$ is the measurement prediction covariance given by (2.14), then the true measurement will be in the following region

$$\mathcal{V}(k+1, \gamma) = \{z : [z - \hat{z}_{k+1|k}]' S_{k+1}^{-1} [z - \hat{z}_{k+1|k}] < \gamma\} \tag{2.31}$$

with the probability determined by the gate threshold γ. The region defined by (2.31) is called gate or validation region,

Figures 2.3 and 2.4 illustrate the gating for two well-separated and closely-spaced targets, respectively. In the figures, $\bullet$ indicates the expected measurement and the $*$ indicates the received measurement.

Any measurement falling inside the validation region is called a validated measurement. Since more than one measurement is validated in Figure 2.3, there is an association ambiguity as to which, if any, of the validated measurements is target

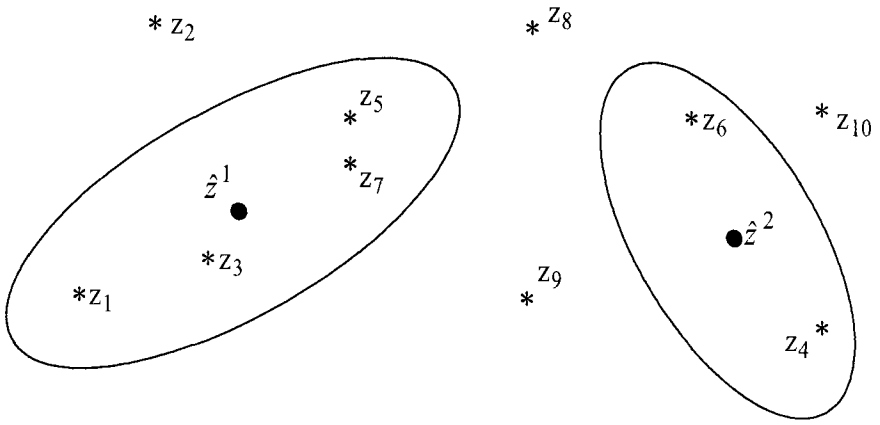


Figure 2.3: Validation regions of two well-separated targets.

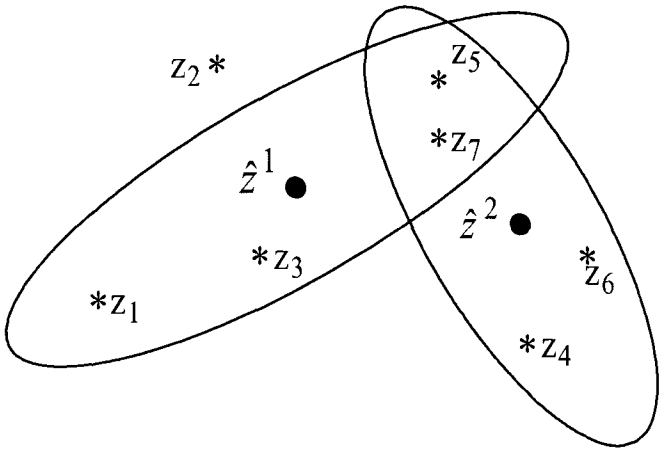


Figure 2.4: Validation regions of two closely spaced targets and the measurements inside those validation region

originated, assuming that at most one measurement is target generated.

If the targets are well separated in the measurement space, i.e., the validation regions are not overlapping as shown in Figure 2.3, the problem is basically same as the single target tracking. However, if the targets are closely spaced, which causes the corresponding validation regions to overlap (see Figure 2.4), the validation region for a particular target may also contain detections from nearby targets as well as clutter detections. Hence, there is a need for a data association technique in order to resolve the measurement origin uncertainty.

2.2.1 Data association

2.2.1.1 Well-separated targets

The problem of tracking well-separated multiple targets in clutter considers the situation where there are possibly several measurements in the validation region of each target. The set of validated measurements consists of:

- true measurement (if detected and falls within the gate)
- false alarms

The common mathematical model for such false measurements is that

- their spatial distribution is uniform within the surveillance region
- they are independent across time.

Then the problem, which is called as data association, is that of associating the measurements in each validation region with the corresponding track (target). The possible approaches are [4]:

- Nearest Neighbor (NN): this is the simplest possible approach, and uses the measurement nearest to the predicted measurement as if it were the correct one.
- Strongest Neighbor (SN): select the strongest measurement among the validated ones.
- Probabilistic Data Association (PDA): this is a Bayesian approach, and associates probabilistically all the validated measurements to the target of interest.

PDA is the standard technique used for data association in conjunction with the Kalman filter or the extended Kalman filter. In most of the particle filtering algorithms, NN is used since it requires less computation.

2.2.1.2 Closely-spaced targets

When the target are closely spaced, one measurement could be originated from any one of the target or clutter. Then the following assumptions are made

- one measurement is originated from at most one target
- one target can generate at most one measurement

Under the above assumptions, following approaches are possible for data association: [4]

- Joint Probabilistic Data Association (JPDA): This is a target oriented approach, and it is an extension of PDA.
- Multiple Hypothesis Tracking (MHT): This is a measurement oriented approach, in which probability that a measurement sequence is originated from an established target or a new target is calculated.

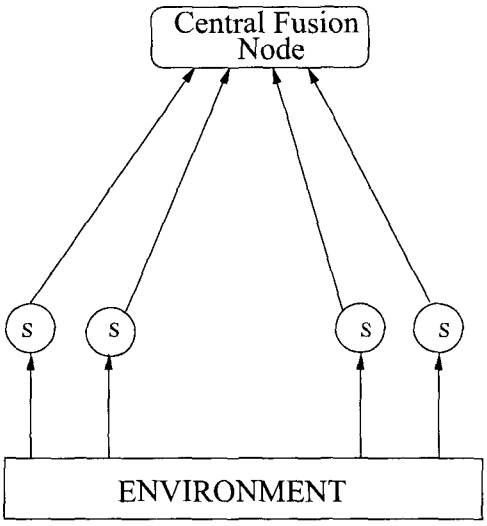


Figure 2.5: Centralized architecture

JPDA can only be applied to already established tracks, while MHT can be used for track initiation as well. Even though MHT is the optimal approach, it is not feasible when a large number of measurement steps are considered. Then, suboptimal version of MHT, called S-D assignment, is the widely used technique for data association [19, 57].

2.3 Architectures

Three major types of architecture are commonly used in multisensor-multitarget tracking applications. Those are centralized, distributed and decentralized architectures, and explained in detail in the following sections [4, 50, 78].

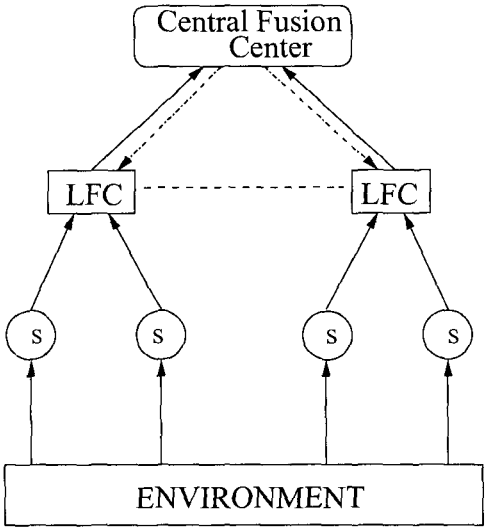


Figure 2.6: Distributed architecture

2.3.1 Centralized tracking

In the centralized architecture, shown in Figure 2.5, in general more than one sensor are monitoring the region of interest for detecting and tracking the targets in that region. All the sensors generate the measurements at each revisit time and report those measurements to a central fusion center that fuses all the measurements and updates the tracks. This is the optimal architecture in terms of tracking performance. However in a very large surveillance region with many sensors, this architecture may not be feasible because of limited resources, e.g., communication bandwidth, computation power etc.

2.3.2 Distributed tracking

In order to avoid the heavy communication and computation requirement of centralized architecture, an alternative architecture called distributed architecture (some authors use the term "hierarchical architecture" instead) is used, and it is shown in

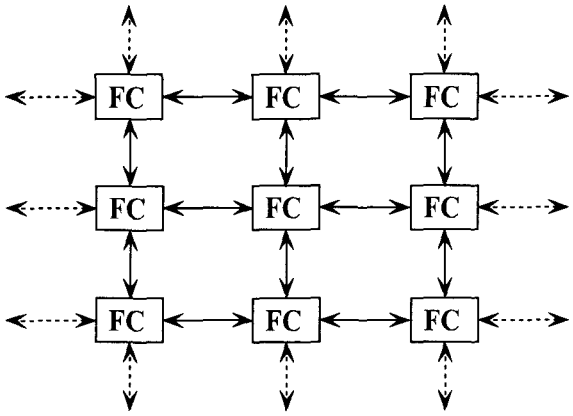


Figure 2.7: Decentralized architecture

Figure 2.6 [78]. In this architecture, sensors are connected to LFCs and LFCs are in turn connected to a CFC. Each LFC updates its local tracks based on the measurements from the local sensors and sends its tracks to CFC. Then, the CFC performs the track-to-track fusion and may sends back the updated tracks to the LFCs, if feedback path is available.

2.3.3 Decentralized tracking

In some cases, neither centralized nor distributed tracking is possible, if there is no central fusion center that can communicate with all the sensors or LFC in a large surveillance region. In these cases, another architecture called decentralized architecture, in which one has multiple FCs and no CFC, is used, and it is shown in Figure 2.7 [78]. In this architecture, each FC gets the measurements from one or more sensors that are connected to it, and uses those measurements to update its tracks. In addition to that, tracks are also updated whenever an FC gets additional information from its neighbors. Note that even though many FCs are available, each FC can communicate only with its neighbors, the FCs within the communication

distance, every few measurement time steps.

2.3.4 Track fusion

The main challenge in distributed and decentralized tracking is deciding what to communicate between the FCs so that communication requirement is low, fusion can be made easily and the result will be close to optimal. Even though communicating the measurements is the optimal way, in general, it requires a lot of communication. Hence other approaches that require less communication must be considered.

The first option is to communicate the tracks instead of raw measurements. If the tracks are uncorrelated, then they can be fused easily at FCs [4]. However, due to common prior information and common process noise, they are correlated and fusion becomes more complicated. Hence, in order to avoid double counting of common information, the error cross-covariance must be calculated and accounted for during fusion. However, exact error cross covariance calculation needs the information about when a measurement update is made along with the Kalman gain used at each measurement step by each tracking platform. This imposes additional computational and communication load.

The second option is to communicate tracklets. A tracklet is a track computed such that its errors are not cross-correlated with the errors of any other data in the system for the same target [24]. It is equivalent to a track for a target based on only the most recent measurements since the data from the tracker was last communicated for that target. The main advantages of tracklet are reduced communication and computation requirements. Hence, in this thesis, we assume that tracklets are communicated between the fusion centers. The calculation of tracklet is explained below.

The state and measurement equations are given by (2.2) and (2.3) respectively. Suppose last communication was performed at time $k-l$ and the next communication is at time k. The state estimates and its covariances of a target just after the last communication and just before the next communication are $\hat{x}_{k-l|k-l}$, $P_{k-l|k-l}$ and $\hat{x}_{k|k}$, $P_{k|k}$. Then the information filter based tracklet is given by [25, 71]:

$$y_k = P_{k|k}^{-1}\,\hat{x}_{k|k} - P_{k|k-l}^{-1}\,\hat{x}_{k|k-l} \tag{2.32}$$

$$Y_k = P_{k|k}^{-1} - P_{k|k-l}^{-1} \tag{2.33}$$

where

$$\hat{x}_{k|k-l} = F_{k,k-l}\,\hat{x}_{k-l|k-l} \tag{2.34}$$

$$P_{k|k-l} = F_{k,k-l}\,P_{k-l|k-l}\,F'_{k,k-l} + \Gamma_{k,k-l} \tag{2.35}$$

In (2.2), F_k means the state transition matrix from time step $k-1$ to k. In the above equations $F_{k,k-l}$ is the state transition matrix from time step $k-l$ to k, and $\Gamma_{k,k-l}$ is the corresponding process noise covariance.

If the particular target has been initialized only after last communication, then

$$y_k = P_{k|k}^{-1}\,\hat{x}_{k|k} \tag{2.36}$$

$$Y_k = P_{k|k}^{-1} \tag{2.37}$$

Fusion centers must update their tracks whenever they get tracklets from other fusion centers. As the first step, they have to associate the tracklets with existing tracks. To check whether the tracklet j is corresponding to track i, the errors in the difference between the estimates and the corresponding covariances are calculated as

follows [4]:

$$\Delta_{ij} = \hat{x}^i_{k|k} - Y^{j^{-1}}_k \; y^j_k \tag{2.38}$$

$$T_{ij} = P^i_{k|k} + Y^{j^{-1}}_k \tag{2.39}$$

If $\Delta'_{ij} T^{-1}_{ij} \Delta_{ij} \leq D$, where D is the threshold, then we accept that both estimates belong to same target.

After a tracklet is associated with a track, that track is updated as follows:

$$(P^{\text{fused}}_{k|k})^{-1} = P^{-1}_{k|k} + Y_k \tag{2.40}$$

$$(P^{\text{fused}}_{k|k})^{-1} \; \hat{x}^{\text{fused}}_{k|k} = P^{-1}_{k|k} \; \hat{x}_{k|k} + y_k \tag{2.41}$$

If no target is associated to a tracklet, then a new track is formed and it is given by:

$$P_{k|k} = Y^{-1}_k \tag{2.42}$$

$$\hat{x}_{k|k} = P_{k|k} \; y_k \tag{2.43}$$

2.3.4.1 Performance of distributed tracking with full feedback at every measurement step

In this section, let us see the performance of distributed tracker, if LFCs send their tracks at every measurement steps, and CFC send the updated tracks back to LFCs.

If the estimate and corresponding covariance of the j-th LFC at time k is given by $\hat{x}^j_{k|k}$ and $P^j_{k|k}$, $j = 1, 2, \ldots, M$, respectively, the central estimate in terms of the

local estimates is given by [4, 15]

$$P_{k|k}^{-1}\,\hat{x}_{k|k} \;=\; P_{k|k-1}^{-1}\,\hat{x}_{k|k-1} + \sum_{j=1}^{M}\left[(P_{k|k}^{j})^{-1}\,\hat{x}_{k|k}^{j} - \,(P_{k|k-1}^{j})^{-1}\,\hat{x}_{k|k-1}^{j}\right] \quad (2.44)$$

and the corresponding covariance is given by

$$P_{k|k}^{-1} \;=\; P_{k|k-1}^{-1} + \sum_{j=1}^{M}(P_{k|k}^{j})^{-1} - (P_{k|k-1}^{j})^{-1} \quad (2.45)$$

where M is the number of LFCs.

Under the assumption that the LFCs have the feedback from the CFC at every measurement step, $P_{k|k-1}^{j} = P_{k|k-1}$ for all j. Then $(P_{k|k}^{j})^{-1} - (P_{k|k-1}^{j})^{-1}$ gives the information gained from the j-th LFC's sensors.

Hence, the performance of distributed tracking with feedback at every measurement time step is equivalent to the performance of centralized tracking.

2.4 Sensor Management

Popoli [60]: "Sensor management is really just the study of ways to improve or optimize the measurement process in a tracking system". In sensor management, one has to decide how to use the sensor resources efficiently in the future time steps in order to achieve the better tracking performance, given the current target estimates [78]. Major issues in sensor management for target tracking are optimal path planning, sensor placement, and sensor selection.

In the optimal path planning, we have to decide the states of the moving sensors in the future revisit time steps by considering the maximum velocities and turn rates of the sensors [30, 62, 68].

The objectives of optimal sensor placement are to decide [2, 32, 61, 82, 84]:

- the time at which new sensors should be deployed, given the current estimate of the (distribution of the) target state.
- the size and configuration of the next deployment.

An optimal sensor selection algorithm, the major topic of this thesis, selects which of the sensors already deployed should be used at each measurement time [38, 49].

Sensor subset selection is crucial in tracking applications in which a large number of sensors are available, however only a limited number of sensors can be used at any one time step due to physical limitations. For an example, within the domain of Anti-Submarine Warfare, because of physical and communication bandwidth limitations, at any one time the central tracker may only be able to receive measurements from a limited number of the total deployed sensors (sonobuoys).

In this thesis, we consider the problem of sensor subset selection for multitarget tracking under centralized, distributed and decentralized architectures. PCRLB [72], which gives a lower bound of estimation error covariance, is used as the performance measure of tracking systems.

Chapter 3

Posterior Cramér-Rao Lower Bound

3.1 Background

Let X_k be an unknown and random state vector, and let $\hat{X}_k(Z_k)$ be an unbiased estimate of X_k based on the measurement data, Z_k. The PCRLB, which is defined to be the inverse of the Fisher Information Matrix (FIM), $J(k)$ [75], then gives a lower bound of the error covariance matrix, i.e.,

$$C(k) \triangleq \mathbb{E}\{[\hat{X}_k(Z_k) - X_k][\hat{X}_k(Z_k) - X_k]'\} \geq J(k)^{-1} \tag{3.1}$$

where $\mathbb{E}$ denotes expectation over $(X_k,\ Z_k)$. The inequality in (3.1) means that $C(k) - J(k)^{-1}$ is a positive semi-definite matrix.

A recursive formula for the evaluation of the posterior FIM, $J(k)$, is given by [73]:

$$J(k+1) = \underbrace{D_k^{22} - D_k^{21}\left(J(k) + D_k^{11}\right)^{-1} D_k^{12}}_{J_X(k+1)} + J_z(k+1) \tag{3.2}$$

where

$$\begin{aligned}
D_k^{11} &= \mathbb{E}\left\{-\Delta_{X_k}^{X_k} \ln p(X_{k+1}|X_k)\right\} && (3.3)\\
D_k^{12} &= \mathbb{E}\left\{-\Delta_{X_k}^{X_{k+1}} \ln p(X_{k+1}|X_k)\right\} && (3.4)\\
D_k^{21} &= (D_k^{12})' && (3.5)\\
D_k^{22} &= \mathbb{E}\left\{-\Delta_{X_{k+1}}^{X_{k+1}} \ln p(X_{k+1}|X_k)\right\} && (3.6)\\
J_z(k+1) &= \mathbb{E}\left\{-\Delta_{X_{k+1}}^{X_{k+1}} \ln p(z_{k+1}|X_{k+1})\right\} && (3.7)
\end{aligned}$$

and Δ_α^β is a second-order partial derivative operator whose (i,j)th term is given by

$$\Delta_\alpha^\beta(i,j) = \frac{\partial^2}{\partial\alpha(i)\partial\beta(j)} \tag{3.8}$$

$\alpha(i)$ and $\beta(i)$ are the i-th components of vectors α and β, respectively. In the above, $z_{k+1} = \left[z_{k+1}(1),\ z_{k+1}(2),\ \ldots,\ z_{k+1}(n)\right]$, where $z_{k+1}(i)$ is the measurement vector at sensor i at sampling time $k+1$ and n is the number of sensors utilized at sampling time $k+1$.

3.2 PCRLB for Multitarget Tracking

Let the state vector at time k, obtained by stacking the state vectors of all targets, be denoted by $X_k = \left[x_k^1, x_k^2, \ldots, x_k^T\right]'$, where x_k^t is the state vector of target t and T is the total number of targets in the surveillance region. If we assume that targets are

moving independently and the state equation of each target is linear, then the overall state equation is given by

$$X_{k+1} = F_k X_k + \nu_k \tag{3.9}$$

where

$$F_k = \text{diag}\left(F_k^1,\ F_k^2,\ \ldots,\ F_k^T\right) \tag{3.10}$$

$$\nu_k = \left[\nu_k^1,\ \nu_k^2,\ \ldots,\ \nu_k^T\right]' \tag{3.11}$$

In the above, F_k^t is the state transition matrix and ν_k^t is the process noise of target t. If ν_k^t is Gaussian with zero mean and covariance Γ_k^t, then the covariance matrix of ν_k, Γ_k, is given by

$$\Gamma_k = \text{diag}\left(\Gamma_k^1,\ \Gamma_k^2,\ \ldots,\ \Gamma_k^T\right) \tag{3.12}$$

It can easily be shown that in the case of linear, Gaussian dynamics (e.g., [65]) we have:

$$D_k^{11} = F_k' \Gamma_k^{-1} F_k \tag{3.13}$$

$$D_k^{12} = -F_k' \Gamma_k^{-1} \tag{3.14}$$

$$D_k^{22} = \Gamma_k^{-1} \tag{3.15}$$

Using the Matrix Inversion Lemma and (3.13) – (3.15), it can then be shown that

$$J_X(k) = \left[\Gamma_{k-1} + F_{k-1} J(k-1)^{-1} F_{k-1}'\right]^{-1} \tag{3.16}$$

The matrix $J_X(k)$ gives the prior information regarding the target states at time k. The measurement contribution is then given by $J_z(k)$. We note that $J_z(k)$ is dependent on the states of the operational sensors. The derivation of $J_z(k)$ is the focus of the next section.

3.2.1 Measurement contribution to the PCRLB

Consider the general case in which there is measurement origin uncertainty, with measurements originating from one of the targets or from clutter. The j-th measurement at the i-th sensor is given by

$$z_k(j,i) = \begin{cases} h_k^i(x_k^t) + \omega_k^i(j) & \text{if originated from target } t \\ v_k^i(j) & \text{if false alarm} \end{cases} \tag{3.17}$$

where h_k^i is (in general) a nonlinear function, $\omega_k^i(j)$ is a zero mean Gaussian random variable with covariance Σ_k and $v_k^i(j)$ is uniformly distributed across the surveillance region A (with hyper-volume, V). The probability mass function of the number of false alarms, $\mu_{FA}(m)$, which is Poisson-distributed with mean λV, is given by

$$\mu_{FA}(m) = \frac{e^{-\lambda V}(\lambda V)^m}{m!} \tag{3.18}$$

where m is the number of false alarms and λ is the spatial density of the false alarms.

When multiple targets are present, the association between the measurements and the targets is not known and must be considered in the PCRLB calculation. The following assumptions are made regarding the measurements:

- Each measurement can be generated by one of the targets or the clutter.
- Each target can produce zero or one measurement at any one time.

If sensors have independent measurement processes, J_z can be written as

$$J_z(k) = \sum_{i=1}^{n} J_{z_i}(k) \tag{3.19}$$

where

$$J_{z_i}(k) = \sum_{m_k(i)=0}^{\infty} p(m_k(i)) J_{z_i}(m_k(i), k) \tag{3.20}$$

$$J_{z_i}(m_k(i), k) = \mathbb{E}\left\{ -\Delta_{X_k}^{X_k} \ln p(z_k(i)|X_k, m_k(i)) \middle| m_k(i) \right\} \tag{3.21}$$

In the above, n is the number of sensors used at time k and $m_k(i)$ is the number of measurements at sensor i at time k.

The probability of receiving $m_k(i)$ measurements, $p(m_k(i))$, from sensor i is given by

$$p(m_k(i)) = \sum_{d=0}^{\min(T, m_k(i))} \left(\mu_{FA}\left(m_k(i) - d\right) \sum_{D_k^i} \left(\prod_{t=1}^{T} \left(P_D^t(i)\right)^{D_k^i(t)} \left(1 - P_D^t(i)\right)^{(1-D_k^i(t))} \right) \right) \tag{3.22}$$

where D_k^i is the detection vector that indicates which targets are detected at sensor i (at time k)[1]. The total number of targets that are detected is d, i.e., $\sum_{t=1}^{T} D_k^i(t) = d$. $P_D^t(i)$ is the probability of detection of target t by sensor i.

The probability density function of the measurement $z_k(i)$ conditioned on X_k and $m_k(i)$ is given by [42]

$$p(z_k(i)|X_k, m_k(i)) = \sum_{a_k(i)} p(z_k(i)|X_k, m_k(i), a_k(i)) p(a_k(i)|m_k(i)) \tag{3.23}$$

[1] $D_k^i(t)$ takes the value 1 if target t is detected and 0 otherwise.

where $a_k(i)$ is the association vector that indicates which measurement originated from which target. Each element $a_k(j,i)$ of $a_k(i)$ is a random variable that takes a value in [0, 1, ..., T], with 0 indicating a false alarm. $a_k(j,i) = t$ indicates that the measurement j originates from target t. If the targets are well separated in the measurement space, there is no measurement origin uncertainty in terms of targets and any one measurement can be originated from a known target or clutter. However, if the targets are closely-spaced or cross one another, it is hard to find the association vector and all possible associations must be considered in the calculation of measurement information $J_z(k)$.

In the Appendix A, we show that the (t_1, t_2)th block of $J_{z_i}(k)$ is given as follows:

$$\left[J_{z_i}(k)\right]_{t_1 t_2} = \mathbb{E}\left\{ \left[H_k^i\right]'_{t_1} \left[Q_k^i\right]_{t_1 t_2} \Sigma_k^{-1} \left[H_k^i\right]_{t_2} \middle| X_k \right\} \tag{3.24}$$

where

$$\left[H_k^i\right]_t(\alpha, \beta) = \frac{\partial h_k^i(\alpha, x_k^t)}{\partial x_k^t(\beta)} \tag{3.25}$$

$$\left[Q_k^i\right]_{t_1 t_2} = \sum_{m_k(i)=0}^{\infty} p(m_k(i)) \left[Q_k^i(m_k(i))\right]_{t_1 t_2} \tag{3.26}$$

and $\left[H_k^i\right]_t(\alpha, \beta)$ denotes the (α, β)th element of matrix $\left[H_k^i\right]_t$. Q_k^i is the information reduction matrix (IRM) for sensor i and $\left[Q_k^i\right]_{t_1 t_2}$ is the (t_1, t_2)th block of the IRM. $\left[Q_k^i(m_k(i))\right]_{t_1 t_2}$ is given in (A.9) of the Appendix A. No closed form analytical solution exists for the IRM Q_k^i, which must then be calculated using a numerical integration technique.

Now, in general the probability of detection depends on the range of the target from the sensor. As a result, the IRM is also a function of the target range. In the

single target case, the relationship between the IRM and range can be determined off-line during a pre-processing stage [32]. Importantly then, real-time sensor management can be performed because the one-time computational "hit" has already be taken, prior to online implementation.

However, in the multitarget case, the IRM depends not only on the range of each target from the sensor, but also on the distances between the targets in the measurement space. Hence, the number of variables on which the IRM depends increases exponentially with the number of targets. Even though off-line calculation is viable in small scale problems, it is difficult for large scale problems because of memory limitation. However, even if we have many targets, frequently only a few will be in close proximity to one another (i.e., be in one group), and off-line calculation is again viable. This idea is explored (along with other potential simplifications) in the next section.

3.2.2 Reducing the computational complexity

We propose the following approximations in order to reduce the computational load of the ensuing calculation.

- **Approximation 1:** Measurements can be restricted to the validation gate of each target. We can then group the targets whose validation gates overlap with one another. After that, each group can be treated separately. For example, in Figure 3.1, there are 8 targets and the ellipses indicate the corresponding validation gates. Only targets 2 and 3 affect target 1, hence they are considered as a group. Note that even though target 1 does not overlap with target 3, they are connected by target 2 and they affect each other indirectly. Similarly, targets 5, 6, 7 and 8 are in one group and target 4 is alone in another group. Hence, in

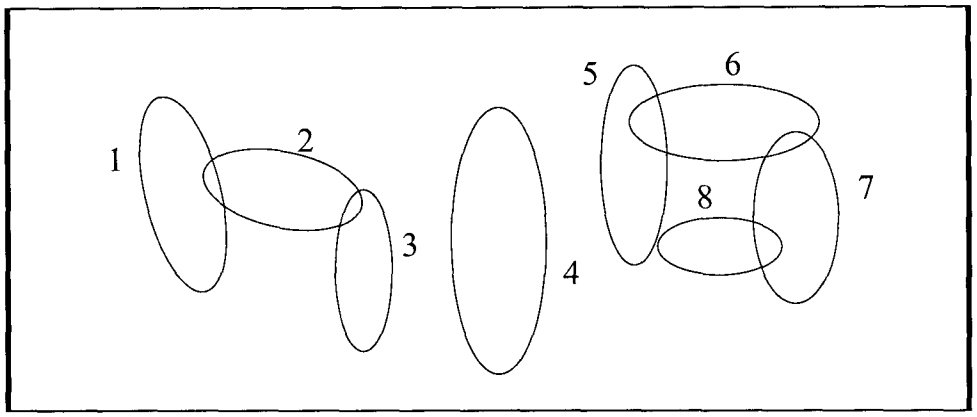


Figure 3.1: Target grouping

order to calculate $\left[Q_k^i\right]_{t_1 t_2}$, we check whether the corresponding targets (t_1 and t_2) are in the same group, If they are, then all the members in that group are considered in calculating $\left[Q_k^i\right]_{t_1 t_2}$, otherwise $\left[Q_k^i\right]_{t_1 t_2} = 0$.

- **Approximation 2:** The IRM is a symmetric matrix, i.e., $\left[Q_k^i\right]_{t_1 t_2} = \left[Q_k^i\right]_{t_2 t_1}$. Hence, we can reduce the computational burden by calculating only the upper or lower triangular entries of Q_k^i.

- **Approximation 3:** A validation matrix is used to reduce the joint association events, by eliminating the ones with negligible probabilities [4]. The validation matrix is defined by

$$\Omega = [\varpi_{jt}] \quad j = 1,\ 2,\ \ldots,\ m; \quad t = 0,\ 1,\ \ldots,\ T \tag{3.27}$$

where $\varpi_{jt} (\in \{0, 1\})$ indicates whether measurement j lies inside the validation region of target t.

In addition to the above simplifications, the computation time can be further reduced by calculating the $J_{z_i}(k)$ values for each sensor in parallel.

3.3 PCRLB for Distributed Tracking

In this thesis, for distributed tracking we assume that LFCs communicate with CFC at each measurement time steps. In section 2.3.4.1, we have shown that the performance of distributed tracking with communication at every time step is equivalent to the centralized tracking. Hence the equation derived for centralized can be still used for the distributed architecture under the assumption considered in this thesis.

An approximate PCRLB for a generalized, decentralized, architecture is shown in the next section.

3.4 PCRLB for Decentralized Tracking

In the decentralized tracking, equations derived for centralized tracking will be used to update the PCRLB whenever a FC gets measurements from its local sensors. However, if a FC gets a tracklet from its neighbor, then we cannot use the equations derived so far, and an approximate PCRLB updating equation for track fusion is given below.

Suppose, the neighbors of FC f are FCs $f_{n_1}, f_{n_2}, \ldots, f_{n_j}$, and their last communication time steps are l_{n_1}, l_{n_2}, ..., l_{n_j}, respectively. The FIMs at last communication time steps are $J^{f_{n_1}}(l_{n_1}), J^{f_{n_2}}(l_{n_2}), \ldots, J^{f_{n_j}}(l_{n_j})$. The FIMs, which are updated using only their local sensor measurements, at current time step k are

$J^{f_{n_1}}(k), J^{f_{n_2}}(k), \ldots, J^{f_{n_j}}(k)$. An approximate PCRLB equation, based on information filter (see section 2.3.4), for track fusion at FC f is:

$$J^f_{\text{fused}}(k) = J^f(k) + \sum_{i=1}^{j} c_i \left(J^{f_{n_i}}(k) - J_X^{f_{n_i}}(k|l_{n_i}) \right) \tag{3.28}$$

where

$$J_X^{f_{n_i}}(k|l_{n_i}) = \left[\Gamma(k, l_{n_i}) + F(k, l_{n_i}) J^{f_{n_i}}(l_{n_i})^{-1} F(k, l_{n_i})' \right]^{-1} \tag{3.29}$$

and c_i takes one if FC f received tracklets from neighbor n_i at time k, and zero otherwise.

3.5 Scaler Performance Measure

Since the PCRLB, which is the primary performance criterion in our optimization, is a matrix, it needs to be converted into a corresponding scalar quantity for ease of operation. Possible scalar performance metrics based on the PCRLB are its trace, determinant or the maximum eigenvalue [37]. The trace of PCRLB (i.e., trace(J^{-1})) and the determinant of the PCRLB (i.e., det(J^{-1})) are proportional to the circumference and the volume of the rectangular region enclosing the minimum achievable covariance ellipsoid, respectively. The maximum eigenvalue of PCRLB (i.e., $\lambda_{\max}(J^{-1})$) is the maximum distance of any point in the minimum achievable covariance ellipsoid from its center.

In case the smallest principal axis shrinks to zero while the uncertainties along the remaining principal axes might remain large, the volume of the uncertainty ellipsoid

is zero. Hence, the sensor manger with determinant of PCRLB as the scaler performance measure may select sensors, which give poor tracking accuracy. In addition, calculation of the determinant or the maximum eigenvalue of a matrix is computationally more expensive than calculating the trace of a matrix. Hence, in this thesis the trace of the PCRLB is used as the scalar performance metric. But our algorithm is not tied to the trace of the PCRLB or any other scalar metric.

Chapter 4

Sensor Array Management for Centralized Tracking

4.1 Problem Description

In this section we present the sensor subset selection problem for multi-target tracking for centralized architecture.

A large number of sensors are already deployed through out the surveillance region in order to monitor the surveillance region, and we assumed that the sensor locations are fixed and known. Multiple, possibly time varying, number of targets are present in the surveillance region. A sample scenario, in which 3 targets are present and each enter the surveillance region at different time steps, is shown in Figure 4.1. Even though large number of sensors available to use at any time, due to physical limitations (such as available communication bandwidth, central fusion center capacity, etc.) only a certain number of sensors can be used at any one time step. Then the problem is to select a subset of sensors such that tracking performance is optimized.

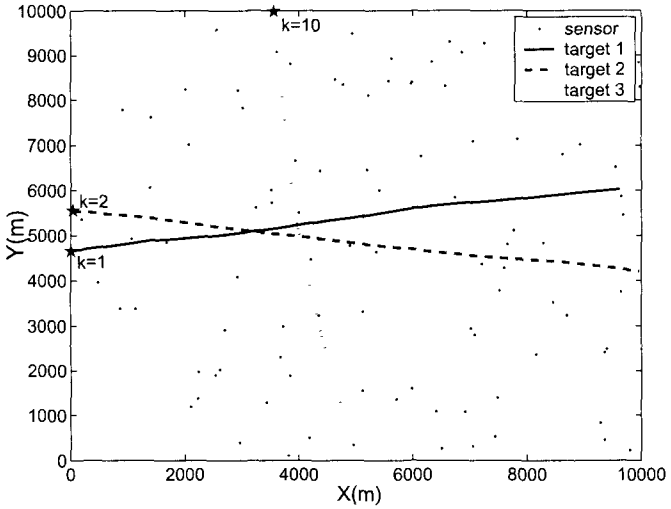


Figure 4.1: A sample scenario

Formulation of the above problem based on PCRLB is given in the next section.

4.2 Problem Formulation

Suppose, a total of N sensors are available in the surveillance region. Out of these N sensors, only n can be utilized at any one time. Measurements are received at time intervals of T_s. A sensor lead time, T_L $(= \rho \times T_s)$, is needed to activate a sensor subset, i.e., the sensor subset that is to be activated at sampling time k has to be selected at or before sampling time $k - \rho$. Furthermore, the active sensor set can only be changed every $T_c = l \times T_s$ $(l \geq 1)$ time interval. Hence, once activated, a particular sensor subset remains active for one or more revisits.

4.2.1 Sensor subset selection for fixed number of targets

In this section we consider the sensor selection problem for a fixed and known (say T) number of targets. The problem is then to select a subset consisting of n sensors

from the N available in order to optimize the tracking performance of the T targets over the activation period of a particular sensor set.

Sensors are selected based on the predictive PCRLB, which is dependent on the predicted target states. In order to minimize the difference between the predictive PCRLB and the true value *given the measurements were available*, we wait until the latest possible sampling time, $k-\rho$ before selecting the sensors that are to be activated at sampling time k. The target states in the time interval $k-\rho$ to $k+l-1$ are then predicted and used to calculate $J_X(k)$ and $J_{z_i}(\tau)$ for $i = 1, 2, \ldots, N$; $\tau = k, (k+1), \ldots, (k+l-1)$. The values of $J_X(k)$ and $J_{z_i}(\tau)$ do not depend on the sensors that are to be selected at sampling time k. Hence, they need to be calculated only once.

4.2.1.1 Formulation 1

Let S_{ik} be the indicator of the state of sensor i at sampling time k that takes value 1 if that sensor is active at that time and 0 otherwise. Because the active sensor set is only changed every l time steps, $S_{ik} = S_{i(k+1)} = \cdots = S_{i(k+l-1)}$. The optimal sensor subset $S^*(k)$ is then given by

$$S^*(k) = \arg\min_{\{S_{ik}\}} \sum_{\tau=k}^{k+l-1} \text{trace}\left\{ \left[J_X(\tau) + \sum_{i=1}^{N} S_{ik} C_{\tau i} J_{z_i}(\tau) \right]^{-1} \right\} \tag{4.1}$$

subject to:

$$\sum_{i=1}^{N} S_{ik} = n \tag{4.2}$$

$$S_{ik} \in \{0,1\}, \quad i = 1, 2, \ldots, N \tag{4.3}$$

where

$$C_{\tau i} = \text{diag}[c_{\tau 1 i} I_{r\times r}, c_{\tau 2 i} I_{r\times r}, \ldots, c_{\tau T i} I_{r\times r}] \tag{4.4}$$

$$c_{\tau t i} = \begin{cases} 1 & \text{if target } t \text{ is within the coverage region of sensor } i \text{ at time step } \tau \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

r is the dimensionality of the state vector of each target[1].

$J_X(k)$ is calculated using the PCRLB at sampling time $k-\rho$, together with the expected measurement information in the time interval $k-\rho$ to k based on the current active sensor set. $J_X(k+1)$, $J_X(k+2), \ldots$ are then calculated using the following recursion

$$J_X(k+1) = \left(\Gamma_k + F_k\Big[J_X(k) + \sum_{i=1}^{N} S_{ik} C_{ki} J_{z_i}(k)\Big]^{-1} F_k'\right)^{-1} \tag{4.6}$$

In the objective function minimized in equation (4.1), equal weights are given to all time steps. However, the accuracy with which we can predict the target states decreases with time. Hence, the accuracy with which we can predict the PCRLBs also decreases with time. Therefore, we may give different weights to each time step. These weights may, for example, be inversely proportional to the prediction interval and also dependent on the process noise covariance.

4.2.1.2 Formulation 2

If the targets are well-separated in the state space, even if they are close in the measurement space, the above formulation gives impressive results. However, if the

[1]The dimensionality is the same purely for brevity, and the general approach is in no way restricted by this assumption.

targets are close in the state space, in addition to minimizing the trace of PCRLB, we also attempt to reduce the error covariance in the direction of the vector that connects the state estimates of each two targets. This approach helps to reduce the possibility of track swapping.

The vector that connects the predicted states of target t_1 and t_2 at time τ is given by

$$Y_{t_1t_2}(\tau) = \hat{x}^{t_1}(\tau|k-\rho) - \hat{x}^{t_2}(\tau|k-\rho) \tag{4.7}$$

where $\hat{x}^t(\tau|k-\rho)$ is the predicted state of target t at time τ, given estimate at time $k-\rho$.

If the PCRLB at time τ, $B(\tau)$, is given by

$$B(\tau) = \left[J_X(\tau) + \sum_{i=1}^{N} S_{i\tau} C_{\tau i} J_{z_i}(\tau) \right]^{-1} \tag{4.8}$$

then, $b_{t_1t_2}(\tau)'[B(\tau)]_{t_1t_1} b_{t_1t_2}(\tau)$ gives the (mean square) error bound in estimating target t_1's state in the direction of target t_2 (at time τ). In the above, $b_{t_1t_2}(\tau)$ is a unit vector along $Y_{t_1t_2}(\tau)$ and $[B(\tau)]_{tt}$ is the t-th block diagonal matrix of $B(\tau)$ that gives the covariance of target t.

Now, we can modify the optimization as follows

$$S^*(k) = \arg\min_{\{S_{ik}\}} \sum_{\tau=k}^{k+l-1} \left\{ \sum_{t_1=1}^{T} \left[w_{t_1} \text{ trace } \left([B(\tau)]_{t_1t_1} \right) + \sum_{\substack{t_2=1 \\ t_2 \neq t_1}}^{T} w_{t_1t_2} \; b'_{t_1t_2} [B(\tau)]_{t_1t_1} b_{t_1t_2} \right] \right\} \tag{4.9}$$

where

$$w_{t_1t_2} = \min\left\{1, \frac{g \times \left(\sqrt{b'_{t_1t_2}[J_X(\tau)^{-1}]_{t_1t_1}b_{t_1t_2}} + \sqrt{b'_{t_1t_2}[J_X(\tau)^{-1}]_{t_2t_2}b_{t_1t_2}}\right)}{\| Y_{t_1t_2}(\tau) \|}\right\} \quad (4.10)$$

$$w_{t_1} = 1 - \max\{w_{t_1t_2} | t_2 \in \{1, 2, \ldots, T\}, t_2 \neq t_1\} \quad (4.11)$$

g is the number of standard deviations of the measurement gate. The constraints are again given by (4.2) and (4.3).

The weights in the objective function, w_{t_1} and $w_{t_1t_2}$, are selected based on the distances between the targets and the predicted covariances. If there are no targets in close proximity to target t_1, then $w_{t_1t_2}$ approaches zero and w_{t_1} approaches one. Conversely, if there is a target, t_2, close to target t_1, then w_{t_1} approaches zero and $w_{t_1t_2}$ approaches one.

If we are only interested in estimating the locations of the targets, we need to consider only the PCRLB terms relating to the target locations in the above equations.

4.2.2 Sensor subset selection for a varying number of targets

In the previous section we assumed that the total number of targets in the surveillance region is fixed and known. However, in most realistic scenarios the above assumption is not valid. In this section we present a sensor subset selection algorithm for a scenario in which the total number of targets in the surveillance region is not known and may vary with time, i.e., either some new targets may enter or some existing targets may leave the surveillance region. The problem is then to select a sensor subset in order to: (i): optimize the tracking performance of existing targets and (ii): quickly identify new targets.

The following assumptions are made:

- New targets appear by passing through the perimeter of the surveillance region (i.e. they do not "pop-up" in the interior of the surveillance region).
- The probability mass function of the number of new targets, $\mu_{NT}(\tau)$, at any one time has a Poisson distribution with mean $\lambda_b V_p$, and given by

$$\mu_{NT}(\tau) = \frac{e^{-\lambda_b V_p}(\lambda_b V_p)^{\tau}}{\tau!} \tag{4.12}$$

  where τ is the number of new targets, λ_b is the spatial distribution of the new targets and V_p is the volume of the perimeter.
- The targets die (disappear) only when they leave the surveillance region.

The sensor selection algorithm presented herein can, with slight modification, cater for a different set of assumptions.

When a target leaves the surveillance region its track is deleted and the PCRLB is adjusted by removing the entries that relate to this target. A bi-criterion optimization approach with two objectives, one to accurately track existing targets and the other to quickly identify new ones, is then used to select the sensors.

The first objective is to minimize the PCRLB (see (4.1)) for established (i.e., detected) targets that are still within the surveillance region. If we have T targets in the surveillance region at sampling time $k-1$, then the prior information matrix of those targets at sampling time k, $J_{X_{old}}(k)$, is a $Tr \times Tr$ matrix, and can be calculated using the PCRLB at sampling time $k-1$ and the state transition matrices. If a new target is initialized at sampling time k, then the updated prior information matrix is

augmented as follows

$$J_X(k) = \begin{bmatrix} J_{X_{old}}(k) & 0 \\ 0 & J_{X_{new}} \end{bmatrix} \tag{4.13}$$

where $J_{X_{new}}$ is the prior information matrix of the new target, initialized as: $J_{X_{new}} = \text{diag}\{1/V_{x1}^2, 1/V_{x2}^2, \ldots, 1/V_{xr}^2\}$. V_{xi} is the surveillance volume, the size of the range of possible values, of the i-th component of the state vector. If a target, t leaves the surveillance region, then $J_X(k)$ is adjusted by eliminating the rows and columns corresponding to target t. The adjustments above are modified in an obvious manner if more than one target enters or leaves the surveillance region at any one time.

The second objective is to maximize the probability of detecting new targets. First, let us assume that only one new target may enter the surveillance region at any one time. To find the probability of detecting this new target, P_{new} particles, each of which represents a possible state of the new target, are uniformly distributed along the perimeter of the surveillance region[2]. We note that we could distribute the particles non-uniformly if certain points on the perimeter are more likely entry points for new targets (e.g., based on some prior threat assessment). After selecting a point on the perimeter, we have to determine the distance that the target could potentially have travelled inside the surveillance region. If the velocity of the new target in the direction perpendicular to the perimeter is uniformly distributed between 0 and $V_{\max}$, then the probability density function of the distance of the new target's position from

[2]If new targets can appear anywhere in the surveillance region, particles should be placed uniformly all over the surveillance region.

the perimeter, q, is given by:

$$\begin{aligned} p(q) &= \int_0^{V_{\max}} p(q|v)\ p(v)dv \\ &= \int_{q/T_s}^{V_{\max}} \frac{1}{vT_s}\frac{1}{V_{\max}}dv \\ &= \frac{1}{V_{\max}T_s}\ln\left(\frac{V_{\max}T_s}{q}\right) \end{aligned} \tag{4.14}$$

where T_s is the sampling time. The velocity components of the particles are uniformly distributed between the known lower and upper velocity limits. The probability of detection of the new target at sampling time k, p_D^k, is given by

$$p_D^k = 1 - \frac{1}{P_{\text{new}}}\sum_{j=1}^{P_{\text{new}}}\prod_{i=1}^{N}(1 - S_{ik}P_D(i,j)) \tag{4.15}$$

where $P_D(i,j)$ is the probability of detecting, at sensor i, a target whose state is given by particle j. In general this probability of detection depends on both the sensor state and the target state.

We can improve the performance even further by considering the possibility that a new target had entered the surveillance region at sampling time $k-1$ but had not been detected at that time. The probability of not detecting a new target at sampling time $k-1$ is given by

$$p_M^{k-1} = \frac{1}{P_{\text{new}}}\sum_{j=1}^{P_{\text{new}}}\prod_{i=1}^{N}(1 - S_{i(k-1)}P_D(i,j)) \tag{4.16}$$

In order to incorporate the missed target, while P_{new} particles are used for the new target that enter the region at sampling time k, $p_M^{k-1} \times P_{\text{new}}$ particles are used for the new target that was not detected at sampling time $k-1$. The particles representing

the new target that enters the region at sampling time k are uniformly distributed along the perimeter of the surveillance region. However, the particles representing the missed target are distributed according to the probability of missing a new target at each point on the perimeter of the surveillance region. After selecting a point on the perimeter, the probability density function of the depth at which the target could now lie is given by

$$\begin{aligned} p(q) &= \int_{q/2T_s}^{\min(V_{\max},\, q/T_s)} \frac{1}{vT_s}\frac{1}{V_{\max}}dv \\ &= \begin{cases} \frac{1}{V_{\max}T_s}\ln(2) & \text{if } q < V_{\max}T_s \\ \frac{1}{V_{\max}T_s}\ln\left(\frac{2V_{\max}T_s}{q}\right) & \text{else if } q < 2V_{\max}T_s \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{4.17}$$

Now, the number of particles representing both a new detected target and a missed target (that entered on the previous time step) is $\bar{P}_{\text{new}} = (1 + p_M^{k-1}) \times P_{\text{new}}$. Hence, the second objective function now becomes

$$OBJ(2) = \frac{1}{\bar{P}_{\text{new}}} \sum_{j=1}^{\bar{P}_{\text{new}}} \prod_{i=1}^{N} (1 - S_{ik} P_D(i,j)) \tag{4.18}$$

In the above equations we have considered only one new target. However, if we consider many new targets, then under the assumption that the new targets are moving independently, the above objective function can be modified as follows

$$OBJ(2) = \left(\sum_{t_n=1}^{T_N} \mu_{NT}(t_n) \times t_n \right) \times \frac{1}{\bar{P}_{\text{new}}} \sum_{j=1}^{\bar{P}_{\text{new}}} \prod_{i=1}^{N} (1 - S_{ik} P_D(i,j)) \tag{4.19}$$

where T_N is the maximum number of new targets that can enter the surveillance region at any one time.

Finally, the bi-criterion optimization problem is given as follows

$$S^*(k) = \arg\min_{\{S_{ik}\}} \left\{ \left(\sum_{\tau=k}^{k+l-1} \text{trace} \left\{ \left[J_X(\tau) + \sum_{i=1}^{N} S_{ik} C_{\tau i} J_{z_i}(\tau) \right]^{-1} \right\} \right) + w_b \left(\sum_{t_n=1}^{T_N} \mu_{NT}(t_n) \times t_n \right) \times \frac{1}{\bar{P}_{\text{new}}} \sum_{j=1}^{\bar{P}_{\text{new}}} \prod_{i=1}^{N} (1 - S_{ik} P_D(i,j)) \right\} \quad (4.20)$$

In the above, w_b is the weight applied to target birth function. This weight both balances the units and specifies the relative importance of the two objectives.

In the above equations, we have considered the effect of missed detections only at sampling time $k-1$. The performance can be improved by considering the effect of missed detections at earlier time steps ($k-2$, $k-3$, etc.) as well. However, the discounted effects do indeed become negligible very quickly, and our approach is a reasonable approximation.

4.3 Solution Technique

The optimization problems formulated in the previous two sections are all NP-hard [35, 56]. Enumerating every possibility in order to find the optimal solution is feasible only in small scale problems. Complete enumeration is not viable when the number of possibilities is very large. However, for large scale problems, an iterative local search technique is used to find a near optimal solution [43]. This is given as follows:

- **Step 1**: Select an initial solution as follows:
  - Select the sensor that gives optimal performance when only one sensor is used.

- Select a second sensor that results in optimal performance when only two sensors are used: including the first selected sensor.
- Continue adding sensors until we have a total of n sensors.

At each stage, a complete enumeration is performed to determine which sensor should be added to the selection.

- **Step 2**: Search for better solutions in the neighborhood (until we reach some pre-specified time limit). Initially, form the neighborhood by swapping only one sensor from the current solution. If the current solution is optimal in the above neighborhood, then form a new neighborhood by swapping two sensors. Keep on increasing the number of sensors that are swapped for forming the neighborhood until the time limit is reached.

The local optimum of a certain neighborhood can be obtained quickly by swapping the sensors in the order of their rank, where the rank of each sensor is based on its individual performance. A detailed explanation of local search technique is given in Appendix B.

4.4 Simulation

In the following scenarios, we assume that the targets move according to independent constant velocity models [5]. The state of target t at time k is given by $x_k^t = \left[\xi_k^t,\ \dot{\xi}_k^t,\ \eta_k^t,\ \dot{\eta}_k^t\right]'$, where ξ_k^t and η_k^t are the x and y coordinates of target t, respectively, and $\dot{\xi}_k^t$ and $\dot{\eta}_k^t$ are the velocities of target t in the corresponding directions, respectively. The sensors have independent measurement processes. The instantaneous probability of detection of each target at each sensor is given by a

Swerling I model [4], i.e.,

$$P_D^t = P_F^{\frac{R_t^2}{R_t^2+S_0R_0^2}} \tag{4.21}$$

where R_t is the target range from the sensor and S_0 is the signal to noise ratio at a nominal range R_0.

In simulations, we use either an array of bearings-only sensors or an array of range-only sensors. The j-th measurement at sensor i is given by

$$z_k(j,i) = \begin{cases} \left.\begin{array}{ll} \tan^{-1}\left(\frac{\eta_k^t-\eta_s^i}{\xi_k^t-\xi_s^i}\right)+\omega_k^i(j) & \text{if bearing-only} \\ \sqrt{(\xi_k^t-\xi_s^i)^2+(\eta_k^t-\eta_s^i)^2}+\omega_k^i(j) & \text{if range-only} \end{array}\right\} & \text{if target } t \text{ generated} \\ v_k^i(j) & \text{if false alarm} \end{cases} \tag{4.22}$$

where $(\xi_s^i,\ \eta_s^i)$ is the i-th sensor location, $\omega_k^i(j) \sim \mathcal{N}(0,\sigma^2)$ and

$$v_k^i(j) \sim \begin{cases} U[-\pi,\pi] & \text{if bearing-only} \\ U[0,R_c] & \text{if range-only} \end{cases} \tag{4.23}$$

R_c is the coverage range of the sensor.

We also include a nominal minimum detection range of 1m, inside which the target cannot be detected. This is necessary because, otherwise, as shown in [38], the bearing-only PCRLB is noninformative and approaches zero as the number of Monte Carlo samples tends to infinity. For brevity, the minimum detection range is not included in the notation.

Because the measurement equation is nonlinear, we use a stacked single particle filter to track the targets. The reader is referred to [41, 48] for details of the approach.

4.4.1 Measurement contribution to FIM

For both types of sensor, the (t_1, t_2)th block matrix of $J_z(k)$, $\left[J_z(k)\right]_{t_1 t_2}$, is given by

$$\left[J_z(k)\right]_{t_1 t_2} = \begin{pmatrix} j_{11}^{t_1 t_2} & 0 & j_{13}^{t_1 t_2} & 0 \\ 0 & 0 & 0 & 0 \\ j_{31}^{t_1 t_2} & 0 & j_{33}^{t_1 t_2} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.24}$$

For bearings-only sensors, $j_{11}^{t_1 t_2}, j_{13}^{t_1 t_2}, j_{31}^{t_1 t_2}$ and $j_{33}^{t_1 t_2}$ are then given by

$$\begin{aligned}
j_{11}^{t_1 t_2} &= \frac{1}{\sigma^2} \sum_{i=1}^{n} \mathbb{E}\left[\left[Q_k^i\right]_{t_1 t_2} \frac{(\eta_k^{t_1} - \eta_s^i)(\eta_k^{t_2} - \eta_s^i)}{\left((\xi_k^{t_1} - \xi_s^i)^2 + (\eta_k^{t_1} - \eta_s^i)^2\right)\left((\xi_k^{t_2} - \xi_s^i)^2 + (\eta_k^{t_2} - \eta_s^i)^2\right)} \right] \\
j_{13}^{t_1 t_2} &= -\frac{1}{\sigma^2} \sum_{i=1}^{n} \mathbb{E}\left[\left[Q_k^i\right]_{t_1 t_2} \frac{(\eta_k^{t_1} - \eta_s^i)(\xi_k^{t_2} - \xi_s^i)}{\left((\xi_k^{t_1} - \xi_s^i)^2 + (\eta_k^{t_1} - \eta_s^i)^2\right)\left((\xi_k^{t_2} - \xi_s^i)^2 + (\eta_k^{t_2} - \eta_s^i)^2\right)} \right] \\
j_{31}^{t_1 t_2} &= -\frac{1}{\sigma^2} \sum_{i=1}^{n} \mathbb{E}\left[\left[Q_k^i\right]_{t_1 t_2} \frac{(\xi_k^{t_1} - \xi_s^i)(\eta_k^{t_2} - \eta_s^i)}{\left((\xi_k^{t_1} - \xi_s^i)^2 + (\eta_k^{t_1} - \eta_s^i)^2\right)\left((\xi_k^{t_2} - \xi_s^i)^2 + (\eta_k^{t_2} - \eta_s^i)^2\right)} \right] \\
j_{33}^{t_1 t_2} &= \frac{1}{\sigma^2} \sum_{i=1}^{n} \mathbb{E}\left[\left[Q_k^i\right]_{t_1 t_2} \frac{(\xi_k^{t_1} - \xi_s^i)(\xi_k^{t_2} - \xi_s^i)}{\left((\xi_k^{t_1} - \xi_s^i)^2 + (\eta_k^{t_1} - \eta_s^i)^2\right)\left((\xi_k^{t_2} - \xi_s^i)^2 + (\eta_k^{t_2} - \eta_s^i)^2\right)} \right]
\end{aligned} \tag{4.25}$$

and for range-only sensors

$$
\begin{aligned}
j_{11}^{t_1t_2} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbb{E}\left[\left[Q_k^i\right]_{t_1t_2}\frac{(\xi_k^{t_1}-\xi_s^i)(\xi_k^{t_2}-\xi_s^i)}{\sqrt{\left((\xi_k^{t_1}-\xi_s^i)^2+(\eta_k^{t_1}-\eta_s^i)^2\right)\left((\xi_k^{t_2}-\xi_s^i)^2+(\eta_k^{t_2}-\eta_s^i)^2\right)}}\right]\\
j_{13}^{t_1t_2} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbb{E}\left[\left[Q_k^i\right]_{t_1t_2}\frac{(\xi_k^{t_1}-\xi_s^i)(\eta_k^{t_2}-\eta_s^i)}{\sqrt{\left((\xi_k^{t_1}-\xi_s^i)^2+(\eta_k^{t_1}-\eta_s^i)^2\right)\left((\xi_k^{t_2}-\xi_s^i)^2+(\eta_k^{t_2}-\eta_s^i)^2\right)}}\right]\\
j_{31}^{t_1t_2} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbb{E}\left[\left[Q_k^i\right]_{t_1t_2}\frac{(\eta_k^{t_1}-\eta_s^i)(\xi_k^{t_2}-\xi_s^i)}{\sqrt{\left((\xi_k^{t_1}-\xi_s^i)^2+(\eta_k^{t_1}-\eta_s^i)^2\right)\left((\xi_k^{t_2}-\xi_s^i)^2+(\eta_k^{t_2}-\eta_s^i)^2\right)}}\right]\\
j_{33}^{t_1t_2} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbb{E}\left[\left[Q_k^i\right]_{t_1t_2}\frac{(\eta_k^{t_1}-\eta_s^i)(\eta_k^{t_2}-\eta_s^i)}{\sqrt{\left((\xi_k^{t_1}-\xi_s^i)^2+(\eta_k^{t_1}-\eta_s^i)^2\right)\left((\xi_k^{t_2}-\xi_s^i)^2+(\eta_k^{t_2}-\eta_s^i)^2\right)}}\right]
\end{aligned}
\tag{4.26}
$$

4.4.2 Simulation results

4.4.2.1 Fixed number of targets

In order to test the performance of the formulations presented in section 4.2.1, the following scenario is considered. A total of 15 sensors are in the surveillance region, but only two of them can be used at any one time. Measurements are available at 1 minute intervals. The active sensor subset can be modified at 1 minute intervals (i.e., at every revisit). The sensors are selected with a lead time of $T_L = 1$ minute, $S_0R_0^2 = 3.2 \times 10^9\,\mathrm{m}^2$, $P_F = 10^{-3}$. The particle filter uses 1000 sample points.

We first consider a scenario with a known number (= 2) of targets. We use bearing-only sensors, with the measurement error of target generated measurements set at 0.1 radians (= 5.7°). The measurement contribution to the PCRLB decreases with range because of both increasing bearings cross-section (making targets unobservable at long ranges) and the range dependent P_D. Therefore Formulation 1 (defined in

Section 4.2.1) is likely to select sensors in close proximity to the targets, irrespective of the affect this has on our ability to resolve measurement association ambiguities. Hence results can be expected to be similar to those obtained using the formulation in [74] where the measurement origin uncertainty is neglected, and this was indeed the case. A better comparison is obtained when using range-only sensors (with a 40m range error standard deviation). The root mean square error (RMSE) values of the state estimates based on 100 Monte Carlo runs are shown in Table 4.1.

Table 4.1: RMSE values of state estimates when ignoring and accounting for measurement origin uncertainty

Sensor selection strategy	Position RMSE (m)	
	Target 1	Target 2
Neglecting measurement origin uncertainty	89.68	86.54
Accounting for measurement origin uncertainty (Formulation 1)	75.11	75.59

Comparison of the performances of Formulations 1 and 2, in a scenario with two closely-spaced (but, resolved) targets is considered next. Bearing-only sensors are again used. The sensors selected at sampling time $k = 1$ under these two formulations are shown in Figure 4.2(a) and 4.2(b) respectively. The particle distributions obtained subsequent to resampling are then shown in Figures 4.3(a) and 4.3(b), respectively. We note that Formulation 2 reduces the association ambiguity, with the particle distributions relating to these two targets well separated after resampling. As a result, association ambiguities and estimation errors will be reduced at subsequent sampling times. Indeed, comparisons (based on 100 Monte Carlo runs each of 5 time steps) show that Formulation 2 significantly improves performance in tracking both targets (see Table 4.2).

Table 4.2: RMSE values of state estimates using formulations 1 and 2

Sensor selection strategy	Position RMSE (m)	
	Target 1	Target 2
Formulation 1	33.393	33.053
Formulation 2	25.192	23.029

4.4.2.2 Variable number of targets

We now analyze the performance of the formulation that allows for a time-varying number of targets (see section 4.2.2). In the focal scenario a total of 100 sensors are in the surveillance region, but only 12 of them can be used at any one time step. Sensors give the target bearing (with a measurement error standard deviation of 0.05 radians ($\approx 3^o$) for target generated measurements). Measurements are available at 30 seconds intervals. The active sensor subset is changed at 30 seconds intervals (i.e., again at every revisit). The sensors are selected with a lead time of $T_L = 30$ seconds. The sensor field of view $V = 2\pi$ radians, $S_0 R_0^2 = 3.2 \times 10^7\,\mathrm{m}^2$, $P_F = 10^{-3}$, $\lambda = 0.004\,/\mathrm{radian}$ and $\lambda_b = 0.00001\,/\mathrm{m}$. The surveillance region is $5000\,\mathrm{m} \times 5000\,\mathrm{m}$. The particle filter uses 5000 sample points. The target trajectories and sensor positions are shown in Figure 4.4. There are three targets, and they enter and leave the surveillance region at different times.

Figure 4.3 shows the sensors selected at time steps 13 to 18. At time step 13 there are two established targets (shown in pink and red) and one new target (in green), that has just entered the surveillance region. Because there was no information available regarding the new target when the sensor selection was performed, 9 sensors are assigned to cover the established targets and 3 sensors are assigned in an attempt to detect any new ones. Note that the sensors assigned to the established targets can also cover some parts of the perimeter through which new targets can enter. After

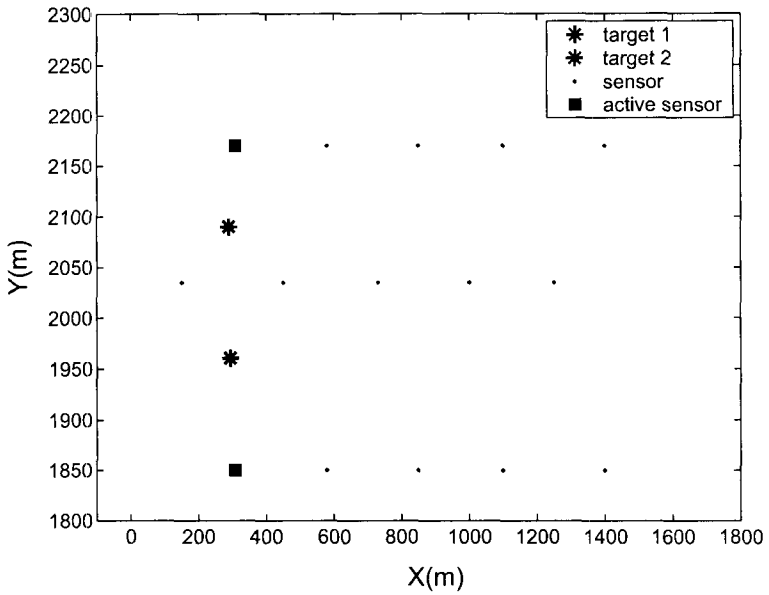
detecting the new target (at time step 14) a greater number of sensors are assigned to this target than to each of the other targets because the prior information regarding the state of this target is minimal.

However, at time steps 15 and 16, the number of sensors that are assigned to the new target is reduced in order to balance the estimation uncertainties of all targets. Target 1 leaves the surveillance region at time step 17. However, 2 sensors were assigned to that target at that time step, as we did not know with certainty that the target was about to leave the surveillance region. At time step 17, the tracker deletes the track corresponding to target 1, and at time step 18 the sensors that would have been assigned to target 1 return to the surveillance task.

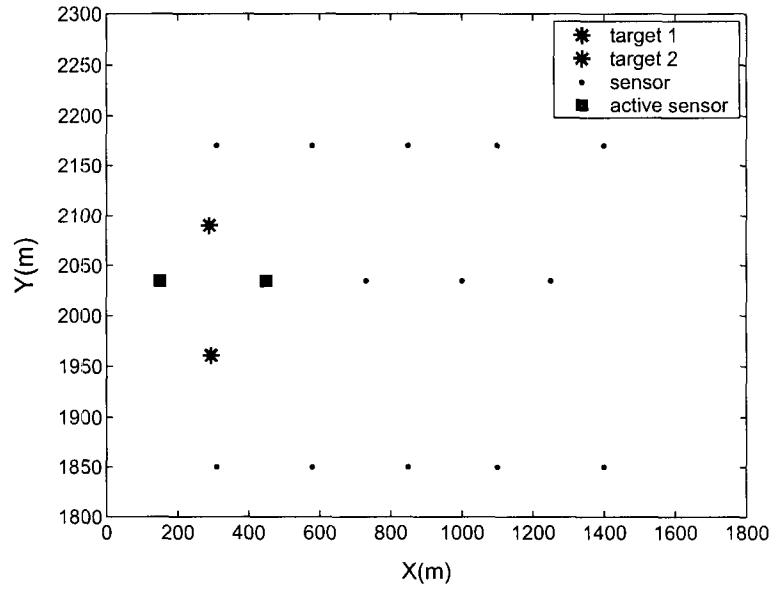
In the above scenario, an average of 0.08 time steps are required to detect each new target. However when using Formulation 2 (that does not task assets specifically for surveillance: again see Section 4.2.1), an average of 1.4 time steps are required before a new target track is established. The improvement in performance (of the formulation in Section 4.2.2) would have been even more significant had the surveillance region been larger, in which case scheduling some sensors to perform surveillance tasks would have been of even more critical importance. Indeed, in Figures 4.4 and 4.5 we have a large surveillance region (10000m$\times$10000m) with 400 sensors. The sensors selected using Formulation 2 and the formulation that allows for a time varying number of targets are shown in Figures 4.4 and 4.5 respectively. In the first case (Figure 4.4), the second target is never detected, because all of the active sensors are being used to track the established target, and the second target has entered the surveillance region through a point on the perimeter that is not within the coverage region of any active sensor.

4.4.2.3 Considering the effect of missed targets

We now analyze the effect of taking into consideration the possibility that a target could have entered the surveillance region on the previous time step, but not been detected at that time. We consider a scenario in which there is no established target in the surveillance region and only three sensors can be used at any one time. In Figure 4.6 we show the sensors selected at time steps $k-1$ and k when we do not give consideration to the possibility that targets could have been entered the surveillance region at time step $k-1$ and not been detected at that time. However, in Figure 4.7 we consider the effect (on the surveillance strategy) of possible missed detections. In Figure 4.6, the sensors selected at time step k are independent of the sensors selected at time step $k-1$. However, in Figure 4.7 the sensors selected at time step k attempt to minimize the probability of again missing any target that could have entered the surveillance region on the previous time step, by prioritizing regions that were not covered well at time step $k-1$.

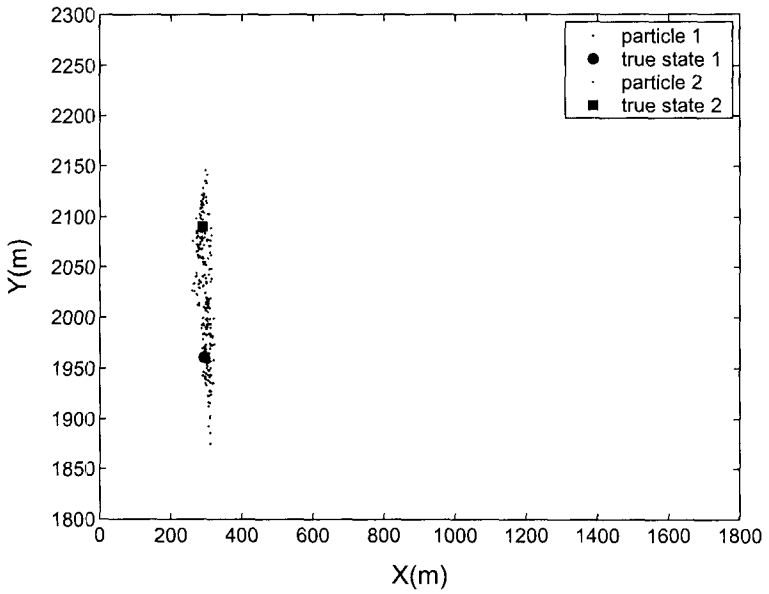


(a) Formulation 1

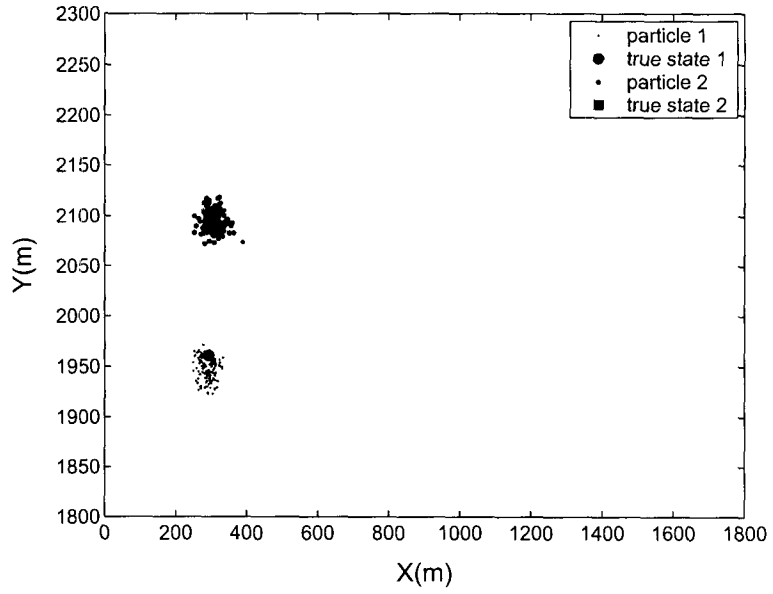


(b) Formulation 2

Figure 4.2: Sensors selected at sampling time $k = 1$



(a) Formulation 1



(b) Formulation 2

Figure 4.3: Resampled particles at sampling time $k = 1$

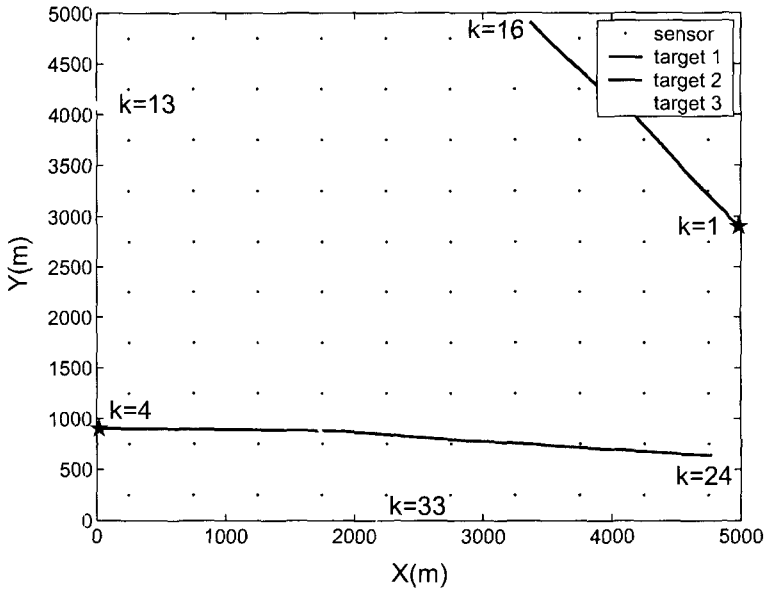
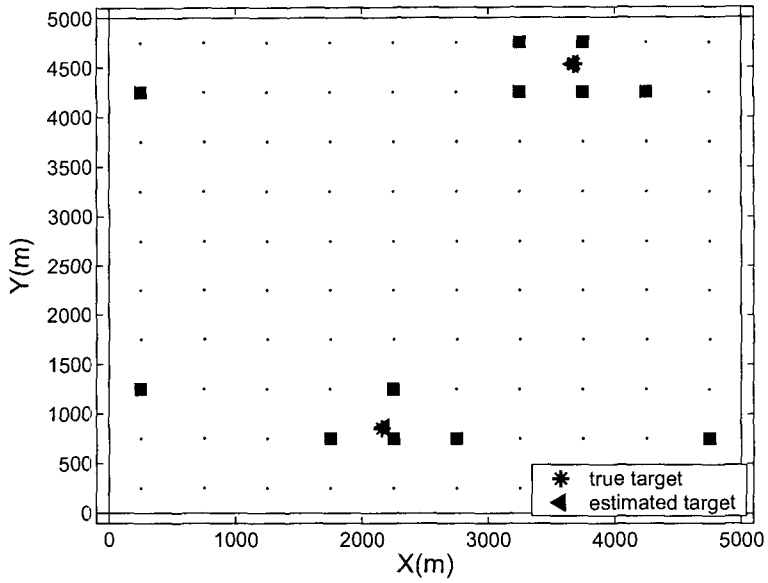
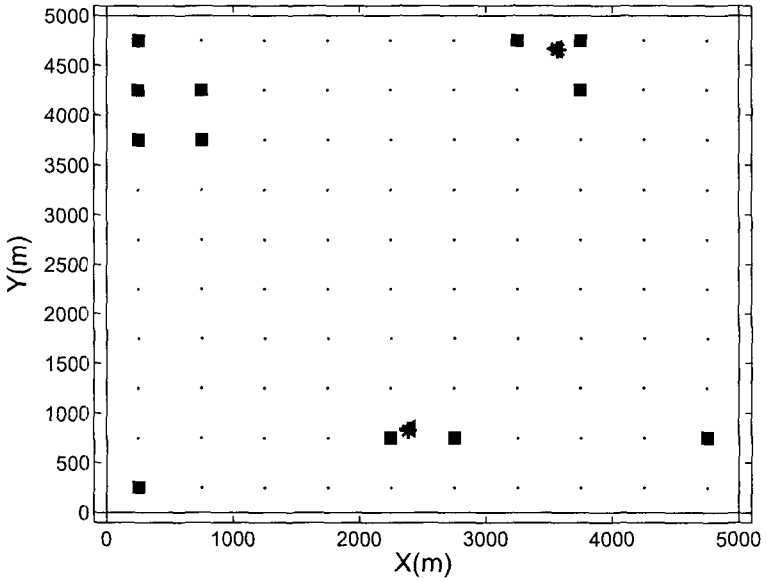


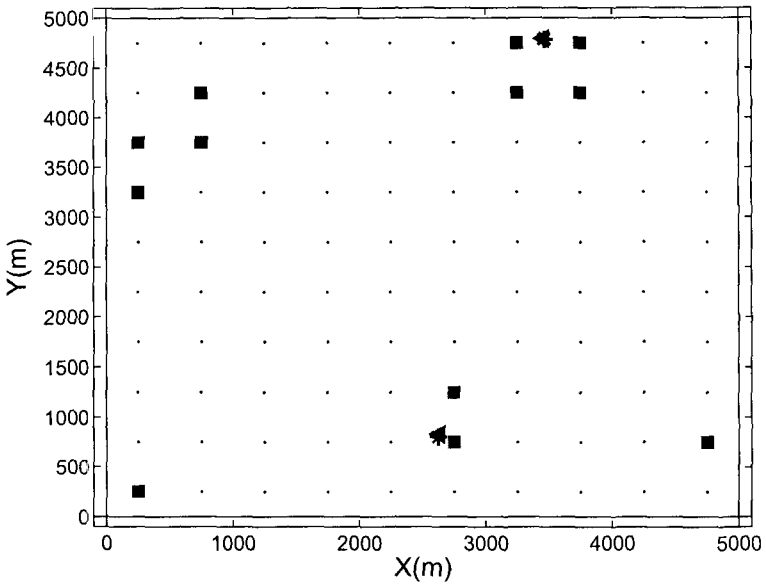
Figure 4.4: Target trajectories (with entering and leaving times) and sensor positions



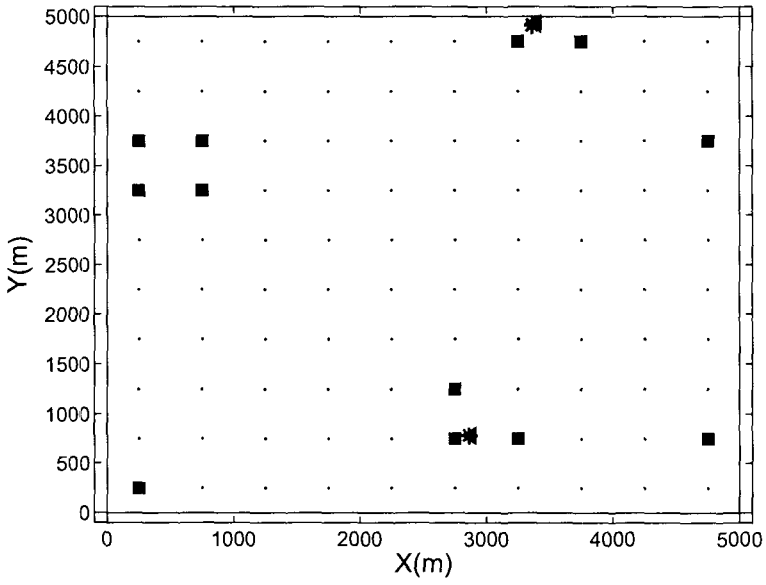
(a) at time $k = 13$



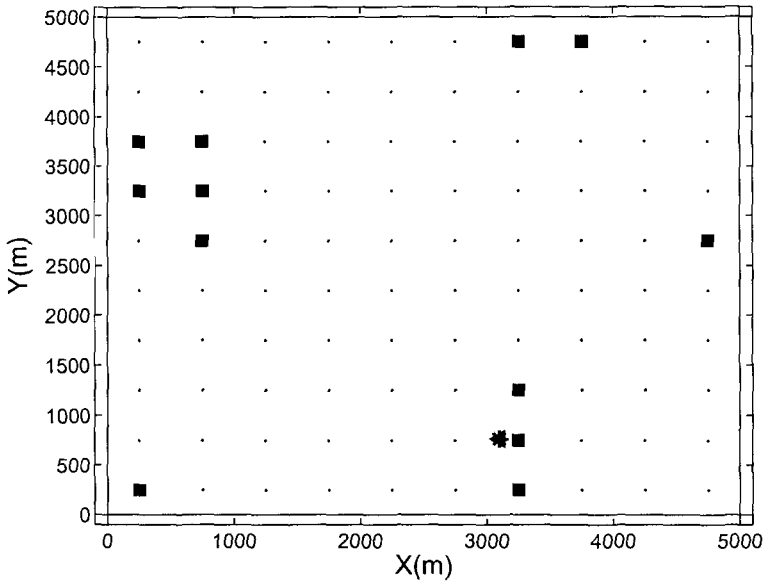
(b) at time $k = 14$



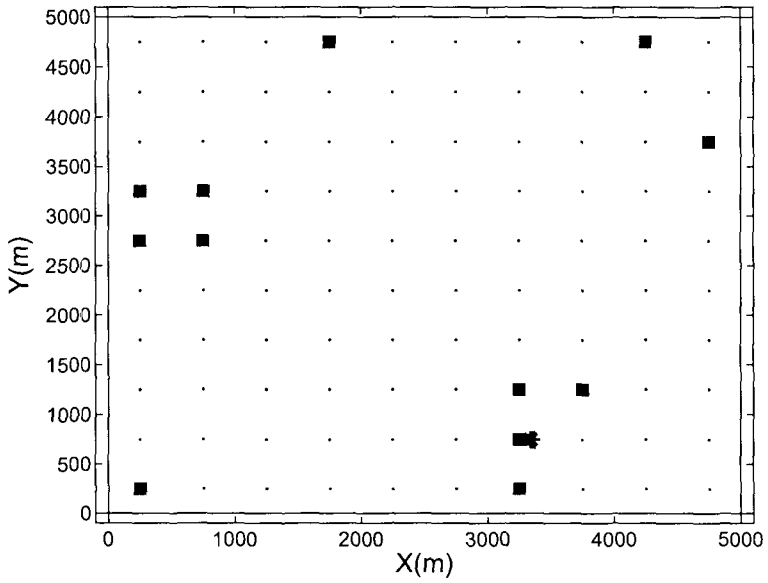
(c) at time $k = 15$



(d) at time $k = 16$

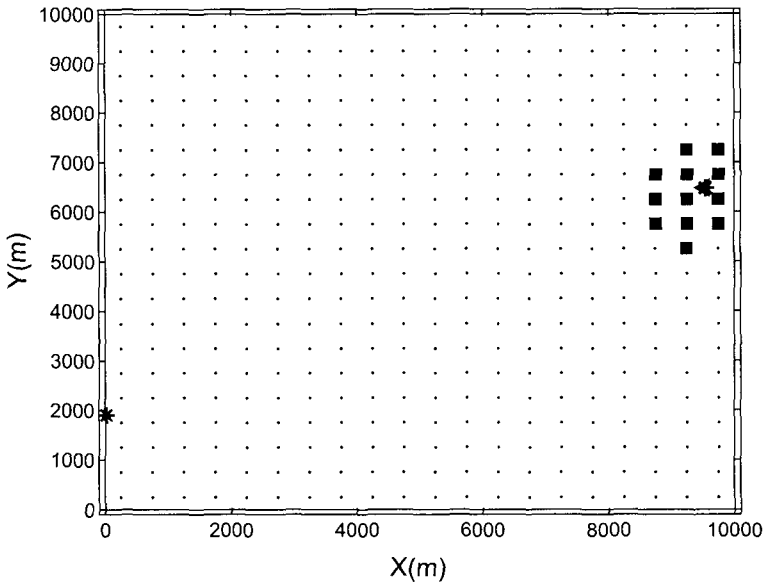


(e) at time $k = 17$

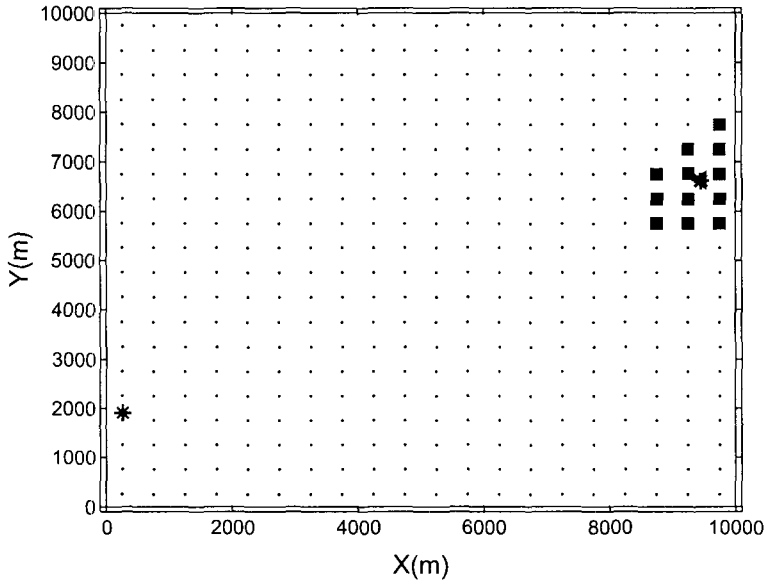


(f) at time $k = 18$

Figure 4.3: Sensors selected in a scenario with a 5000m×5000m surveillance region and 100 sensors

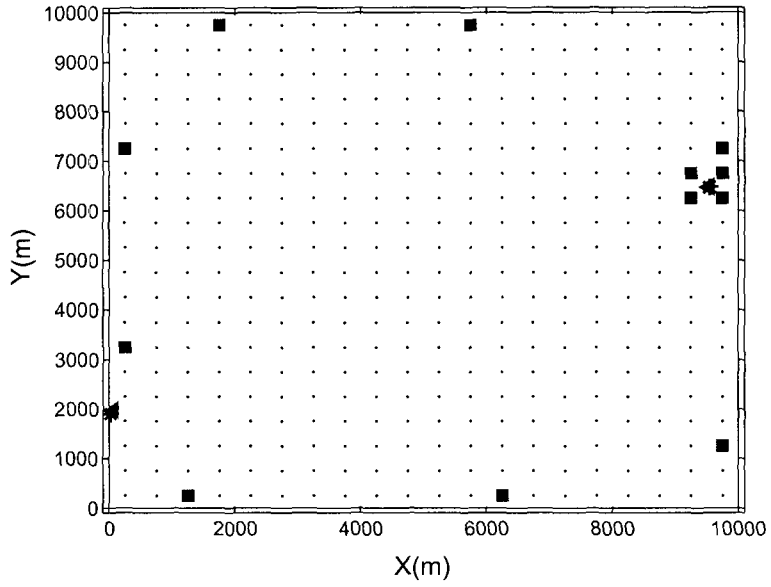


(a) at time $k = 4$

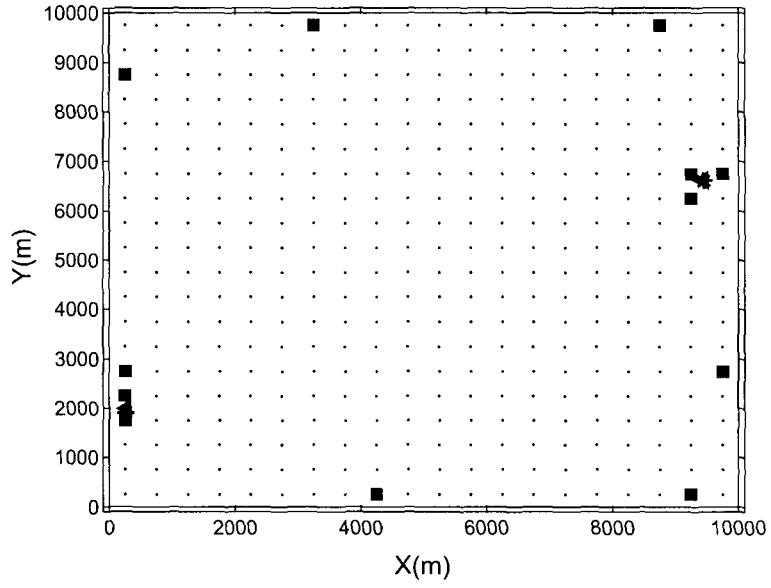


(b) at time $k = 5$

Figure 4.4: Sensors selected using Formulation 2 of Section 4.2.1, in a scenario with a 10000m×10000m surveillance region and 400 sensors

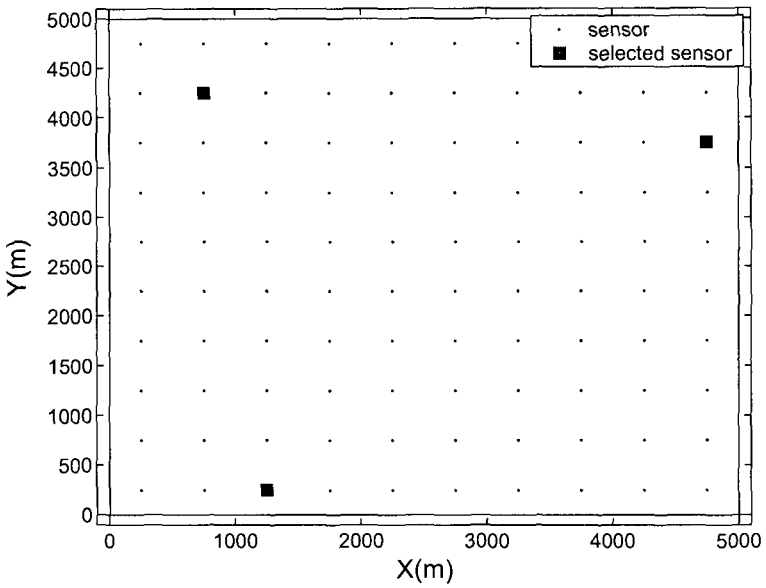


(a) at time $k = 4$

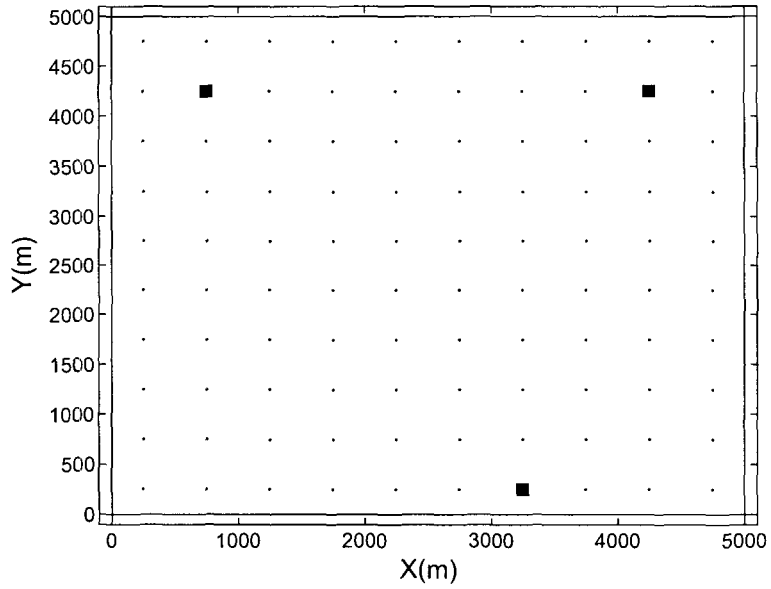


(b) at time $k = 5$

Figure 4.5: Sensors selected using the formulation that allows for a time varying number of targets (see Section 4.2.2), in a scenario with 10000m×10000m surveillance region and 400 sensors

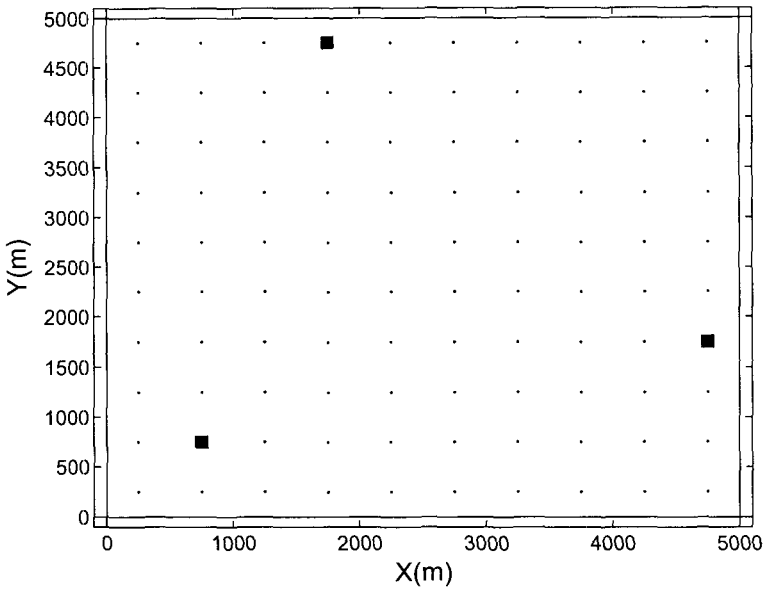


(a) at first sampling time

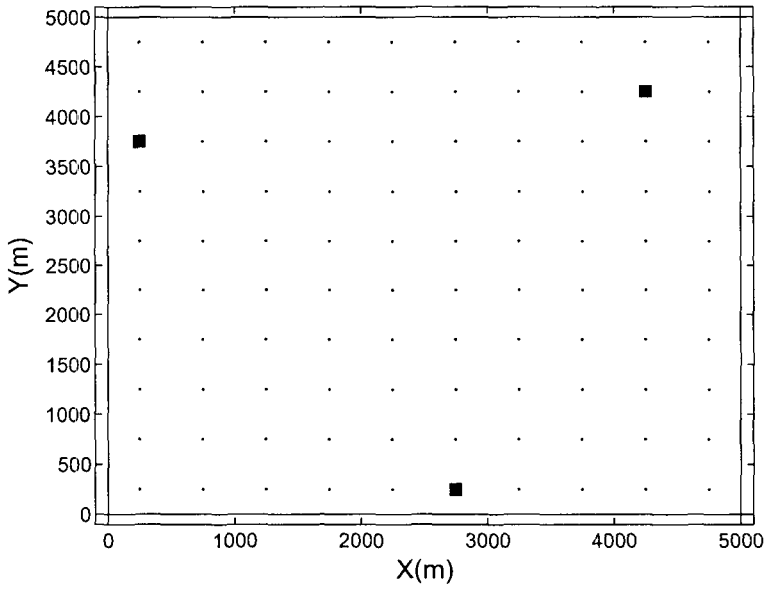


(b) at next sampling time

Figure 4.6: Sensors selected without considering missed targets



(a) at first sampling time



(b) at next sampling time

Figure 4.7: Sensors selected by considering the possibility of missed targets

Chapter 5

Sensor Array Management for Distributed Tracking

In the previous chapter we discussed the sensor management algorithms for centralized architecture, and in this chapter we extend that work for a distributed tracking system that has a hierarchical level architecture with full feedback, as shown in Figure 2.6. A sample scenario with 3 targets, each enter the surveillance region at different time, 100 sensors and 4 FCs is shown in Figure 5.1. The time steps at which targets enter the surveillance region are shown next to the initial positions ($\star$).

5.1 Problem Description

There are M LFCs and N sensors, and we assume that the number of LFCs and sensors, and their locations are fixed and known. Each LFC can handle a certain maximum number of sensors, n_j, because of physical limitations [11]. Sensors transmit their measurements to their LFCs through the allocated frequency channels. The available number of channels, F, is also limited. Thus, even though we have large

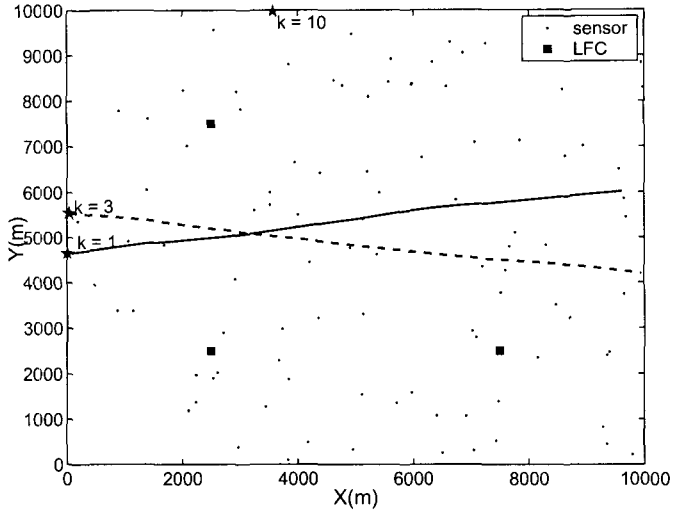


Figure 5.1: A sample scenario

number of sensors, only a few sensors can be used at any one time. Hence, sensors will be activated or deactivated by a control message whenever we change the active sensors [80]. Any active sensor will be connected to one of the LFCs through a frequency channel. Note that one sensor can be connected to at most one LFC to avoid the redundancy and to reduce the correlation between the tracks from each LFC. The transmission powers of the sensors are software controllable within certain lower and upper limits. In order to recover the signals, which are sent by the sensors, at the LFCs, the signal-to-noise ratio at each LFC for each channel must be greater than or equal to a known threshold level.

The objective is to maximize the tracking performance of the system. The tracking performance is measured by the accuracies of the existing targets' estimates and the detection probabilities of new incoming targets. Then, the problem is to assign the sensor subset for each LFC, and assign the transmitting frequency and the power to each sensor, such that SNR is above the threshold level, in order to maximize the tracking performance.

Since we consider a hierarchical architecture with feedback, each LFC has the knowledge of the entire system. Therefore, even if sensor management is performed at every LFC, the result will be the same. In order to avoid the redundancy and to reduce the workload of the LFCs, sensor management is performed at the CFC and the results, i.e., which sensors should be used by each LFC and what frequency channel and how much transmission power should be assigned to each sensor, are sent to all the LFCs. However, if a LFC does not receive any message from the CFC due to communication problems, it will perform the sensor management based on its own knowledge.

5.2 Problem Formulation

In section 3.3, we have shown that the PCRLB for the architecture considered in this chapter is same as the PCRLB of centralized tracking. Hence, centralized PCRLB equations are used in formulation. Note that the ensuing algorithm is not restricted to the architecture with feedback at every measurement step. We can still use the proposed algorithm by deriving and replacing the PCRLB equations for the scenario, in which feedback is not at every measurement step.

5.2.1 Objectives

Let S_{ijf} be the indicator function that takes the value 1 if sensor i is assigned to LFC j through the frequency channel f and 0 otherwise.

The first objective is to minimize the PCRLB of existing targets and given by

$$\min \quad \text{trace} \left\{ \left[J_X(k) + \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{f=1}^{F} S_{ijf} J_{z_i}(k) \right]^{-1} \right\} \tag{5.1}$$

The second objective is to quickly detect the new incoming targets. We assume that the new targets appear only through the perimeter of the surveillance region. As explained in section 4.2.2, we distribute P_{new} particles along the perimeter of the surveillance region to represent the possible states of the new target. Then the second objective, which is similar to (4.18), is given by

$$\min \quad \frac{1}{P_{\text{new}}} \sum_{p=1}^{P_{\text{new}}} \prod_{i=1}^{N} \left(1 - \sum_{j=1}^{M} \sum_{f=1}^{F} S_{ijf} P_d(i,p) \right) \tag{5.2}$$

The objectives are slightly modified in order to avoid wasting a lot of power for marginal improvement in the tracking performance. The modified objectives are

$$\min \quad \sum_{t=1}^{T} \max \left(V_L, \ \text{trace} \left\{ \left[\left(J_X(k) + \sum_{i=1}^{N} \alpha_i J_{z_i}(k) \right)^{-1} \right]_t \right\} \right) \tag{5.3}$$

$$\min \quad \max \left(M_L, \ \frac{1}{P_{\text{new}}} \sum_{p=1}^{P_{\text{new}}} \prod_{i=1}^{N} (1 - \alpha_i P_d(i,p)) \right) \tag{5.4}$$

where $\alpha_i = \sum_{j=1}^{M} \sum_{f=1}^{F} S_{ijf}$, V_L is the tolerable variance of the estimation error of a target, M_L is the tolerable missing probability of a new target and $[\cdot]_t$ is the t-th block diagonal matrix that corresponds to target t.

5.2.2 Constraints

In this section, we will see the constraints that has to be used in our model. First constraint is that a sensor either connected to a LFC through a frequency or not, i.e., S_{ijf} can take only one or zero,

$$S_{ijf} \ \in \ \{0, \ 1\} \qquad \forall \ i, j \text{ and } f \tag{5.5}$$

Since we introduce α_i in the objective for simplification, the relationship of α_i with the S_{ijf}'s must be added as a constraint and it is

$$\sum_{j=1}^{M}\sum_{f=1}^{F} S_{ijf} = \alpha_i \qquad \forall\, i \tag{5.6}$$

From (5.5),(5.6), α_i is an integer, takes zero if it is not used by any fusion center. If a sensor is used by more than one LFC, then duplicate information will be sent to CFC from those LFCs and it is useless. And also the tracks from those LFCs will be correlated, and must be considered in fusion. Hence, a sensor can be assigned to at most one LFC,

$$\alpha_i \leq 1 \qquad \forall\, i \tag{5.7}$$

We have already seen that α_i is an integer, and now from the above constraint α_i is a binary integer.

Due to physical limitations, a LFC can handle a maximum of certain number, n_j, of sensors:

$$\sum_{i=1}^{N}\sum_{f=1}^{F} S_{ijf} \leq n_j \qquad \forall\, j \tag{5.8}$$

Note that there is no need to use two channels to connect a sensor to a LFC, and in the above equations we used that a sensor will be connected a LFC through at most one frequency channel.

Two sensors that are connected to the same LFC cannot use the same frequency since both information will be interfering with each other. Hence at most one sensor

can be connected to one LFC through one frequency:

$$\sum_{i=1}^{N} S_{ijf} \leq 1 \qquad \forall\, j \text{ and } f \tag{5.9}$$

We assumed that transmitting power of the sensors are software controllable with a minimum and maximum limit. Hence, if a sensor is not active then transmitting power should be zero, otherwise its transmission power should be greater than the lower limit, p_l, and less than the upper limit, p_u:

$$P_t^i \leq p_u\, \alpha_i \qquad \forall\, i \tag{5.10}$$

$$P_t^i \geq p_l\, \alpha_i \qquad \forall\, i \tag{5.11}$$

where P_t^i is the transmitting power of sensor i.

The last constraint is that, in order to extract the signal, SNR at each LFC for each frequency must be greater than or equal to the threshold level, $\sigma_{\min}$ [47]:

$$\frac{P_t^i S_{ijf} r_{ij}^{-\lambda}}{\sum_{\substack{x=1 \\ x\neq i}}^{N} \sum_{\substack{y=1 \\ y\neq j}}^{M} P_t^x S_{xyf} r_{xj}^{-\lambda} + N_0} \geq S_{ijf}\, \sigma_{\min} \qquad \forall\, i, j \text{ and } f \tag{5.12}$$

where r_{ij} is the distance between sensor i and LFC j, λ is the decaying factor and N_0 is the environmental noise.

As a summary, the optimization problem that we have to solve is

$$\min \sum_{t=1}^{T} \max \left(V_L,\ \text{trace} \left\{ \left[\left(J_X(k) + \sum_{i=1}^{N} \alpha_i J_{z_i}(k) \right)^{-1} \right]_t \right\} \right) \tag{5.13}$$

$$\min \max \left(M_L,\ \frac{1}{P_{\text{new}}} \sum_{p=1}^{P_{\text{new}}} \prod_{i=1}^{N} (1 - \alpha_i P_d(i,p)) \right) \tag{5.14}$$

Subject to

$$S_{ijf} \in \{0,\ 1\} \qquad \forall\ i, j \text{ and } f \tag{5.15}$$

$$\sum_{j=1}^{M}\sum_{f=1}^{F} S_{ijf} = \alpha_i \qquad \forall\ i \tag{5.16}$$

$$\alpha_i \leq 1 \qquad \forall\ i \tag{5.17}$$

$$\sum_{i=1}^{N}\sum_{f=1}^{F} S_{ijf} \leq n_j \qquad \forall\ j \tag{5.18}$$

$$\sum_{i=1}^{N} S_{ijf} \leq 1 \qquad \forall\ j \text{ and } f \tag{5.19}$$

$$P_t^i \leq p_u\ \alpha_i \qquad \forall\ i \tag{5.20}$$

$$P_t^i \geq p_l\ \alpha_i \qquad \forall\ i \tag{5.21}$$

$$\frac{P_t^i S_{ijf} r_{ij}^{-\lambda}}{\sum_{\substack{x=1\\x\neq i}}^{N}\sum_{\substack{y=1\\y\neq j}}^{M} P_t^x S_{xyf} r_{xj}^{-\lambda} + N_0} \geq S_{ijf}\ \sigma_{\min} \qquad \forall\ i, j \text{ and } f \tag{5.22}$$

The solution methodology to this problem is given in the next section.

5.3 Solution Technique

The original problem is a multi-objective NP-hard combinatorial optimization problem [56]. Finding the optimal solution in real time is not easy for large scale problems. Hence, we propose an algorithm to find a suboptimal solution in real time. The flowchart of the proposed algorithm is shown in Figure 5.2.

In the algorithm, first we select $\min\left(\sum_{j=1}^{M} n_j,\ F\right)$ optimal or suboptimal sensors from the available sensors by considering only the objective function. Note that even though we do not consider the constraints to select the above sensor subset, this solution will always satisfy all the constraints. Then we will add the sensors one by

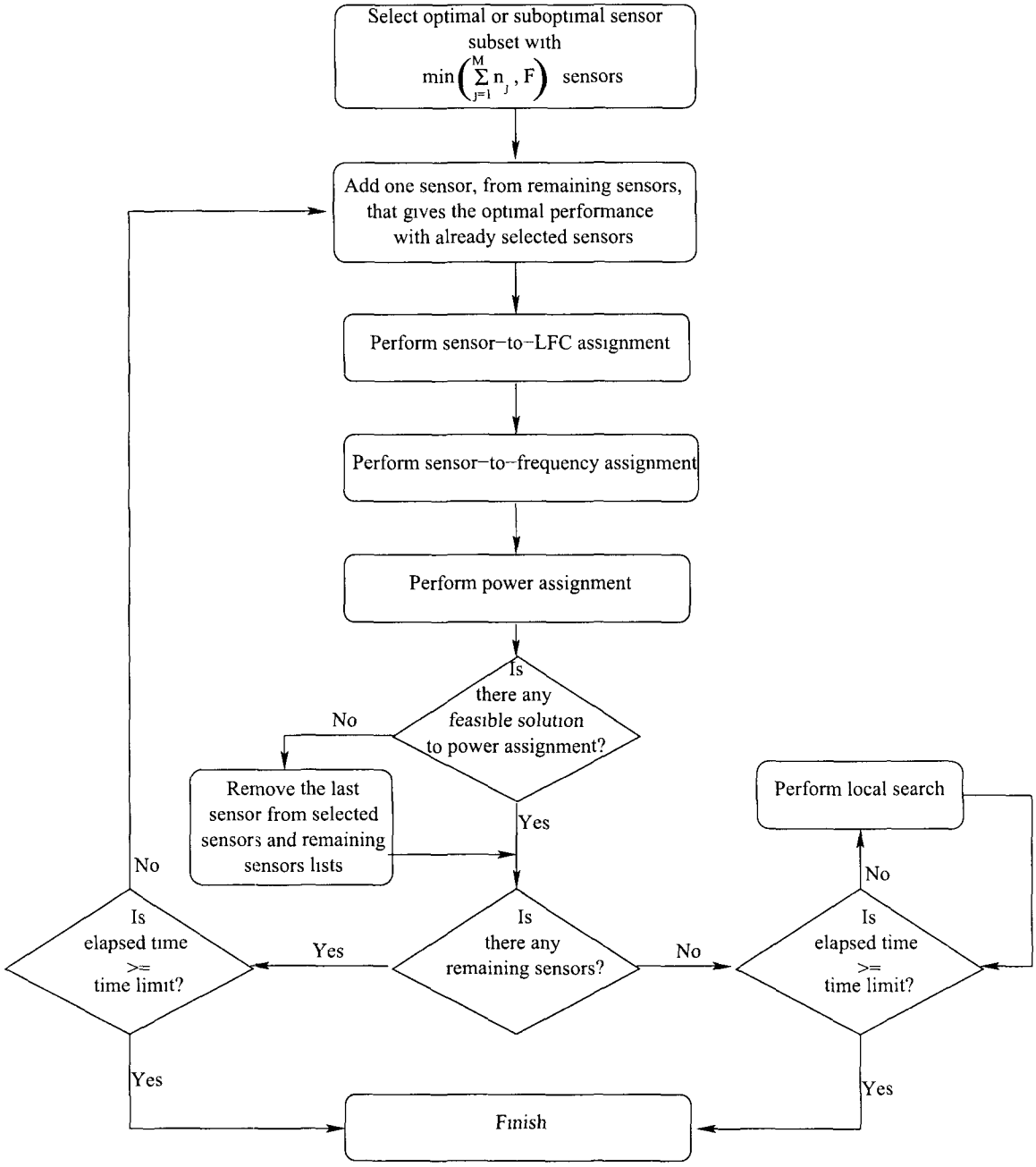


Figure 5.2: Algorithm used to find a suboptimal solution

one so that the objective value is minimized and all the constraints are satisfied. After adding a sensor to already selected sensors, feasibility check is performed as follows:

1. Sensors are assigned to LFCs. Note that this assignment has to be performed from the beginning whenever a new sensor is added to the selected sensor subset.

2. After sensors are assigned to LFCs, the transmitting frequencies will be assigned to each sensor.

3. Finally, transmitting power is assigned to each sensor. However, if no feasible solution is found for power assignment, the selected sensor subset is not feasible and the sensor that was added last is removed from the selected sensors. Note that this removed sensor is not needed to check for feasibility in the future after adding one or more sensors to current sensor subset, since this is obviously infeasible.

After adding possible maximum number of sensors, we will perform local search to find a better solution by swapping the sensors in and out until the stopping criteria is satisfied. While performing the local search, all the remaining sensors including sensors that were ended up with infeasible solutions should be considered.

The detail explanations of the above steps are given next. Let $S_{ijf} = \alpha_i * \beta_{ij} * \gamma_{if}$, where α_i takes the value 1 if that sensor is selected and 0 otherwise, β_{ij} takes the value 1 if sensor i is assigned to LFC j and 0 otherwise, and γ_{if} takes the value 1 if sensor i is assigned to frequency f and 0 otherwise.

The original problem is decomposed into four subproblems:

1. Finding α_i (explained in section 5.3.1)

2. Finding β_{ij} (explained in section 5.3.2)

3. Finding γ_{if} (explained in section 5.3.3)

4. Finding P_t^i (explained in section 5.3.4)

5.3.1 Active sensor selection

In this section, we describe briefly how to select the sensors that should be active. Since handling multiple objectives is difficult, we combine both objectives, by giving suitable weights to each objective, to form a single objective which is easy to handle. The weights can be selected based on the requirements. The combined objective is

$$\min \quad \left\{ \sum_{t=1}^{T} \max \left(V_L,\ \text{trace} \left\{ \left[\left(J_X(k) + \sum_{i=1}^{N} \alpha_i J_{z_i}(k) \right)^{-1} \right]_t \right\} \right) + W_d * \max \left(M_L,\ \frac{1}{P_{\text{new}}} \sum_{p=1}^{P_{\text{new}}} \prod_{i=1}^{N} (1 - \alpha_i P_d(i,p)) \right) \right\} \quad (5.23)$$

where W_d is the weight given to the detection function.

To find α_i, first we have to decide how many sensors to select. The maximum number of sensors that can be active at any one time is $\sum_{j=1}^{M} \min(n_j, F)$. However, if we select $\sum_{j=1}^{M} \min(n_j, F)$ sensors by considering only the objective, we might not be able to find a feasible β_{ij}, γ_{if} and P_t^i with selected sensors, α_i. Hence, first we will select $\min\left(\sum_{j=1}^{M} n_j,\ F\right)$ sensors, which definitely give a feasible solution to the original problem, by finding a better initial solution followed by a local search. The solution consists of the following steps:

1. Select the best sensor that gives the minimum objective value when only one sensor is used.

2. Keep on adding sensors one by one until $\min\left(\sum_{j=1}^{M} n_j,\ F\right)$ sensors selected. If

n sensors are already selected, then the $(n+1)$th sensor is the one that gives the minimum objective value when added with the already selected sensors.

3. Perform a local search by swapping sensors in and out such that the objective value is minimized.

After finding a better solution with $\min\left(\sum_{j=1}^{M} n_j,\ F\right)$ sensors, add sensors one by one until no more sensors can be (or need to be) added. The steps to select the $(n+1)$th sensor are:

1. Rank the remaining sensors based on their performances combined with already selected sensors.

2. Select the best one as the $(n+1)$th sensor and solve for β_{ij}, γ_{if} and P_t^i as explained in sections 5.3.2, 5.3.3 and 5.3.4. If there is no feasible solution for P_t^i, select the next best sensor as the $(n+1)$th sensor and do the feasibility check again. Keep on changing the sensor in the rank order until get a feasible solution.

After selecting the maximum possible number of sensors, a local search is performed by swapping the sensors in and out such that the objective value is minimized and a feasible solution is obtained for β_{ij}, γ_{if} and P_t^i.

Note that the algorithm is interrupted at any stage if the time limit is reached.

5.3.2 Sensor-to-LFC assignment

After selecting the active sensors, we have to decide to which LFCs the selected sensors should be connected. In order to improve the SNR at each LFC for each frequency, the distances of the sensors from the assigned LFCs must be minimized while the distances from other LFCs are maximized.

Then, the sensor-to-LFC assignment, β_{ij}, can be formulated as:

$$\max \quad \sum_{i=1}^{N}\sum_{j=1}^{M}\left(\frac{r_{ij}^{-\lambda}}{\sum_{\substack{y=1\\y\neq j}}^{M} r_{iy}^{-\lambda}}\right)\beta_{ij} \tag{5.24}$$

Subject to

$$\sum_{i=1}^{N}\beta_{ij} \leq n_j \qquad \forall\, j \tag{5.25}$$

$$\sum_{j=1}^{M}\beta_{ij} = \alpha_i \qquad \forall\, i \tag{5.26}$$

$$\beta_{ij} \in \{0,\ 1\} \qquad \forall\, i \text{ and } j \tag{5.27}$$

The above problem is a linear binary integer programming, and it can be solved using CPLEX solver [17].

5.3.3 Sensor-to-frequency assignment

After finding the β_{ij} values for the selected sensors, α_i, the remaining problem is to assign the transmitting frequencies, γ_{ij}, and transmitting powers, p_t^i, to all the selected sensors. Since the sensor battery energies are limited, we have to minimize the power consumption to increase the sensor's life time. The remaining problem is:

$$\min \quad \sum_{i=1}^{N} P_t^i \tag{5.28}$$

Subject to

$$\gamma_{if} \in \{0,\ 1\} \qquad \forall\ i \text{ and } f \tag{5.29}$$

$$\sum_{i=1}^{N} \alpha_i\ \beta_{ij} \gamma_{if} \leq 1 \qquad \forall\ j \text{ and } f \tag{5.30}$$

$$P_t^i \leq p_u\ \alpha_i \qquad \forall\ i \tag{5.31}$$

$$P_t^i \geq p_l\ \alpha_i \qquad \forall\ i \tag{5.32}$$

$$\sum_{i=1}^{N} \alpha_i\ \beta_{ij}\ \gamma_{if}\ P_t^i\ r_{ij}^{-\lambda} \geq \sigma_{min} \left(\sum_{\substack{x=1 \\ x \neq i}}^{N} P_t^x \sum_{y=1}^{M} \gamma_{xf} r_{xj}^{-\lambda} + N_0 \right) \qquad \forall\ j \text{ and } f \tag{5.33}$$

This is also a NP-hard problem. If the transmission power is fixed, the above problem can be solved easily. However, in order to incorporate the advantage of the variable transmission power, γ_{if} and P_t^i have to be considered together. The following approximation is used to solve the above problem.

1. All the LFCs are arranged in an order. Let the first LFC be the one in any one corner. The next LFC is the one closest to the previous LFC form the remaining LFCs.

2. Assign the frequencies to the first ($j = 1$) LFC's sensors. Since the number of sensors assigned to this LFC is less than or equal to F, simply assign any one frequency, but different, to one sensor.

3. Assign the frequencies to the j-th LFC's sensors by solving the following optimization problem:

$$\min \quad \max_{\substack{i=1,\ldots,N_j \\ f=1,\ldots,F}} C_{if}\ \overline{\gamma}_{if} \tag{5.34}$$

Subject to:

$$\overline{\gamma}_{if} \in \{0, 1\} \qquad \forall\, i \text{ and } f \tag{5.35}$$

$$\sum_{f=1}^{F} \overline{\gamma}_{if} = 1 \qquad \forall\, i \tag{5.36}$$

$$\sum_{i=1}^{N_j} \overline{\gamma}_{if} \leq 1 \qquad \forall\, f \tag{5.37}$$

where N_j is the number of sensors assigned to the j-th LFC and C_{if} is the cost of assigning frequency f to sensor i.

Note that $\overline{\gamma}$ is used instead of γ, since only the subset of sensors that are assigned to the j-th LFC is considered. However after finding the solution to $\overline{\gamma}$, γ will be updated.

To find the value of C_{if}, the cost of assigning frequency f to sensor i, we first find the transmission powers correspond to the sensors that use frequency f, including sensor i, by using the formulation given in the next subsection (5.3.4), and then we calculate C_{if} by

$$C_{if} = \max_{y \in \{i_f\}} \left(\frac{\sum_{\substack{x=1 \\ x \neq y}}^{N_f} P_t^x r_{xj_y}^{-\lambda} + N_0}{P_t^y r_{yj_y}^{-\lambda}} \right) \tag{5.38}$$

where i_f indicates the sensors that are using frequency f including sensor i, N_f is the number of sensors using frequency f and j_y is the LFC to which sensor y is assigned.

The C_{if} gives the maximum of the noise-to-signal ratio for frequency f at any LFC, if frequency f is assigned to sensor i. In order to improve the possibility

of getting a feasible solution for P_t^i, the maximum value of the noise-to-signal ratio over all the frequencies is minimized in (5.34).

The above problem can be reformulated as a linear integer problem:

$$\min \quad T \tag{5.39}$$

Subject to (5.35), (5.36), (5.37) and:

$$C_{if}\,\overline{\gamma}_{if} \;\leq\; T \qquad \forall\, i,\; f \tag{5.40}$$

This problem can also be solved using CPLEX solver [17].

4. Continue this process until no more LFC remains.

5.3.4 Transmission power assignment

After finding the α_i, β_{ij} and γ_{if}, the remaining problem is to find the transmission power for each active sensor. In order to increase the life time of the sensors, transmitting powers must be minimized while satisfying the constraints.

Transmitting power assignment, P_t^i, can be formulated as

$$\min \quad \sum_{i=1}^{N} P_t^i \tag{5.41}$$

Subject to

$$P_t^i \leq \alpha_i \, p_u \qquad \forall \, i \tag{5.42}$$

$$P_t^i \geq \alpha_i \, p_l \qquad \forall \, i \tag{5.43}$$

$$\sum_{i=1}^{N} \alpha_i \, \beta_{ij} \, \gamma_{if} \, P_t^i \, r_{ij}^{-\lambda} \geq \sigma_{min} \left(\sum_{\substack{x=1 \\ x \neq i}}^{N} P_t^x \sum_{y=1}^{M} \gamma_{xf} r_{xj}^{-\lambda} + N_0 \right) \qquad \forall \, j, f \tag{5.44}$$

The above problem is a linear convex problem [10], and can be solved easily using the CPLEX solver [17].

Note that for some α_i, β_{ij} and γ_{if}, there will not be any feasible solution for power. In that case, α_i, β_{ij} or γ_{if} has to be changed so that a feasible solution is found.

5.4 Simulation Results

In the simulation scenario, the number of targets $T = 2$; the number of LFCs $M = 4$; the number of sensors $N = 100$; the maximum usable sensors by each LFC $n_j = 4$; the available number of channels $F = 5$; the measurement interval is 30 seconds; measurements are the bearing from the sensors to targets; measurement standard deviation $\sqrt{\Sigma} = 0.05$ radians; the field of view of the sensor $V = 2\pi$ radians; the transmission power bounds $p_l = 0$ and $p_u = 1$ Watts; the threshold of SNR $\sigma_{min} = 10$ dB; the probability of false alarm is 0.001.

Since the measurement model is nonlinear, EKF based information filter (see section 2.1.2) is used to track the targets at each LFC [4, 5]. Because of the existence of multiple targets and false alarms, measurements must be associated to already established tracks, new tracks or a dummy track that corresponds to tracks due to false alarms. First, we used the (S+1)D (or multiframe) assignment algorithm to

associate the measurements with the already established tracks [66]. The remaining measurements, which are not associated with any of the already established tracks, are associated with new tracks or dummy track using the S-D algorithm [57]. New tracks are formed with the associated measurements using Iterated Least Squares (ILS) estimator [5].

Estimation at the CFC is performed using (2.44) and (2.45) that consider a single target. However, multiple targets can be tracked by each fusion center, and the estimate of each target has to be updated. Hence, first CFC will group the estimates, that belong to same target, from all the LFCs. Since LFCs get the feedback from the CFC, track ID can be used to identify the target that corresponding to each estimate [67]. However, track ID will be the same only for the already established tracks. If new tracks are formed by LFCs then the technique explained in section 2.3.4 is used to decide if they are from the same target.

The solution for the sensor management using the proposed algorithm is shown in Figure 5.3. We have shown the selected sensors, their LFC and frequency assignments. In that figure, '■' indicates the LFC and the capital letter (e.g., A, B, ...) near to it indicates its name. The '·' indicates the sensor, '•' indicates the selected sensor and the small letter (e.g., a, b, ...) near to it indicates that the sensor is assigned to the corresponding capital letter LFC (e.g., sensor 'a' is assigned to LFC 'A'). The number near the selected sensor indicates the frequency channel that is assigned to it. The '*' indicates the target. When the decaying factor is high, the co-channel interference is less. Hence we could use more sensors with decaying factor 4 than decaying factor 2. However, even with decaying factor 2, the proposed algorithm assigns enough sensors to the already exiting targets and other sensors to cover the boundaries through which new targets can enter the surveillance region. Since we

could not find any other method that can handle similar type of sensor selection problem in the literature, we could not compare this result to any other method.

In order to compare proposed method with some existing methods, we used a slightly different scenario. The differences are: the sensors are randomly distributed in the surveillance region; the number of targets $T = 3$; the available number of channels $F = 20$. Since the available number of channels (20) is greater than the maximum usable sensors by all the LFCs (4x4), the frequency assignment is not an issue here and the solution can be found easily. However, we do not use any clustering in the proposed algorithm.

In the literature, sensors are clustered based on target or fusion center [79]. Target based clustering cannot be applied to our scenarios, since number of target is time varying and a sensor can get measurements from more than one target. Fusion center based clustering is applicable to the above scenario, and an algorithm using clustering is explained next.

First sensors are clustered based on the LFCs, i.e., a sensor can be used only by the closest LFC. The clustered sensors are shown in Figure 5.4. Clustering helps to perform the sensor-to-LFC assignment, and it is performed prior to selecting the active sensors. The remaining problem is to select the active sensors. In the literature, an LFC considers only its region to select its sensors. However, to reduce the difference between proposed algorithm and this algorithm, we assume that whole region is considered to select the active sensors. Also in many papers in the literature, tracking and coverage problems are considered separately [6, 7, 18, 34, 59]. That is, either they considered only the tracking accuracy of the existing target or the detection probability of existing and newly appearing targets. However, here already existing targets as well as the possible new targets are considered in sensor selection.

The comparisons of proposed algorithm with the algorithm [79] that used LFC based clustering are shown in Figures 5.5, 5.6 and 5.7. Figure 5.5 shows the selected sensors and their LFC and frequency assignments. The symbol notations in this figure are same as in Figure 5.3. There are three targets around the LFC B. In proposed algorithm, enough sensors are assigned to those targets and others sensors are assigned to catch the new targets. In the algorithm with clustering, only four sensors, which are not enough to track all three targets, are assigned to the existing targets. Figure 5.6 and 5.7 show the comparison of the RMSE and the PCRLB, respectively. Proposed algorithm gives more than 30 percent improvement than the algorithm that used clustering.

The average computation time of the proposed algorithm for the above scenario is 4.2 seconds. The corresponding time for LFC based clustering algorithm is 3.1 seconds. Both algorithms are coded in *Matlab* and run on a 2.4 GHz Pentium 4 processor. The proposed algorithm gives around 30 percent improvement in the performance than the algorithm that used clustering with around 35 percent more computation time. Computation time is not much important as long as it is smaller than the allocated time, and it can be further reduced by coding the algorithms in *C*.

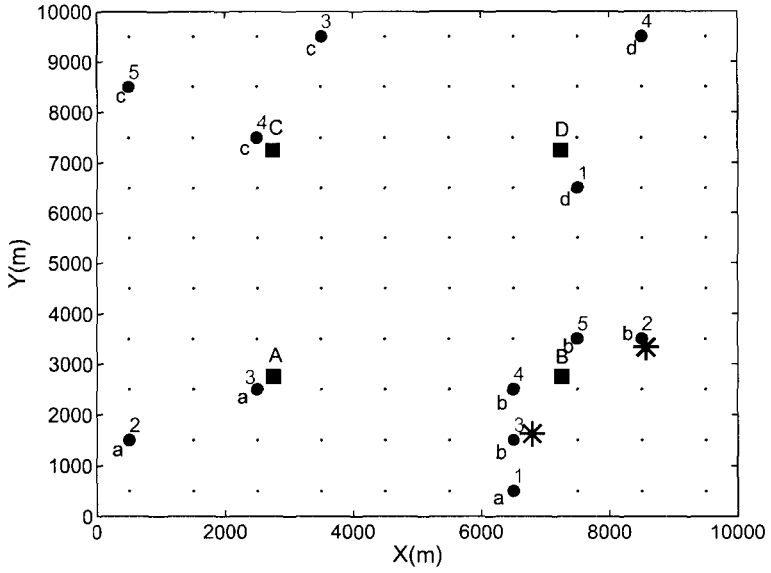
Even though no paper in the literature consider the frequency and transmission power assignment with active sensor selection, to see the affect of clustering on computation and the performance we consider the following scenario: the number of targets $T = 3$; the available number of channels $F = 6$, the maximum usable sensors by each LFC $n_j = 4$; decaying factor $\lambda = 4$. In the algorithm with clustering, LFC based clustering is used instead of 5.3.2. Algorithms are stopped if either allocated time is reached or a local optimum is found with one sensor swapping.

The best objective function values found by both algorithms in the allocated times

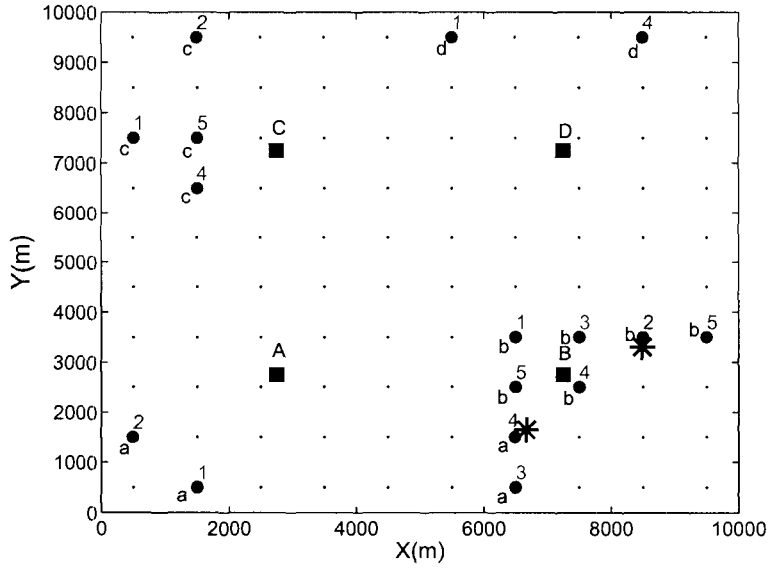
Table 5.1: The best objective function values found using clustering and proposed algorithms

Allocated time (seconds)	Scenario 1		Scenario 2	
	Algorithm with clustering	Proposed algorithm	Algorithm with clustering	Proposed algorithm
5	1222	5851	7214	10966
15	929	5831	6926	10964
25	929	715	6926	3285
35	929	668	6926	3285

are shown in Table 5.1. When the allocated time is less than or equal to 15 seconds, the algorithm that used clustering gives the better solution. However, with more allocated time proposed algorithm find a better solution and it is (30–50 percent) better than the solution obtained by the algorithm with clustering. Note that the computation times can be reduced significantly by coding the algorithm in *C*, and in most cases we will have enough time to reach a better solution using the proposed algorithm.



(a) decaying factor $\lambda = 2$



(b) decaying factor $\lambda = 4$

Figure 5.3: Selected sensors and their LFC and frequency assignments.

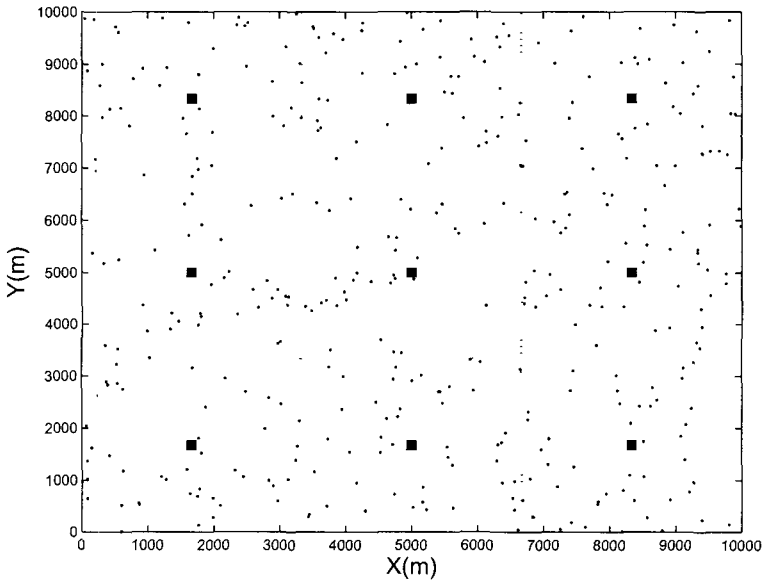
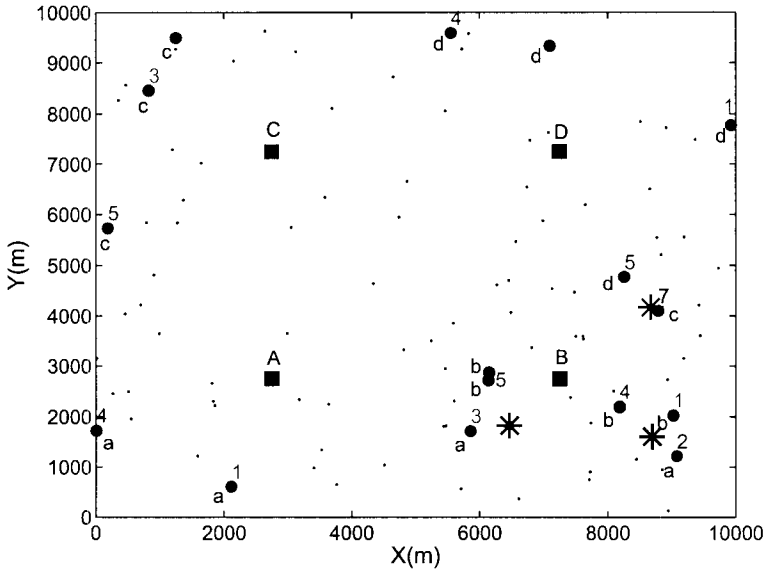
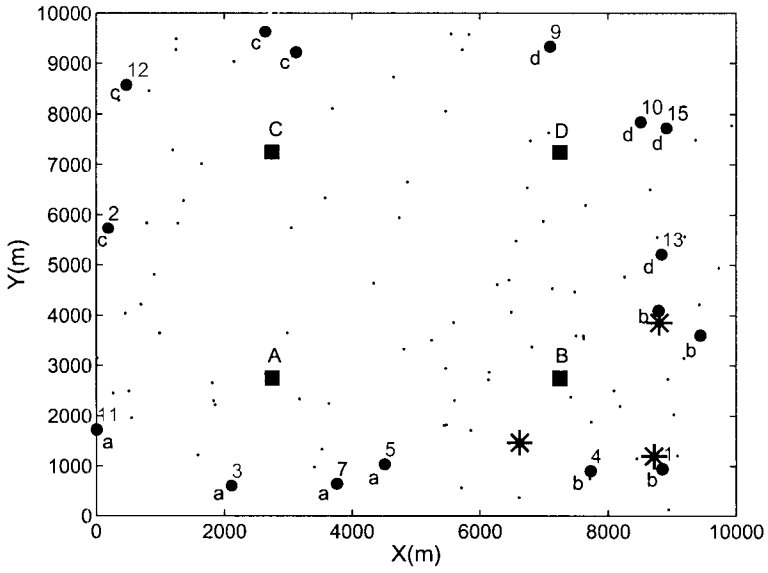


Figure 5.4: A sample scenario with clustered sensors

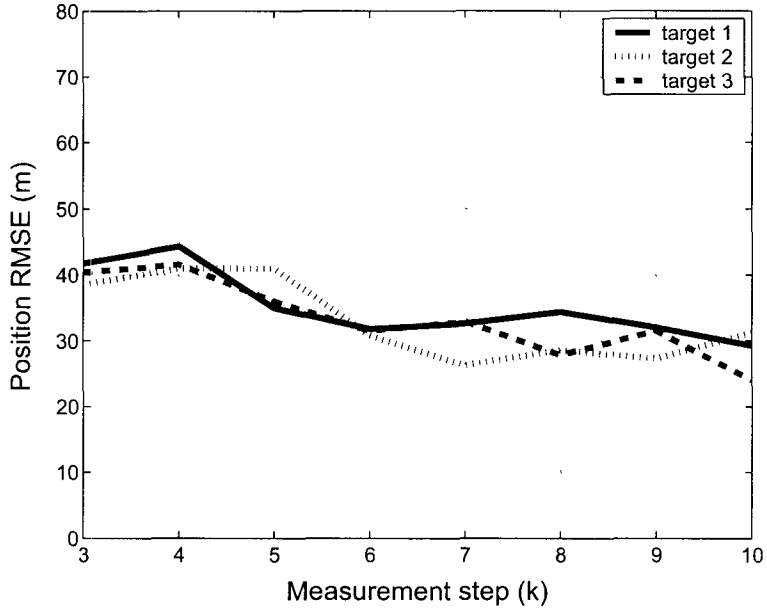


(a) Proposed algorithm

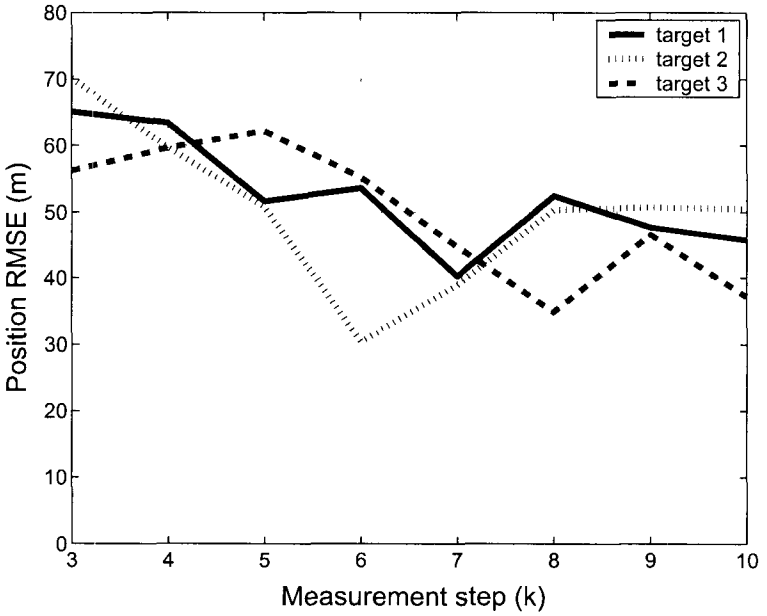


(b) Algorithm using clustering

Figure 5.5: Sensors selected with proposed algorithm and an algorithm that used LFC based clustering.



(a) Proposed algorithm



(b) Algorithm using clustering

Figure 5.6: RMSE comparison of proposed algorithm with an algorithm that used clustering.

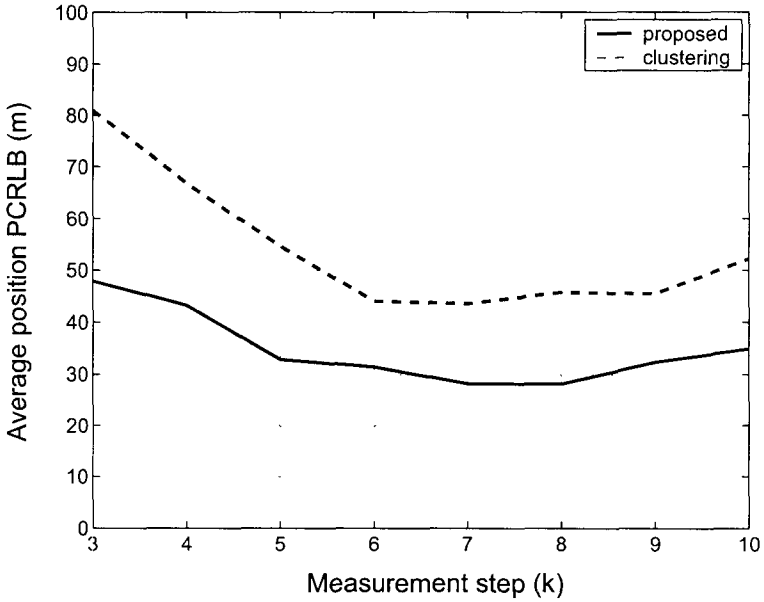


Figure 5.7: PCRLB comparison of proposed algorithm with an algorithm that used LFC based clustering.

Chapter 6

Sensor Array Management for Decentralized Tracking

In this chapter, we extend the works of previous two chapters to decentralized architecture, shown in Figure 2.7, in which there is no central fusion center and each FC gets information only from its neighbors.

6.1 Problem Description

In our problem, there are few fusion centers in the surveillance region. Each FC can communicate only with its neighbors. The FCs that are in the communicable distance from a FC are considered as its neighbors. There are a large number of sensors, which are already deployed, and their locations are fixed and known. However, due to physical limitations, only a maximum of certain number of sensors, n_f, can be used by each FC, f, at any one time. In addition, the active sensor subset can only be changed after a certain minimum number of time steps. The number of targets in the surveillance region is time-varying, i.e., a new target can appear or an already

existing target can disappear at any time. A sample scenario is shown in Figure 6.1, where · indicates sensor, ■ indicates FC and ★ indicates target.

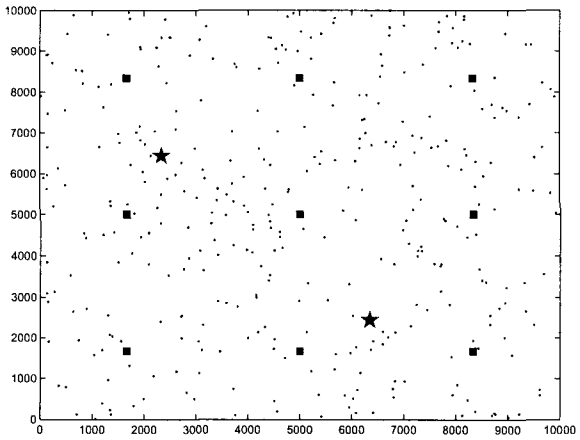


Figure 6.1: A sample scenario

The problem is to assign a sensor subset to each FC such that the tracking performance is maximized, i.e., the already established tracks are maintained accurately and the newly incoming targets are detected immediately and accurately. Note that because of the additional difficulties due to lack of global knowledge, we assume that the frequencies are already allocated to each FC, and only problem is to find the active sensors.

6.2 Problem Formulation

We can obtain the optimal solution, if there is a central fusion center and it performs the sensor selection considering all the possibilities. However, in the decentralized architecture this is not feasible, since FCs may not have the global knowledge, communication might not be reliable, and it is computationally very demanding. Hence, each FC has to decide its own sensors. Even though each FC selects its own sensors,

it cannot make the decision based on its region only. In order to ensure better sensor selection and thus better tracking results, FCs have to consider which of the sensors will be used by other FCs, at least the neighboring ones, as well. Considering all other FCs is not feasible due to lack of global knowledge and heavy computation requirements in large scale problems. Hence, each FC considers only its neighbors while performing the sensor selection.

First, in order to avoid one sensor being used by more than one fusion center, which will introduce redundancy as well as correlation between tracklets, sensors are clustered based on geographic location. That is, a sensor can only be used by the nearest FC. Then, each FC will select the sensors from its region. However, in the later sections we propose some techniques to avoid clustering, which will help to improve the performance as shown in the previous chapter.

6.2.1 Synchronous sensor selection

In this section, it is assumed that FCs exchange their information just before they select their sensors, and all the FCs change their sensors at the same time. Active sensor subsets of all the FCs are changed after every l measurement time steps. A sample communication and sensor change time steps with $l = 5$ are shown in Figure 6.2, where $\square$ indicates the tracklet transmission and fusion, and $\diamond$ indicates the sensor change.

Let us consider the sensor management at FC a. Suppose the next sensor change occurs at time $k + 1$, and the FIM, after fusing neighbors' information, at time k is $J^a(k)$. Suppose it has M neighbors (including itself). It has to decide which of the sensors should be used at time steps $k+1, k+2, \ldots, k+l$. Note that the same sensors

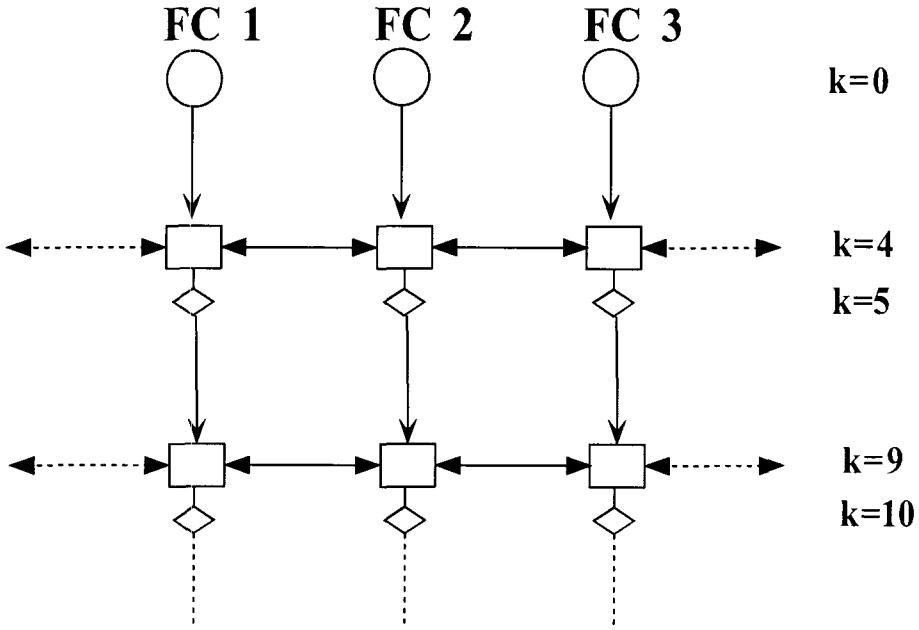


Figure 6.2: A sample sensor change and communication time steps

will be used in all these time steps. Then the objective is to:

$$\arg \min_{\{S_i^f; f=1,2,\ldots,M\}} \left(\sum_{j=1}^{l} \left(W_j \sum_{t=1}^{T} \left(W_t \min_{f=1,2,\ldots,M} \text{trace}\left(\left[J^f(k+j)^{-1} \right]_t \right) \right) \right) \right) + W_{\text{new}} \left(\mathbb{E} \min_{f=1,2,\ldots,M} \text{trace}\left(J^f_{\text{new}}(x_{\text{new}})^{-1} \right) \right) \quad (6.1)$$

where

$$J^f(k+j) = [F_{k+j} J^f(k+j-1)^{-1} F'_{k+j} + \Gamma_{k+j}]^{-1} + \sum_{i=1}^{N_f} S_i^f J^f_{z_i}(k+j) \quad (6.2)$$

Note that instead of $J^f(k+l)$, $J^f_{\text{fused}}(k+l)$ is used, and it is given by

$$J^f_{\text{fused}}(k+l) = J^f(k+l) + \sum_{\substack{b=1 \\ b \neq f}}^{M} \left(J^b(k+l) - J^b_X(k+l|k) \right) \quad (6.3)$$

In (6.1), W_{new} is the weight given to the new targets, W_j is the weight given to $(k+j)$th time step and N_f is the number of sensors that belong to FC f. S_i^f takes one if i-th sensor of FC f is selected and zero otherwise. The $\mathbb{E}$ denotes the expectation over all possible new target positions. In general, the expectation has to be found numerically, as explained in the previous two chapters.

The information about the new target, J_{new}, is given by:

$$J_{\text{new}}^f(x_{\text{new}}) = \sum_{i=1}^{N_f} S_i^f J_{z_i}(x_{\text{new}}) \tag{6.4}$$

Since there is no prior information about the new target, the information from the sensors is only considered in the above equation. However, if there is any prior information, that also can be added. If the velocity information of the new target is not available in one time step, then only the position PCRLB can be considered.

Note that in the previous two chapters we used the probability of detection as the cost function for newly incoming targets. In those two chapters, we assumed that the new targets appear only through the boundaries. In that case, even one sensor is enough to initialize the new target. However, if the new targets can appear anywhere in the surveillance region, then, in general, more than one sensor may be necessary to initialize a track. For an example, if the measurements are the bearings from sensor to target, then at least two sensors are necessary to initialize the track. In that case, PCRLB of the new targets is the suitable and better choice for cost function, and used in this chapter. Even in the formulations of previous two chapters, we can replace the probability of detection with PCRLB, when new targets can appear anywhere in the surveillance region.

The constrains are:

$$\sum_{i=1}^{N_f} S_i^f = n_f \qquad \text{for } f = 1, 2 \ldots, M \tag{6.5}$$

$$S_i^f \in \{0, 1\} \qquad \forall\, f, i \tag{6.6}$$

Note that only the targets that are in the neighboring regions are considered in the above equations. Even for the new target, the $\mathbb{E}$ is taken only over the neighboring regions.

For the sensor management purpose, we assume that neighbors also have the same amount of information about all the targets at time k. Then, $J_X^f(k+1) = J_X^a(k+1)$ for all f.

6.2.1.1 Sequential Sensor Management

In the above algorithm, each FC selects its own sensors by considering which sensors will be used by its neighboring FCs. However, in the end, the neighbors might select different sensors than what a FC thought, since neighbors consider their neighbors that are not considered by the FC, and also they might have different estimates. That is, although a FC considers its neighbors' action in its sensor selection, it does not have all the information nor does it have the correct information of neighbors at all times. The effect of this problem can be mitigated by performing the sensor management in a predefined sequence.

An example sequence corresponding to the architecture shown in Figure 2.7 is shown in Figure 6.3. First, the FCs with tag 1 will perform the sensor selection by considering which sensors will be used by its neighbors. After selecting their sensors, they will inform their selection to their neighbors. Then the FCs with tag 2 select

1	2	1	2	1
2	3	2	3	2
1	2	1	2	1
2	3	2	3	2
1	2	1	2	1

Figure 6.3: Predefined sequence of fusion centers

their sensors. However, now they do not need to find the sensors for neighbors with tag 1. Instead they will use the known values. Hence computational load of these FCs is less than FCs with tag 1. After they select their sensors, they will inform their selection to their neighbors with tag 3. Finally FCs with tag 3 perform the sensor selection using the information about the neighboring FCs. Computation load of these FCs is very low compared to others, since they have to find only their sensors.

Since the FCs with tag 2 consider the actual sensors that will be used by their neighbors with tag 1, they will select better sensors that will cover the uncovered areas by neighbors. Similarly, FCs with tag 3 will select the sensor subsets that overcome the neighbors mistakes.

An additional advantage of sequential sensor selection is that clustering is not necessary, since FCs with tag 2 can avoid the sensors selected by FCs with tag 1, similarly FC with tag 3 can avoid the sensors already selected by its neighbors.

However, the disadvantage of sequential sensor selection is that it requires more communication and later FCs must wait until others select their sensors.

6.2.2 Asynchronous sensor selection

In the previous section, we assumed that communication and sensor selection are synchronized over the FCs, and sensor change intervals are fixed and known. However, in general scenario FCs may not change their sensors at the same time, and sensor change interval may not be fixed. Hence, in this section we consider a more general scenario with an additional objective to maximize the sensor life times, when sensors have limited energy resources.

The active sensor subset of each FC is changed if the estimation uncertainty of any target goes above a certain limit. However, there is a lower and upper limit for sensor change interval. Let say, last sensor change occurred at time step k and current time step is $k+j$, then a binary variable, c, that takes value 1 if need to change the current active sensor subset, and zero otherwise is given by:

$$c = \begin{cases} 0 & \text{if } j \leq C_l \text{ or } (j \leq C_u \text{ and } U_t < L_u \text{ for all } t) \\ 1 & \text{if } j > C_u \text{ or } (j > C_l \text{ and } U_t > L_u \text{ for any } t) \end{cases} \tag{6.7}$$

where C_l and C_u are lower and upper limits of sensor change interval respectively, U_t is the uncertainty of state estimate of target t and L_u is tolerable estimation uncertainty. Thus, FCs may change their sensor subsets at different time steps and any FC cannot predict the exact next sensor change times of its neighbors. Rather, it will know only an interval, which can be calculated using the last sensor change time, lower and upper limit of sensor change interval of its neighbor.

Fusion centers transmit the tracklets to their neighbors just before they change their active sensor subset. In addition to tracklets, the detail of new sensor subset is also transmitted at that time. Since the sensors are changed asynchronously and FCs send their track information just before they change their sensors, communication

times are also asynchronous. A sample sensor change time steps are shown in Figure 6.4, where □ indicates the track fusion, and ◇ indicates the sensor change and the transmission of tracklets.

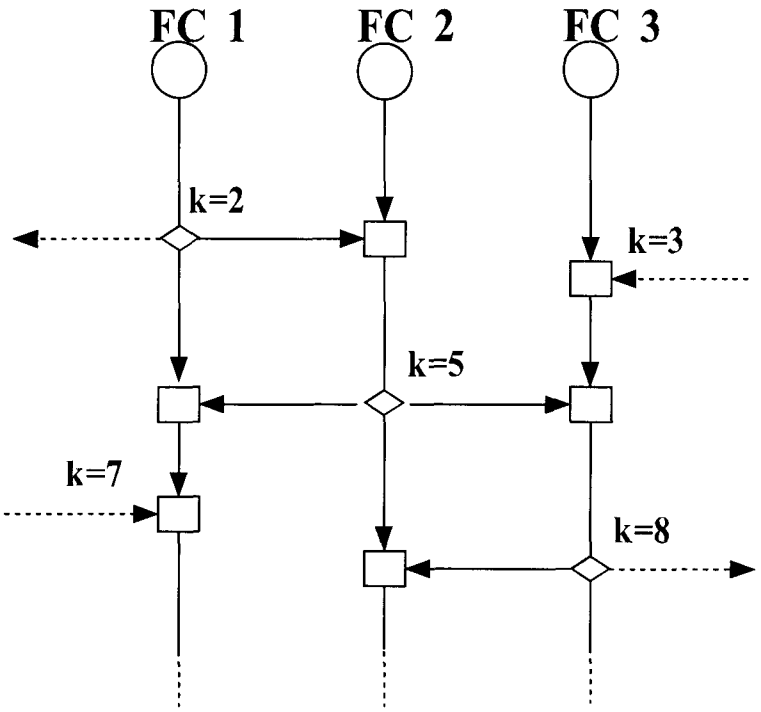


Figure 6.4: A sample sensor change and communication time steps

Let us consider the sensor selection at FC a at time k, at which sensor selection criteria is satisfied.

The objectives are:

1. Optimize the tracking performance of existing targets:

$$\min \sum_{j=1}^{C_u} \sum_{t=1}^{T} \left(W_{jt} \min_{f=1,2,\ldots,M} \max_{v=1,2,\ldots,n_v^f(k+j)} \text{trace}\left(\left[J_v^f(k+j)^{-1} \right]_t \right) \right) \tag{6.8}$$

where $n_v^f(k+j)$ is the number of possible versions of FIMs at FC f at time step

$k+j$ and W_{jt} is the weight given to target t at time step $k+j$. It is possible to have more than one version of FIM at any time because of the uncertainties in sensor change times and tracklet transmission times. For an example, assume FC a has only one neighbor, last sensor change of neighbor occurred at $k-3$, $C_l = 5$ and $C_u = 7$.

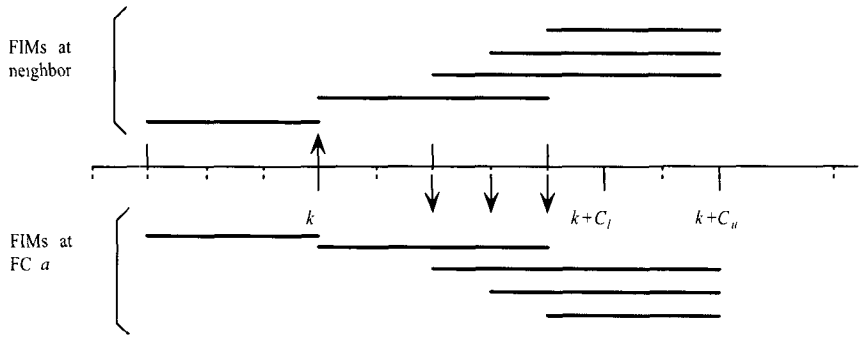


Figure 6.5: Possible versions of FIMs

Figure 6.5 shows the possible versions of FIMs at neighbor and FC a. At time k, FIM of neighbor is updated using the tracklet from FC a, and FC a start to use the new sensor subset. At k and $k+1$, only one version of FIM is available in both FCs due to no uncertainty in those time steps. However, at $k+2$, $k+3$ and $k+4$ there are possibilities for sensor change and tracklet transmission at neighbor. Hence, there are 2 possible versions of FIMs at time $k+2$ and 3 versions at $k+2$ to $k+C_u$, as shown in Figure 6.5. Note that multiple version of FIMs are possible at FC a due to uncertainty in tracklet receive time from neighbor, and at neighbor due to uncertainty in sensor change time. Even though, there is a possibility for sensor change at any time step between $k+C_l$ and $k+C_u$ at FC a, in (6.8) we take the summation over C_u time steps under the assumption that the next sensor subset may be used until $k+C_u$. However the possibility of changing another new sensor subset before $k+C_u$ is incorporated

in W_{jt}.

Note that only the targets in the neighboring regions are considered in the above equation, under the assumption that other targets will be tracked by some other FCs.

2. Detect the newly incoming targets quickly and accurately:

$$\min \sum_{j=1}^{C_u} \sum_{p=1}^{P_{\text{new}}} W_{jp} \min_{f=1,2,..,M} \max_{S^f=\{S^f_{\text{current}},S^f_{\text{new}}\}} \text{trace}\left(J^f_{\text{new}}(x^p_{\text{new}}, S^f, j)^{-1}\right) \quad (6.9)$$

where x^p_{new} is the possible state of new target, P_{new} is the number of particles used for representing possible new target states, S^f_{current} and S^f_{new} are current and next sensor subsets of FC f, and W_{jp} is the weight given to particle p at time step j. Note that new target particles are distributed only over the neighboring regions, and prior information about new target states are used in distributing particles. In the J^f_{new} calculation, only the S^f_{new} is used for FC a, and for others S^f_{current} is used if $k+j-l_f < C_l$, both possibilities, i.e., S^f_{current} and S^f_{new}, are considered if $C_l \leq k+j-l_f < C_u$, and S^f_{new} is used otherwise.

3. Optimize the sensor life times:

$$\min \sum_{f=1}^{M} \sum_{i=1}^{N_f} C^f_i S^f_i \quad (6.10)$$

where C^f_i is the cost of i-th sensor of FC f, which is the function of its remaining energy and the sensors close to it. If there is another sensor, which can give almost equal performance when replaced for the particular sensor, with enough energy, then the cost of particular sensor must be close to zero. On the other

hand if we do not have any sensor close by, then there is no replacement for it, and we want to use it for long time, and the cost must be high. The cost can be calculated as

$$C_i = \max\left(0, \frac{\min(D^i_{\text{neig}}, D_{\max})}{D_{\max}} - \frac{B^i_{\text{rem}}}{B_{\max}}\right) \tag{6.11}$$

where D^i_{neig} is the distance of the nearest sensor, which has more energy than sensor i, B^i_{rem} and $B_{\max}$ are the remaining and initial battery energy of sensor i, respectively. $D_{\max}$ is design parameter, and it should be decided based on sensor type. Note that the cost of each sensor can be calculated prior to sensor selection.

Since it is hard to handle multiple objectives, we form a single objective by combining the objectives with suitable weights, and given by

$$\begin{aligned}\arg \min_{\{S^f_i; f=1,2,..,M\}} \quad & \sum_{j=1}^{C_u}\sum_{t=1}^{T}\left(W_{jt} \min_{f=1,2,\ldots,M} \max_{v=1,2,\ldots,n^f_v(k+j)} \text{trace}\left(\left[J^f_v(k+j)^{-1}\right]_t\right)\right) \\ & + \sum_{j=1}^{C_u}\sum_{p=1}^{P_{\text{new}}} W_{jp} \min_{f=1,2,\ldots,M} \max_{S^f=\left\{S^f_{\text{current}}, S^f_{\text{new}}\right\}} \text{trace}\left(J^f_{\text{new}}(x^p_{\text{new}}, S^f, j)^{-1}\right) \\ & + W_b \sum_{f=1}^{M}\sum_{i=1}^{N_f} C^f_i S^f_i \end{aligned} \tag{6.12}$$

where W_b is the weight given to the objective of sensor life times.

The constrains are:

$$\sum_{i=1}^{N_f} S^f_i \leq n_f \qquad \text{for } f = 1, 2 \ldots, M \tag{6.13}$$

$$S^f_i \in \{0,1\} \qquad \forall\, f, i \tag{6.14}$$

Note that, if W_b is zero, then the solution of the above problem will be same as that of the above problem with modified constraint

$$\sum_{i=1}^{N_f} S_i^f = n_f \qquad \text{for } f = 1, 2 \ldots, M \tag{6.15}$$

instead of (6.13).

Already we have mentioned that sensors are clustered based on FCs in order to ensure that no sensor is used by more than one FC. However, an advantage of asynchronous sensor changes is that clustering can be easily avoided if any two neighbors are not changing the sensors at the same time. However, if two neighbors decided to change their sensors at the same time, then collaboration is necessary to avoid the clustering.

6.2.2.1 Weight selection

In this section, we describe how to select the weights, which are used in the objective functions, scenario dependently.

The weights are:

- Weight of target t at time $k + j$, W_{jt}.

  We give the weight to each target according to the target type and the state. One might want to give more weight to enemy targets compare to friendly targets. Similarly, the state of the target is also an important factor in weight selection. Low speed targets are less dangerous than high speed targets. In addition to the speed, position of the target should also be considered in weight selection. We give more weights to the targets that are close to our important locations.

PCRLB of each target is calculated based on its predicted states. If we have accurate prediction over all the time steps, then we can give equal weight to all the time steps. However, prediction accuracy of target state diminishes with time, and hence the weight of time steps also should be decreased with time.

In (6.8), we assume that the new sensor subset will be used up to $k + C_u$. However, another new sensor subset can be used at anytime after $k + C_l$, and this uncertainty must also be incorporated into the weights.

Then the weight is given by:

$$W_{jt} = w_{jt} \times \frac{E_{\max}}{\max\left(E_{\max}, \sqrt{\text{trace}\left([J_X(k+j|k)^{-1}]_t\right)}\right)} \times \frac{1}{\max(1, j - C_l + 1)} \quad (6.16)$$

where w_{jt} is the weight given to target t at time $k + j$ based only on target type and state, as explained above, and $E_{\max}$ is a design parameter. $E_{\max}$ is the maximum error in predicted state that may not effect the sensor selection significantly, and this is a function of measurement models and sensor locations. For an example, if the measurements are angle only and sensors are placed 500m apart, then a suitable value for $E_{\max}$ can be 125.

Finally, we normalize all the weights so that it is easy to define the weights of other objectives.

- Weight of new target particle p at time $k + j$, W_{jp}.

  Similar to existing targets, we can give more weights to the particles that are close to our important locations. In addition, we may want to give less or more priority to new targets compare to already established tracks. Then the weight

is given by

$$W_{jp} = w_{\text{new}} \times \frac{w_p \times \frac{1}{\max(1, j - C_l + 1)}}{\sum_{p=1}^{P_{\text{new}}} \sum_{j=1}^{C_u} w_p \times \frac{1}{\max(1, j - C_l + 1)}} \tag{6.17}$$

where w_{new} is weight given to new targets compared to already established tracks, w_p is the weight given to particle p based on its state.

- Weight of sensor life times, W_b.

  Since the units of first two objective functions are different from third objective function, we need a suitable weight to balance them. In addition, we can use this weight to specify how important is the sensors' life times compared to tracking performance. For an example, if we expect to have trace of each target PCRLB around 100, and equal importance given to tracking performance and sensor life times, then $W_b = 100$.

6.2.2.2 Collaboration

In the above formulation, we assumed that PCRLB at neighbors are the same as at FC a at last communication time steps, since we do not know the exact values. Even if we predict the PCRLB based on the tracklets that are received from the neighbors, the prediction might not be accurate, since neighbors use their neighbors' tracklets, which are not available at FC a, to update their tracks. In addition to PCRLB, the estimates may also be different at FCs. Note that current estimates will affect the future PCRLB calculation, which is a function of predicted states. In order to mitigate the effect of this problem, we may request additional information, such as current estimates, corresponding covariances and any additional information about next sensor change time, from neighbors just before deciding the new sensor subset.

As we mentioned in the previous section, avoiding clustering will help to improve the tracking performance. However, if we avoid the clustering, then there is a possibility that a sensor being used by more than one FC, which change their sensors at the same time. This problem can also be solved by collaboration. FCs can communicate their tentative selections with neighbors, and make the final selection based on neighbor's selection. However, obviously it will require more communication and computation.

6.3 Solution Technique

The above problems are NP hard combinatorial optimization problems [56]. Finding the optimal solution in real time is really hard, since the complete enumeration is not viable when the number of possibilities is very large. However, for large scale problems, an iterative local search technique is used to find a near optimal solution [43]. This is given as follows:

1. Select an initial solution as follows:

    - Select a sensor for first FC that gives optimal performance when only one sensor is used.
    - Select a second sensor for second FC that gives optimal performance when only two sensors are used including the first selected sensor.
    - Continue to select one sensor for each remaining FCs.
    - Continue to add one sensor at time in the order described above until all FCs reach their limits.

At each stage, a complete enumeration is performed to determine which sensor should be added.

2. Search for better solutions in the neighborhood until reach some pre-specified time limit:

   - Step 1: Set $f = 1$, $i_f = 1$ for $\forall f = 1, 2, \ldots, M$
   - Step 2: Remove the i_f-th sensor selected for FC f, and replace with another sensor.
     - If the new sensor set give better performance, make this as the selected set and
       * if $i_f = N_f$, then set $i_f = 1$, otherwise set $i_f = i_f + 1$,
       * if $f = M$, then set $f = 1$, otherwise set $f = f + 1$
       * go to Step 2.
     - otherwise perform Step 2 with another replacing sensor until no more sensor is remaining.
   - Step 3: For (6.12) with $W_b > 0$, remove a sensor at a time from the selected sensors and then perform the local search with reduced number of sensors as explained in Step 2. Note that we do not need to search for better solution by removing any sensor with $C_i = 0$.

Initially, form a neighborhood by swapping only one sensor from the current solution. If the current solution is optimal in the above neighborhood, then form a new neighborhood by swapping two sensors. Keep on increasing the number of sensors that are swapped for forming the neighborhood until the time limit is reached.

The local optimum of a certain neighborhood can be obtained quickly by swapping the sensors in the order of their rank, where the rank of each sensor is based on its individual performance.

6.4 Simulation Results

6.4.1 Synchronous sensor selection

In the simulation scenario, the number of FCs $F = 9$; the total number of sensors $N = 400$; each FC's capacity $n_f = 5$; the measurement interval is 30 seconds; sensor subsets are changed every five measurement steps; the measurements are the bearing from the sensor to target; measurement standard deviation $\sqrt{\Sigma} = 0.01$ radians; the field of view of the sensor $V = 2\pi$ radians; new targets can appear anywhere in the surveillance region. Communication range of FCs = 4000m, and their neighbors are shown in Figure 6.6. The weights are: $W_{k+1} = W_{k+2} = W_{k+3} = W_{k+4} = W_{k+5} = 1$; $W_{\text{new}} = 5$.

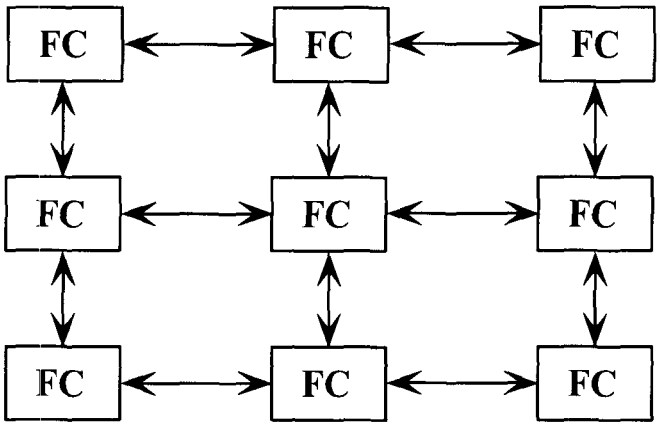


Figure 6.6: Neighbors of FCs

The sensors selected at the beginning, when no tracks are initialized, are shown

in Figure 6.7. Since we assume that new targets can appear anywhere in the region, sensors try to cover the whole region. Note that at least two sensors are needed to initialize a track since the measurement is bearing only. Hence, each point in the surveillance region must be covered by at least by two sensors. In addition to that, for better initialization of X and Y coordinates, sensors have to look from perpendicular directions. Since we used the PCRLB that explicitly considers new target births in sensor selection, the selected sensors, shown in Figure 6.7, provide better coverage of newly emerging targets thereby resulting in superior estimation results later on.

The sensors selected at $k = 13$, with 3 targets, are shown in Figure 6.8. Few sensors are assigned to track the existing targets while the others are assigned to initialize the new targets immediately and accurately. One of the targets has just appeared and, therefore, no sensor is assigned to that target particularly. However, that target is tracked by the sensors that are assigned for new targets.

The position RMSE of all the targets at each FC is shown in Figure 6.9. The minimum RMSE of each target over all the FCs is shown in Figure 6.10. The RMSEs of some targets are high at some FCs since those targets are far away from those FCs. However, each target is tracked accurately at every measurement step by at least one FC.

6.4.1.1 Comparison of proposed solution technique with GA and ACO

In order to compare the proposed solution technique with other heuristic algorithms, we have considered genetic algorithm [40] and ant colony optimization algorithm [69]; see appendices C and D for detail explanation of GA and ACO algorithm, respectively. The parameters of GA are: population size = 50, crossover ratio = 0.9, mutation ratio = 0.001 and elitism ratio = 0.04. The parameters used for the ACO algorithm are:

number of ants, $nAnts = 50$; pheromone trails lower and upper bounds, $\tau_{\min} = 0.01$, $\tau_{\max} = 1$, respectively; pheromone factor weight, $\alpha = 2$; pheromone evaporation rate, $\rho = 0.98$. In order to match the proposed algorithm with other two, we represent 50 sensor swaps by one cycle. All the algorithms are stopped after 500 cycles or 30 minutes, whichever occurs first. In addition, the proposed solution technique is stopped if we reach the local optimum with swapping only one sensor at a time. The performances of all the algorithms in a typical run are shown in Figure 6.11. The proposed algorithm finds the local optimum in 75 cycles and it takes 2.9 seconds. GA takes 21.8 seconds for 500 cycles and ACO takes 30 seconds for 198 cycles. The suboptimal values found by the proposed technique, GA and ACO are 961.9, 1023 and 1156, respectively. Hence proposed algorithm finds a better solution quickly compare to other two algorithms. ACO algorithm is the worst one, since it takes more time for each cycle and converges very slowly.

6.4.2 Asynchronous sensor selection

In the simulation the parameter settings are: the number of FCs $F = 9$; the total number of sensors $N = 400$; each FC's capacity $n_f = 5$; the measurement interval is 30 seconds; lower and upper limits of sensor change interval are $C_l = 6$ and $C_u = 8$ time steps; the measurements are the bearing from the sensor to target; measurement standard deviation $\sqrt{\Sigma} = 0.01$ radians; the field of view of the sensor $V = 2\pi$ radians; surveillance region is 10000m×10000m; new targets can appear anywhere in the surveillance region; New target weight $w_{\text{new}} = 1$; High priority area is a circle with center [6000m 4000m] and radius 1000m; Communication range of FCs = 4000m, and their neighbors are shown in Figure 6.12; the weights of the targets in the high priority area is 4; $D_{\max} = 1200$m, $B_{\max} = 100$ measurement steps; weight of sensor

life times $W_b = 0$ (later a different scenario is considered with $W_b > 0$ to see the effect of remaining sensor energies on sensor selection).

Selected sensors at three time steps are shown in Figures 6.13, 6.14 and 6.15, where • indicate the selected sensors and ◇ indicates the estimated target positions. In Figure 6.13, at $k = 4$, there is only one target in the surveillance region, and sensors of FC 4 are selected such that tracking performance of that target is optimized in the current and next few time steps. Three sensors of the FC 4 cover the target while other two are monitoring for new targets. There is no other target close to other FCs and sensors of those FCs are monitoring the surveillance region for new targets. Note that most of the sensors of FC 5 are covering high priority area.

In Figure 6.14, at $k = 8$, FC 5 changes its sensors, and four of its sensors are covering the target that is moving into its high priority area. Two more new targets appeared in the surveillance region and those are tracked by FCs using the sensors that were monitoring for new targets.

In Figure 6.15, at $k = 11$, FC 4 change its sensors for second time. Earlier, it allocated most of it sensors for first target (see Figure 6.13), which has now moved to the region of FC 5 and tracked by that FC. Currently, there is no target in its area, however there is a target in the area of FC 1, and moving towards its area. Hence, FC 4 allocates 3 sensors for that target and other two for monitoring new target.

Note that, even though we say that some sensors are allocated for a target and others are for new target, all the sensors give some information about new and already existing target. For an example, in Figure 6.15 even though 3 sensors are mostly monitoring existing target, those sensors can also monitor the some part of area for new targets. Similarly other two sensors, which are mainly monitoring for new targets, can generate measurements for existing target.

In order to see the effect of remaining sensor energies on sensor selection, a different scenario with 40 sensors and 2 FCs is considered. Sensors selected with $W_b = 0$, $W_b = 50$ and $W_b = 100$ are shown in Figures 6.16, 6.17 and 6.18, respectively, where $*$ indicate the target, and the numbers next to sensors indicate the remaining energies in term of number of measurement steps. When $W_b = 0$, sensors are selected to optimize the tracking performance without worrying about the remaining sensor energies (see Figure 6.16). With $W_b = 50$, only one sensor with low energy is selected (see Figure 6.17). However, the selected sensors still gives near optimal coverage. With $W_b = 100$, all the selected sensors have high energy (see Figure 6.18) and the tracking performance degrades slightly.

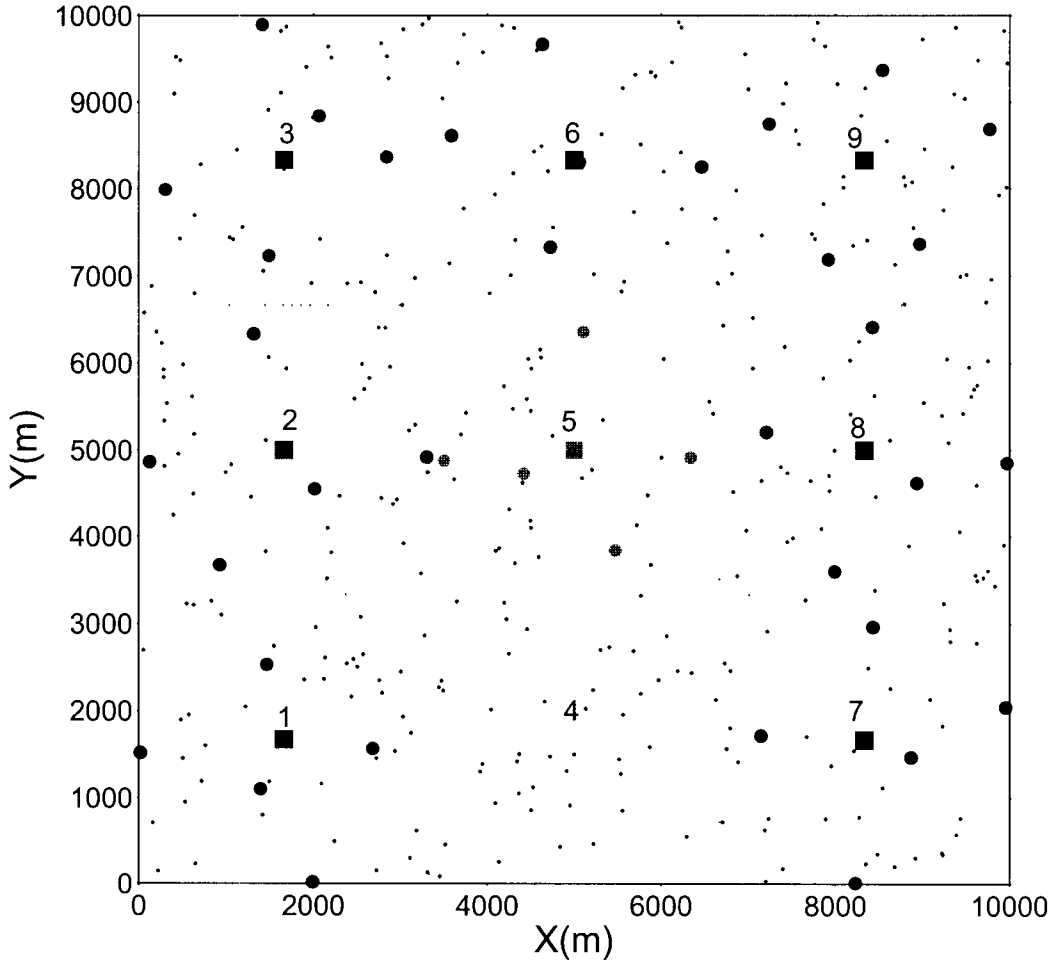


Figure 6.7: Selected sensors at $k = 1$

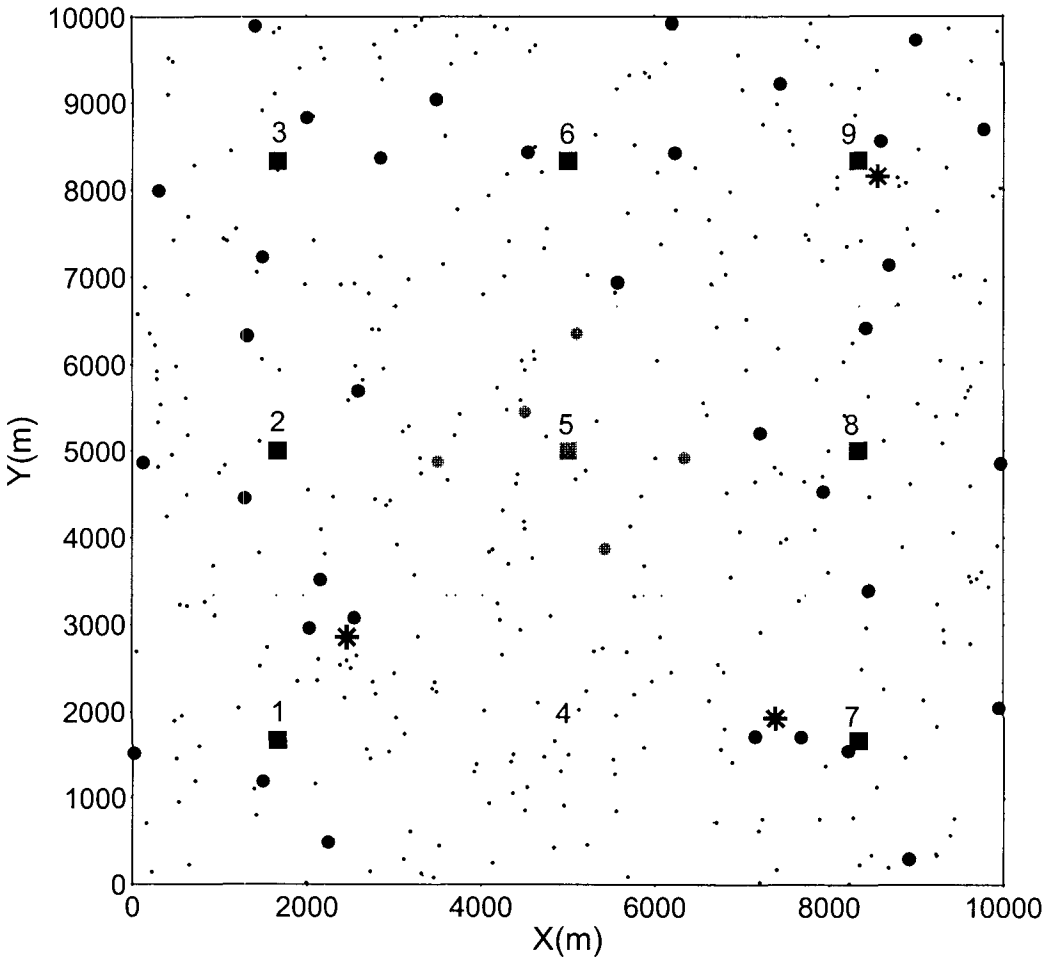


Figure 6.8: Selected sensors at $k = 13$

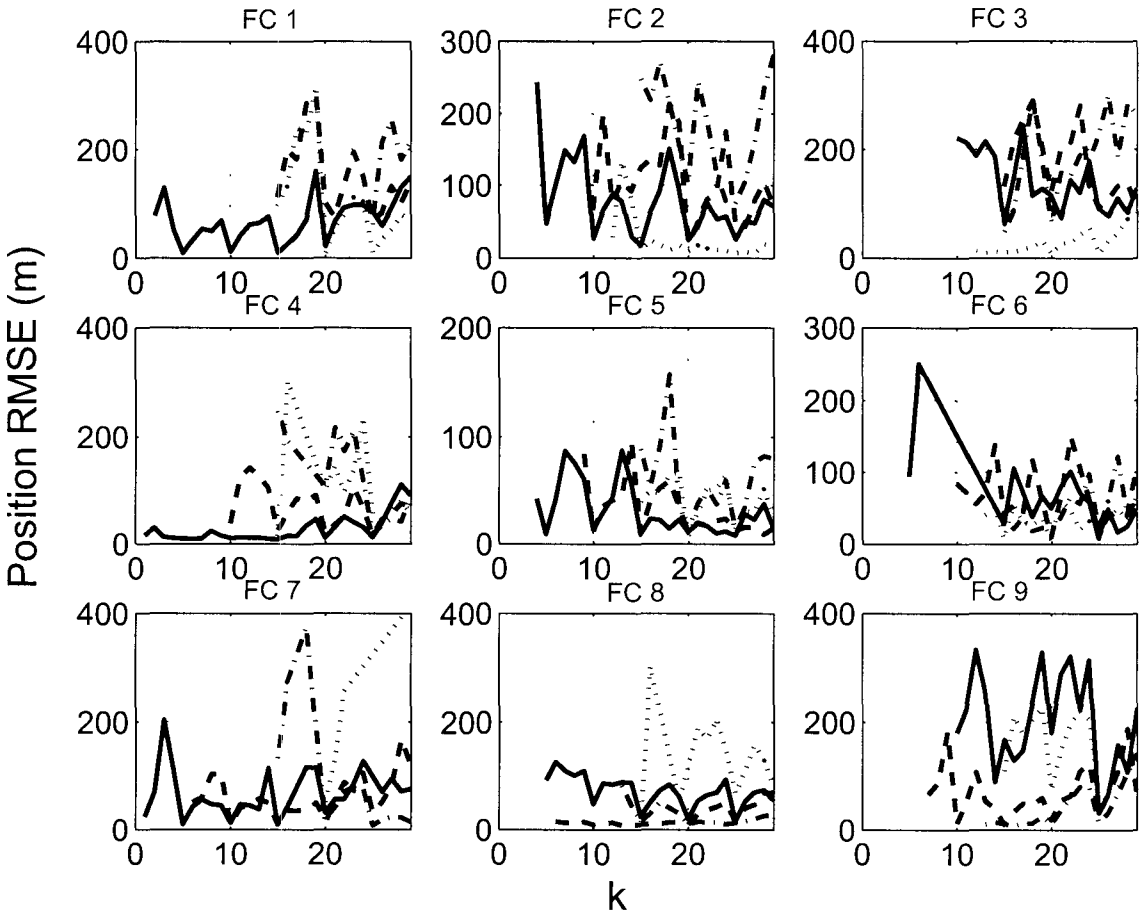


Figure 6.9: RMSEs at all FCs

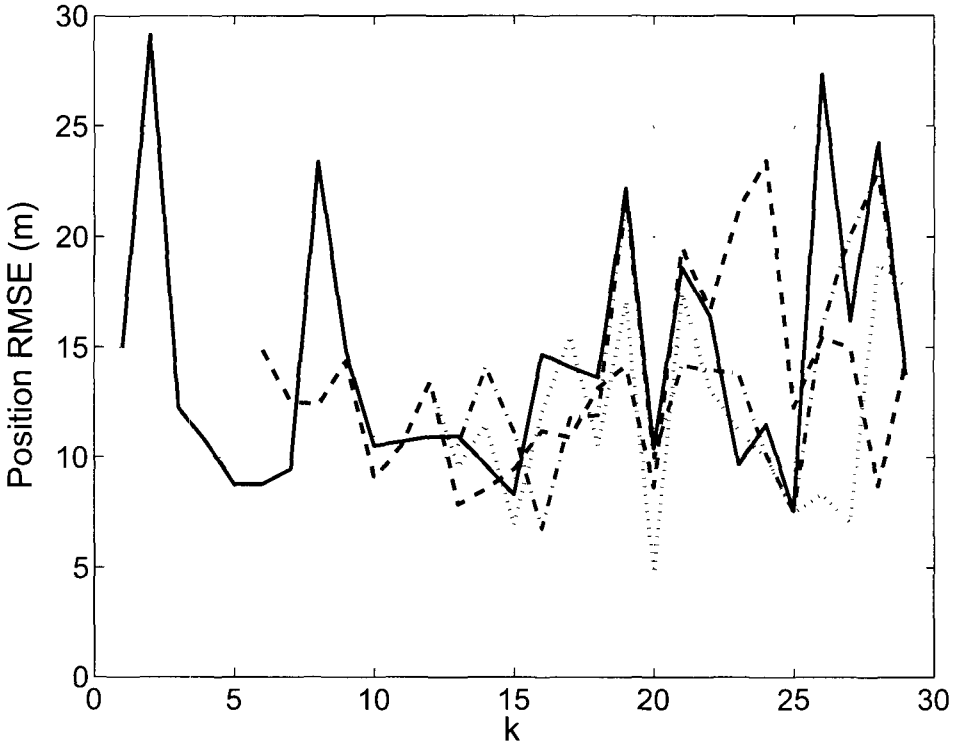


Figure 6.10: The minimum RMSEs over all the FCs

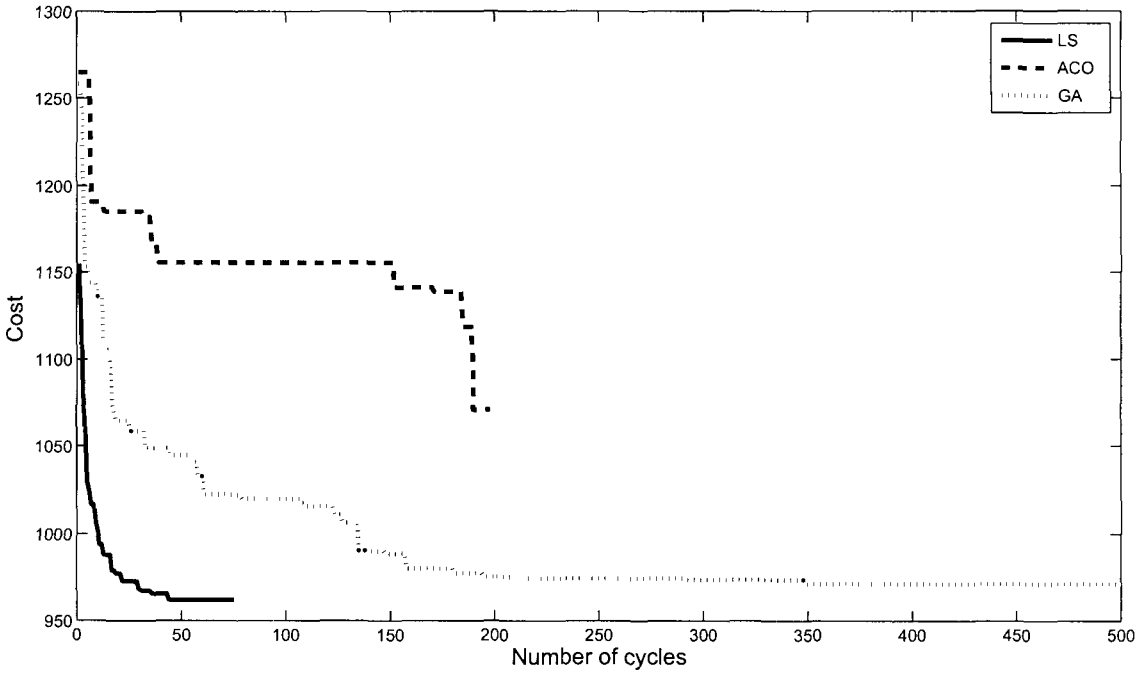


Figure 6.11: Comparison of local search (proposed technique), genetic algorithm and ant colony optimization

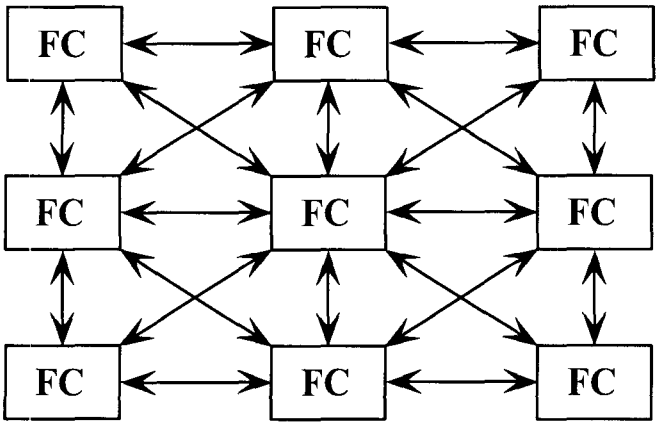


Figure 6.12: Neighbors of FCs

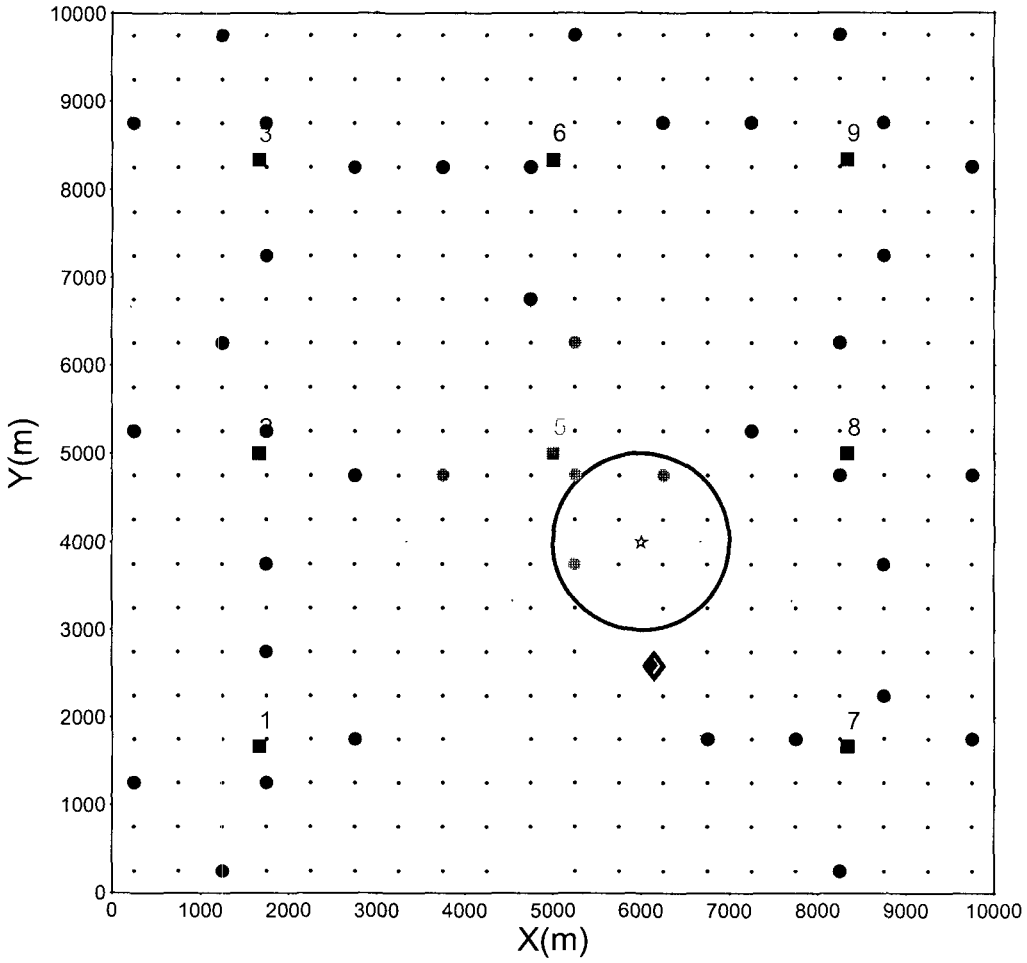


Figure 6.13: Selected sensors at $k = 4$

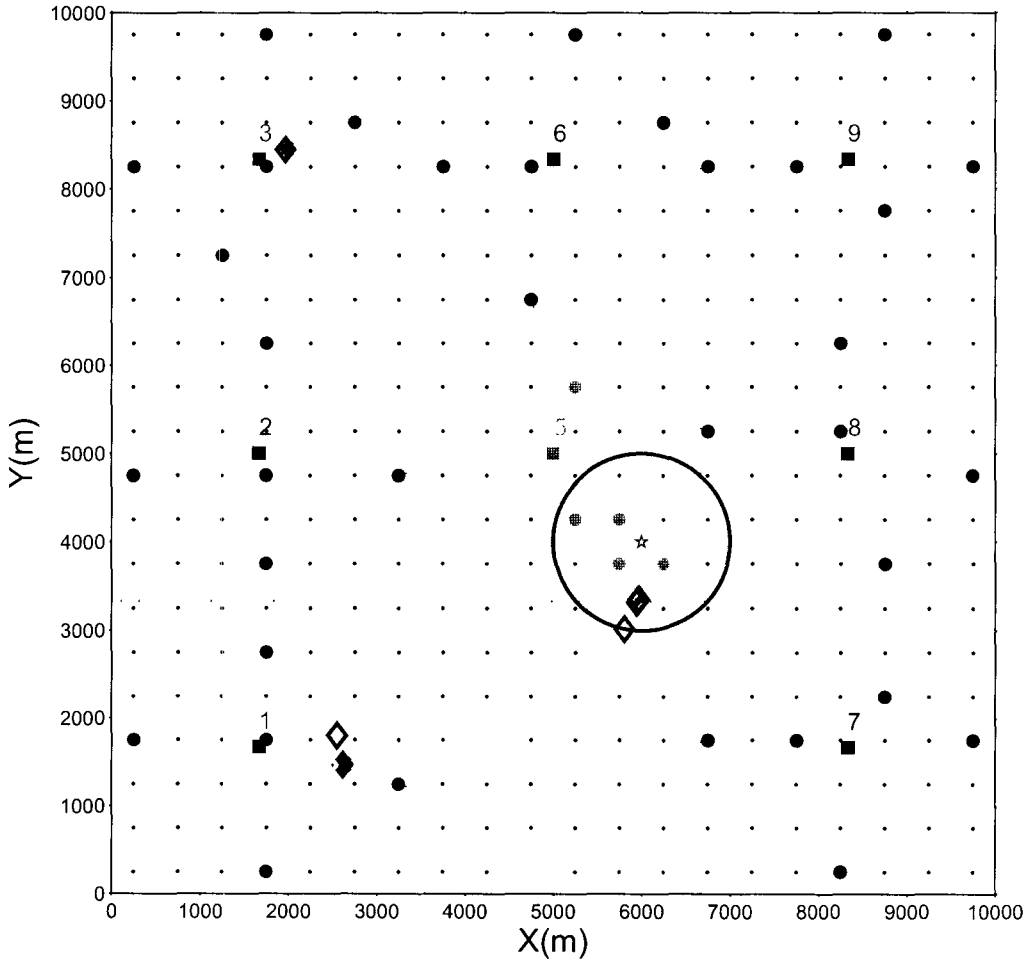


Figure 6.14: Selected sensors at $k = 8$

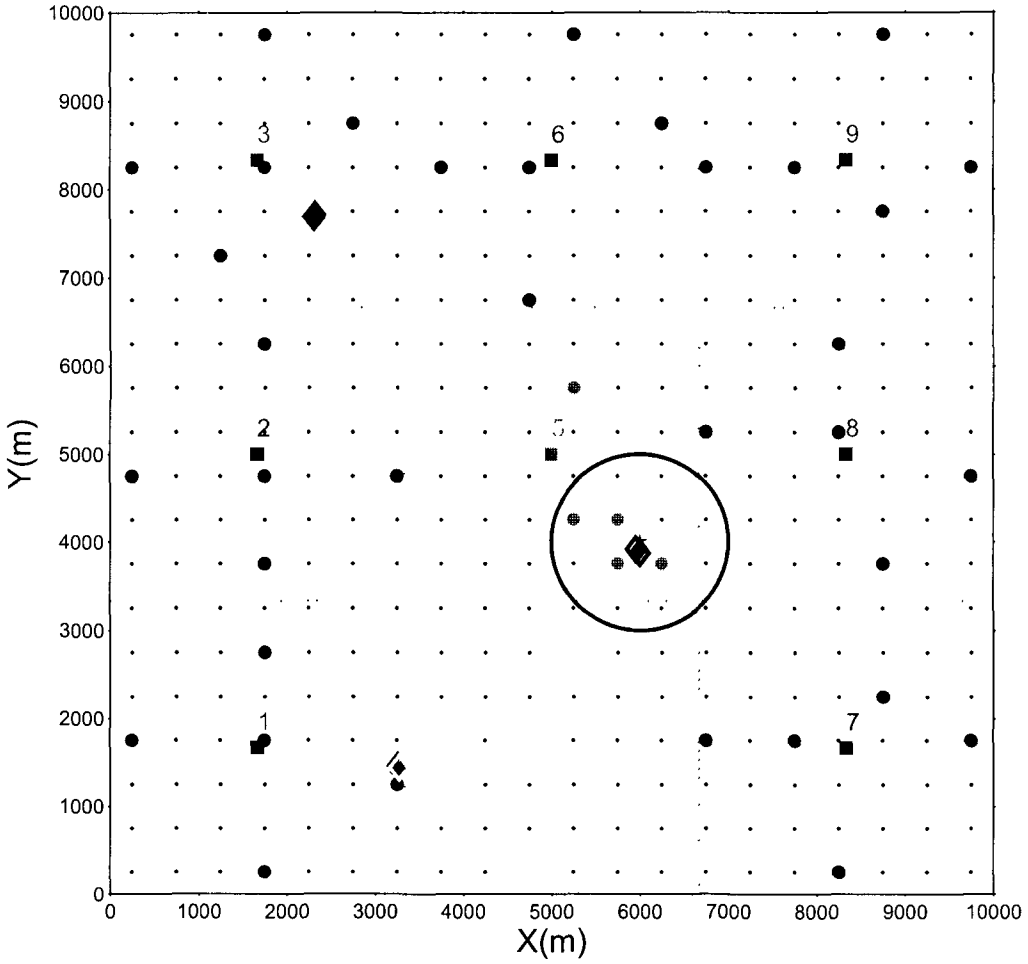


Figure 6.15: Selected sensors at $k = 11$

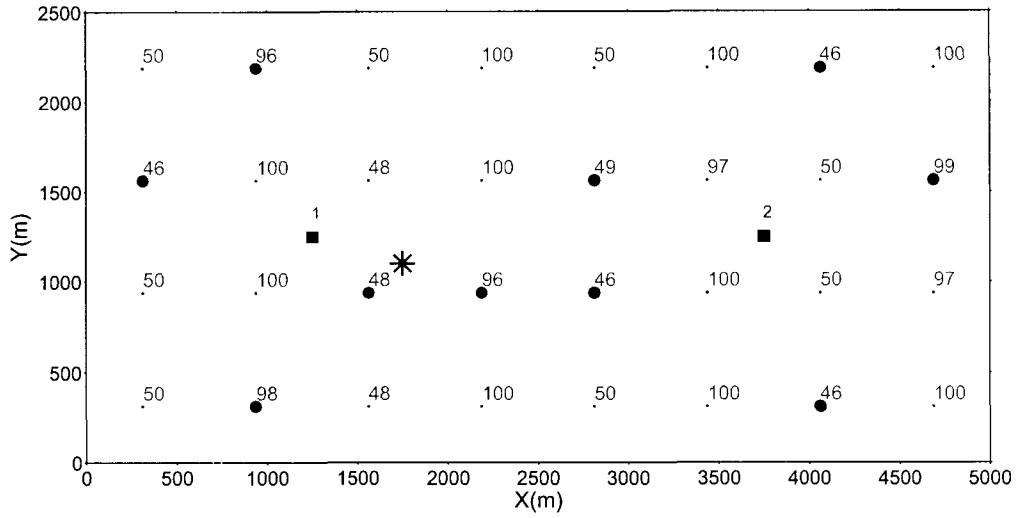


Figure 6.16: Selected sensors with $W_b = 0$

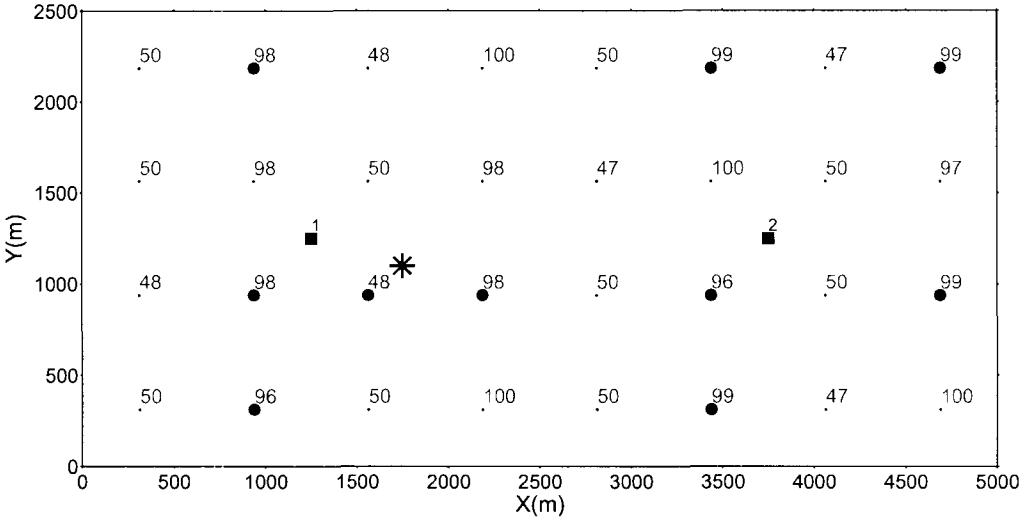


Figure 6.17: Selected sensors with $W_b = 50$

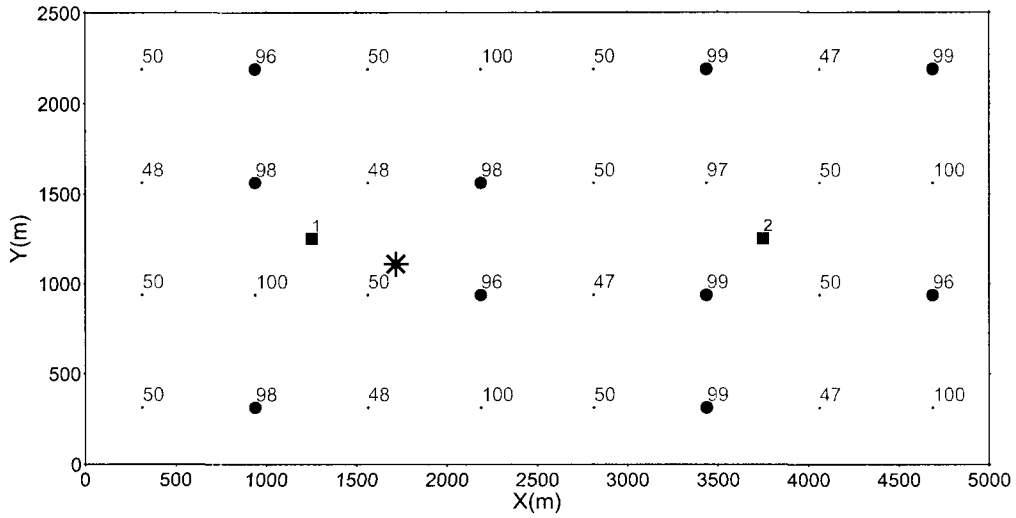


Figure 6.18: Selected sensors with $W_b = 100$

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we have considered the problem of sensor selection for centralized, distributed and decentralized architectures for multitarget tracking with an unknown associations of measurements to targets, and also with unknown and potentially time-varying number of targets.

The Posterior Cramér-Rao lower bound, which provides a measure of the optimal achievable accuracy of target state estimation, was used as the performance measure of the tracking systems. Only recently have expressions for multitarget PCRLBs been determined [42], and the necessary simulation techniques are computationally expensive. However, in this thesis we showed the existence of a multitarget information reduction factor (IRM), which can be calculated off-line in most cases. Additionally, we proposed some approximations that further reduce the computational load of PCRLB calculation.

In the centralized architecture, the problem is to select a sensor subset that gives

the optimal tracking performance through out the measurement steps. Although a particular sensor subset may be optimal at the time of initial activation, due to target motion, it may not remain optimal throughout the entire activation period. In order to get a sensor subset that remains optimal during the entire activation interval and not just at the time of activation, the predicted tracking performance over all the active time steps was considered. The varying number of targets was handled using a bi-criterion optimization technique, where one objective is to maximize the estimation accuracies of established targets and the other is to maximize the probability of detecting new targets. A search technique was presented to find a suboptimal solution in real time for large scale scenarios.

In the distributed architecture, the problem is to select subsets of sensors, assign them to local fusion centers, and assign the frequency and transmission power to each active sensor in order to maximize the tracking performance. The frequency channel limitation and the advantage of the variable transmitting power were not analyzed well in the literature. We derived the optimal formulation for the sensor management for the above problem based on the PCRLB. Finding the optimal solution to the above problem in real time is very hard in large scale scenarios. We presented an algorithm to find a suboptimal solution in real time by decomposing the original problem into four subproblems, which are easy to solve, without using the simple clustering algorithms that are used in the literature.

In the decentralized architecture, the problem is to select subsets of sensors for each FC in order to maximize the overall tracking performance. In the decentralized architecture, there is no central fusion center and each FC communicates only with its neighbors. Hence, the fusion centers do not have the global knowledge, and selecting the optimal sensor subsets, by considering all the FCs at once, is not feasible.

We proposed algorithms to select suboptimal sensor subsets by considering only the neighboring fusion centers for synchronous and asynchronous sensor selection scenarios. In addition to tracking performance, sensor life times, which are limited, were also considered in sensor selection. Several weights were used in the objectives to specify the important of each element, and we proposed how to select those weights scenario dependently. Finding the optimal solution in real time was really hard, and we proposed a solution technique based on iterative local search to find a suboptimal solution in real time for large scale problems. The performance of proposed solution technique was compared with genetic and ant colony optimization algorithms, and results showed that the proposed technique performs much better than those two algorithms.

7.2 Future Work

In this thesis, the PCRLB equations were derived under the assumption that the sensors make the measurements independently. In multistatic sensor networks, which have the potential to improve the tracking performance in ASW, we have few active sensors and many passive sensors [26, 44]. The passive sensors receive the signals that are emitted by the active sensors and reflected from the targets. In this case, sensors' measurements are dependent, and hence we need modification in PCRLB equations. In addition, if the active sensors are movable, then one has to decide the future locations of active sensors while selecting the sensor subsets from passive sensors. Hence, it will be an interesting problem to consider the path planning and sensor selection together.

In the PCRLB calculation for distributed and decentralized architectures, we did

not consider the misassociation of tracks/tracklets, which may occur in closely spaced targets scenarios. By incorporating these uncertainties in PCRLB calculation, we may be able to select better sensor subsets, which will reduce the misassociation.

Due to the additional difficulties of the decentralized architecture compared to distributed architecture, we assumed that the frequencies of each FC are predefined. However, we can improve the performance by allocating the frequencies dynamically, as we did for distributed architecture.

In this thesis, we proposed few collaboration techniques to improve the sensor selection. However, the required communication and computation loads are high, and may not be feasible in many cases. Then, we have to come up with systematic collaboration techniques, which require less communication and computation.

Appendix A

Calculation of Measurement Information, $J_{z_i}(k)$

The probability of having the association vector $a_k(i)$ conditioned on $m_k(i)$ measurements is given by [42]

$$p(a_k(i)|m_k(i)) = \frac{(m_k(i) - d(a_k(i)))!}{p(m_k(i))m_k(i)!}\mu_{FA}(m_k(i) - d(a_k(i))) \times \prod_{t=1}^{T}\left(\left(P_D^t(i)\right)^{D(a_k(i),t)}\left(1 - P_D^t(i)\right)^{(1-D(a_k(i),t))}\right) \quad \text{(A.1)}$$

where $d(a_k(i))$ is the number of detected targets and $D(a_k(i), t)$ takes 1 or 0 depending on whether target t is detected or not, respectively.

The probability of having $z_k(i)$ measurement conditioned on X_k, $m_k(i)$ and $a_k(i)$ is given by

$$p(z_k(i)|X_k, m_k(i), a_k(i)) = \prod_{j=1}^{m_k(i)} p_{a_k(j,i)}(z_k(j,i)) \quad \text{(A.2)}$$

where $p_{t(>0)}$ is the probability density function of a detection from target t and p_0 is the probability density function of the false alarm, given by

$$p_t(z_k(j,i)) = \frac{1}{|2\pi\Sigma_k|^{1/2}} \exp\Big(-\frac{1}{2}\big[z_k(j,i) - h_k^i(x_k^t)\big]'\Sigma_k^{-1}\big[z_k(j,i) - h_k^i(x_k^t)\big]\Big) \quad \text{(A.3)}$$

$$p_0(z_k(j,i)) = \frac{1}{V} \quad \text{(A.4)}$$

From (A.1) and (A.2), we can rewrite (3.23) as

$$\begin{aligned} p(z_k(i)|X_k, m_k(i)) = \frac{1}{p(m_k(i))} \sum_{a_k(i)} \Bigg(& \frac{(m_k(i) - d(a_k(i)))!}{m_k(i)!} \mu_{FA}\big(m_k(i) - d(a_k(i))\big) \\ & \times \prod_{t=1}^{T} \Big(\big(P_D^t(i)\big)^{D(a_k(i),t)} \big(1 - P_D^t(i)\big)^{(1-D(a_k(i),t))} \Big) \\ & \times \prod_{j=1}^{m_k(i)} p_{a_k(j,i)}(z_k(j,i)) \Bigg) \end{aligned} \quad \text{(A.5)}$$

It can be shown that

$$\nabla_{x_k^t} p_t(z_k(j,i)) = p_t(z_k(j,i)) \Big(\big[H_k^i\big]_t{}' \Sigma_k^{-1} \big(z_k(j_t,i) - h_k^i(x_k^t)\big)\Big) \quad \text{(A.6)}$$

where the (α,β)th element of matrix $\big[H_k^i\big]_t$ is given by (3.25).

From (A.5) and (A.6), it can be shown that

$$\begin{aligned}
&\nabla_{x_k^t} \ln\big(p(z_k(i)|X_k, m_k(i))\big) = \\
&\frac{1}{p(z_k(i)|X_k, m_k(i))}\frac{1}{p(m_k(i))} \sum_{\substack{a_k(i) \\ D(a_k(i),t)=1}} \left(\frac{(m_k(i) - d(a_k(i)))!}{m_k(i)!} \mu_{FA}\big(m_k(i) - d(a_k(i))\big) \right. \\
&\times \left(\prod_{\tau=1}^{T} \left(P_D^\tau(i)\right)^{D(a_k(\imath),\tau)} \left(1 - P_D^\tau(i)\right)^{(1-D(a_k(\imath),\tau))} \right) \left(\prod_{\jmath=1}^{m_k(\imath)} p_{a_k(\jmath,\imath)}(z_k(\jmath,i)) \right) \\
&\left. \times \left(\left[H_k^i\right]_t' \Sigma_k^{-1} \left(z_k(j_t,i) - h_k^i(x_k^t)\right) \right) \right) \qquad \text{(A.7)}
\end{aligned}$$

where j_t is the measurement number that is associated to target t, i.e., $a_k(j_t, i) = t$.

Now, from (3.21) and (A.7), the (t_1, t_2)th block matrix of matrix $J_{z_i}(m_k(i), k)$ can be written as

$$\begin{aligned}
\left[J_{z_i}(m_k(i),k)\right]_{t_1 t_2} &= \mathbb{E}\Big\{ \left[\nabla_{x_k^{t_1}} \ln\big(p(z_k(i)|X_k, m_k(i))\big)\right] \left[\nabla_{x_k^{t_2}} \ln\big(p(z_k(i)|X_k, m_k(i))\big)\right]' \Big\} \\
&= \mathbb{E}\Big\{ \left[H_k^i\right]_{t_1}' \left[Q_k^i(m_k(i))\right]_{t_1 t_2} \Sigma_k^{-1} \left[H_k^i\right]_{t_2} \Big| X_k \Big\} \qquad \text{(A.8)}
\end{aligned}$$

The expectations in (A.8) and (A.8) are over $(X_k,\, z_k)$ and X_k, respectively, and

$$\left[Q_k^i(m_k(i))\right]_{t_1 t_2} = \Sigma_k^{-1} \mathbb{E}\left[\frac{G_k^i(t_1) G_k^i(t_2)}{p(z_k(i)|X_k, m_k(i))^2} \right] \qquad \text{(A.9)}$$

where

$$\begin{aligned}
&\mathbb{E}\left[\frac{G_k^i(t_1) G_k^i(t_2)}{p(z_k(i)|X_k, m_k(i))^2} \right] \\
&= \int_{z_k(m_k(\imath),\imath) \in A} \cdots \int_{z_k(1,\imath) \in A} \frac{G_k^i(t_1) G_k^i(t_2)}{p(z_k(i)|X_k, m_k(i))} dz_k(1,i) \cdots dz_k(m_k(i),i) \qquad \text{(A.10)}
\end{aligned}$$

and

$$
\begin{aligned}
G_k^i(t) &= \frac{1}{p(m_k(i))} \sum_{\substack{a_k(i) \\ D(a_k(i),t)=1}} \left(\frac{(m_k(i) - d(a_k(i)))!}{m_k(i)!} \mu_{FA}(m_k(i) - d(a_k(i))) \right. \\
&\quad \times \left(\prod_{\tau=1}^{T} (P_D^\tau(i))^{D(a_k(i),\tau)} (1 - P_D^\tau(i))^{(1-D(a_k(i),\tau))} \right) \left(\prod_{j=1}^{m_k(i)} p_{a_k(j,i)}(z_k(j,i)) \right) \\
&\quad \left. \times \left(z_k(j_t, i) - h_k^i(x_k^t) \right) \right) \qquad \text{(A.11)}
\end{aligned}
$$

From (3.20), it can be shown that

$$
\left[J_{z_i}(k) \right]_{t_1 t_2} = \mathbb{E}\left\{ \left[H_k^i \right]_{t_1}' \left[Q_k^i \right]_{t_1 t_2} \Sigma_k^{-1} \left[H_k^i \right]_{t_2} \middle| X_k \right\} \qquad \text{(A.12)}
$$

where

$$
\left[Q_k^i \right]_{t_1 t_2} = \sum_{m_k(i)=0}^{\infty} p(m_k(i)) \left[Q_k^i(m_k(i)) \right]_{t_1 t_2} \qquad \text{(A.13)}
$$

and Q_k^i is the information reduction matrix (IRM) for sensor i and $\left[Q_k^i \right]_{t_1 t_2}$ is the (t_1, t_2)th block of the IRM. $\left[J_{z_i}(k) \right]_{t_1 t_2}$ gives the (t_1, t_2)th block of $J_{z_i}(k)$.

The integral (A.10) is analytically intractable and therefore numerical integration must be performed. In performing numerical integration there is then no need to expand the product of $G_k(t_1)$ and $G_k(t_2)$, as their values at each sample point can be calculated independently. Even though some simplification is possible after the expansion of the product, finding the values of each term separately and then taking the product is easier and faster.

A.1 Expanded forms

The expanded form of (3.22) is given by

$$p(m_k(i)) = \sum_{d=0}^{\min(T,m_k(i))} \mu_{FA}(m_k(i)-d) \times \prod_{\tau=1}^{T}(1-P_D^{\tau}(i)) \sum_{t_1=1}^{T} \sum_{t_2=t_1+1}^{T} \cdots \sum_{t_d=t_{d-1}+1}^{T} \left(\prod_{j=1}^{d} \frac{P_D^{t_j}(i)}{1-P_D^{t_j}(i)} \right) \quad \text{(A.14)}$$

The expanded form of (A.5) is

$$p(z_k(i)|X_k, m_k(i)) = \frac{1}{p(m_k(i))} \prod_{t=1}^{T}(1-P_D^t(i)) \left(\sum_{d=0}^{\min(T,m_k(i))} \left(\frac{(m_k(i)-d)!}{m_k(i)!} \mu_{FA}(m_k(i)-d) \left(\frac{1}{V}\right)^{m_k(i)-d} \times \sum_{t_1=1}^{T} \sum_{t_2=t_1+1}^{T} \cdots \sum_{t_d=t_{d-1}+1}^{T} \sum_{j_1=1}^{m_k(i)} \sum_{\substack{j_2=1 \\ j_2 \neq j_1}}^{m_k(i)} \cdots \sum_{\substack{j_d=1 \\ j_d \neq j_{d-1} \\ j_d \neq j_1}}^{m_k(i)} \left(\prod_{l=1}^{d} \frac{P_D^{t_l}(i)}{(1-P_D^{t_l}(i))} p_{t_l}(z_k(j_l,i)) \right) \right) \right) \quad \text{(A.15)}$$

The expanded form of (A.11) is

$$
\begin{aligned}
G_k^n(t) \;=\; & \frac{1}{p(m_k(i))} P_D^t(i) \prod_{\substack{\tau=1 \\ \tau \neq t}}^{T} (1 - P_D^\tau(i)) \\
& \times \left(\sum_{d=1}^{\min(T, m_k(i))} \left(\frac{(m_k(i)-d)!}{m_k(i)!} \mu_{FA}(m_k(i)-d) \left(\frac{1}{V} \right)^{m_k(i)-d} \right. \right. \\
& \times \sum_{\substack{\tau_1=1 \\ \tau_1 \neq t}}^{T} \sum_{\substack{\tau_2=\tau_1+1 \\ \tau_2 \neq t}}^{T} \cdots \sum_{\substack{\tau_{d-1}=\tau_{d-2}+1 \\ \tau_{d-1} \neq t}}^{T} \sum_{j_0=1}^{m_k(i)} \sum_{\substack{j_1=1 \\ j_1 \neq j_0}}^{m_k(i)} \cdots \sum_{\substack{j_{d-1}=1 \\ j_{d-1} \neq j_{d-2} \\ j_{d-1} \neq j_0}}^{m_k(i)} (p_t(z_k(j_0, i)) \\
& \left. \left. \times \prod_{l=1}^{d-1} \frac{P_D^{\tau_l}(i) p_{\tau_l}(z_k(j_l, i))}{1 - P_D^{\tau_l}(i)} \left(z_k(j_0, i) - h_k^i(x_k^t) \right) \right) \right) \right)
\end{aligned}
\tag{A.16}
$$

Appendix B

Local Search

A local search algorithm improves the current estimate of the optimum point by searching for better solutions in a local neighborhood of the current solution and stops if the neighborhood does not contain an improving solution [1, 56]. A local search is sensitive to (the quality of) the initial solution and the choice of the neighborhood. The selection of the initial solution to the sensor subset problem is discussed in **Step 1** of section 4.3. The choice of the neighborhood has a very significant influence on the efficiency of local search heuristics and it also determines the computational time of one iteration [43]. A neighborhood structure to our problem is given next.

Let $C_s = [C_s(1), C_s(2), \ldots, C_s(n)]$ be the current solution and $N_S(i) = [n_{si}(1), n_{si}(2), \ldots, n_{si}(n)]$ for $i = 1, 2, \ldots, (Nn)$ be the i-th neighbor. In addition, q and r are the quotient and the remainder of the division of $(i-1)$ by n, respectively, and $b_s(q)$ is the q-th best sensor. Then

- If $b_s(q+1) \neq C_s(k)$ for $k = 1, 2, \ldots, n$

$$n_{si}(j) = \begin{cases} b_s(q+1) & \text{if} \quad j = r+1 \\ C_s(j) & \text{else} \end{cases} \tag{B.1}$$

  for $j = 1, 2, \ldots, n$.

- Else

$$N_s(i) = NULL \tag{B.2}$$

Here, the $NULL$ set is not considered as an element of the neighborhood. In the above structure, only one sensor is changed from the current solution [64]. The size of the above neighborhood structure is $(N-n)n = O(Nn)$. Another neighborhood structure can be created by changing at most two sensors from the current solution. The size of that structure will be $O(N^2n^2)$. The second neighborhood structure might provide a better local optimum but it would take much longer to search. Hence, we use the first structure for real-time capability. However, if time permits we can use the second structure to search for better solution, after finding the possible best solution with one sensor swapping.

There are two possible implementations for a local search procedure [56], namely, 1) the first improving move ("first-fit") implementation moves the solution to an improving neighbor first encountered during the search and 2) the best improving move ("best-fit") implementation, on the other hand, moves the solution to the neighbor that improves the solution the most. In either case, the running time of each iteration is bounded by $O(Nn)$ if the function evaluation time remains constant. The "first-fit" strategy may result in a smaller improvement of the current solution than the "best-fit" strategy. While every neighborhood needs to be searched completely

with the "best-fit" strategy, only the final neighborhood needs to be searched with the "first-fit" strategy. Thus, the former is used in our solution to find the local optimum, resulting in some computational saving.

The order in which the neighborhood is searched also affects the computational time. Intuitively, the probability of having a high rank (based on the individual performance) sensor in the optimal sensor subset is higher than having a randomly selected sensor. Hence neighborhood is searched by swapping the sensors in the ranking order.

The complete algorithm is as follows:

- **Step 1**: Set the initial solution as a new subset, find the objective function value (LB) and assign $i = 1$, $j = 1$ and $iter = 0$.
- **Step 2**: Check whether the i-th best sensor is already in the new subset. If it is in the new subset, set $i = i + 1$ and repeat step 2.
- **Step 3**: Remove the j-th sensor from the new subset and replace it by the i-th best sensor and form a temporary subset. Then find the function value (TLB) using the temporary subset.
- **Step 4**:
  - If ($iter <$ maximum allowable iterations (a predetermined value))
    - if ($TLB < LB$)
      - change the temporary subset as the new subset and set $LB = TLB$.
      - $i = 1$, $j = 1$ and $iter = iter + 1$.
      - go to step 2.

  - else
    - if $(j < n)$, set $j = j + 1$ and go to step 3.
    - else if $(i < N)$, set $j = 1$, $i = i + 1$ and go to step 2.
    - else, go to step 5.

- **Step 5**: Finish.

The running time of the above algorithm can also be controlled by changing the stopping criterion. For example, the maximum running time can be bounded by a constant, instead of bounding the number of iterations.

The computational time of the above local search algorithm is mainly determined by the number of iterations and the time used to calculate the first improving neighbor of the current solution. The computation time of cost of existing targets is very much higher than that of new targets. Hence, a function evaluation can be done in $O(Tln)$ time. But, once we have calculated the function value of the given initial solution, the function value of a neighbor, formed by swapping only one sensor, can be found in $O(Tl)$ time. For example, if sensor a is replaced by sensor b and we have the summation of the values of $J_{z_i}(\tau)$ for the current sensors for all τ, then the new sum can be calculated by just subtracting $J_{z_a}(\tau)$ and adding $J_{z_b}(\tau)$. Since there are $O(Nn)$ neighbors to test, the running time of each iteration is bounded by $O(TNln)$.

Appendix C

Genetic Algorithm

Genetic algorithm is a class of learning algorithms based on parallel search for an optimal solution, and is inspired by the biological processes of evolution [36, 52, 53]. The parallel searches, which are performed synchronously in time steps, are called generations. In each generation, a certain number of search paths are maintained, and these are called individuals. The whole set of individuals in a generation is referred to as the population. The main idea in genetic algorithms is to preserve and create variations of the individuals that seem most promising and remove the others.

The general framework of GA is shown in Figure C.1 [52], and its steps are explained below with an example:

- Initialize the population: individuals of initial population are generated at random. To our problem initial individuals, which are strings of bits [52], are found by randomly selecting n_f sensors for each FC. For an example, individual v_k, for a scenario with 3 FCs, 5 sensors are available for each FC and each FC can handle 2 sensors, is $00101 \cdot 10010 \cdot 01010$.

- Find the fitness: a fitness function must be defined to give a score to each

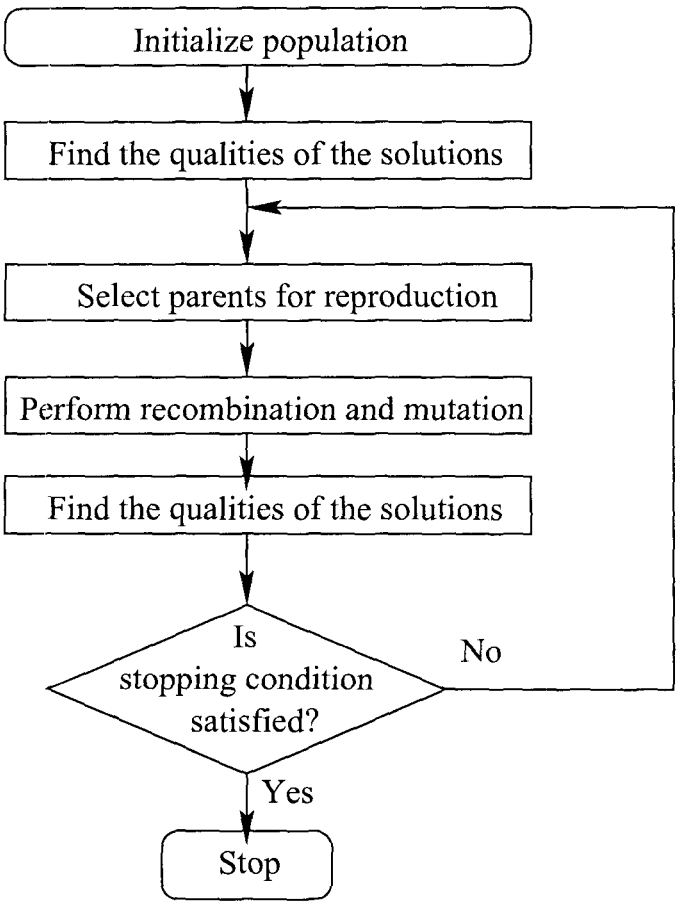


Figure C.1: Framework of genetic algorithm

individual in the current population. Fitness of each individual, v_k, is given by

$$\frac{1}{\text{func}(\text{v}_\text{k}) - 0.5 * \text{bestSoFar}} \tag{C.1}$$

where func refers the function defined in (6.1) and bestSoFar refers the best function value found so far.

- Select parent: selection is based on the fitness. Individuals that have high score will most likely be selected to reproduce, whereas those with low scores will be discarded. Selection probability p_k for each individual v_k is given by

$$p_k = \frac{f(v_k)}{\sum_{i=1}^{\text{popSize}} f(v_i)} \tag{C.2}$$

where popSize is the size of the population.

- Crossover: it is an operator that combines two parents to generate new a individual. Randomly chooses a locus and exchanges the subsequences before and after that locus between two chromosomes to create two new individuals. For an example, consider the following parents and crossover at position 7:

$$\begin{array}{cc} 00101 \cdot 10010 \cdot 01010 & \\ & \Rightarrow \\ \mathbf{10010 \cdot 11000 \cdot 01010} & \end{array} \quad \begin{array}{c} 00101 \cdot 10\mathbf{000} \cdot \mathbf{01010} \\ \\ \mathbf{10010} \cdot \mathbf{11}010 \cdot 01010 \end{array}$$

Crossover operation is normally performed with a certain probability, called crossover ratio. For an example, if the crossover ratio is 0.9, 90% of the new individuals are formed by crossover and remaining 10% are just copied.

- Mutate: change the new individual by flipping each bit with some probability called mutation ratio. For an example, only one bit is flipped and that is 13th bit:

$00101 \cdot 10000 \cdot 01010 \rightarrow 00101 \cdot 10000 \cdot 01\mathbf{1}10$

- Repair: It is a process that rectifies infeasible individuals that may be produced during crossover and mutation [63]. Flip the random bits from the selected bits such that the individual become feasible. For an example, in the above individual that obtained after mutation, only 1 sensor is selected for second FC and 3 sensors are selected for third FC. In order to make the individual feasible, randomly select a 0 bit from 10th to 15th bits and change to 1, similarly select a 1 bit from 16th to 20th bits and change to 0:
  $00101 \cdot 10000 \cdot 01110 \rightarrow 00101 \cdot 1000\mathbf{1} \cdot 011\mathbf{0}0$

Elitism, in which we first copies a few of the top scored individuals to the new population and then continues generating the rest of the population by crossover, can also be used to improve the performance, and prevents loosing the few best found solutions [40].

Choosing the size of the population can be tricky since a small population size provides an insufficient sample size over the space of solutions and large population requires a lot of evaluation and will be slow.

Appendix D

Ant Colony Optimization

Ant colony optimization algorithms are stochastic search procedures [21, 69]. The first ACO algorithms were proposed in the early 1990's [20]. These algorithms are inspired by the behavior of real ants. Real ants are capable of finding the shortest path between their nest and a food source. While moving, ants leave a chemical (pheromone) trail on the ground. When choosing their path, they tend to choose the path with strong pheromone. This indirect communication among the ants via pheromone trails enables them to find the shortest path effectively. That is, information about an ant's experience is communicated via pheromone and utilized by other ants.

The pseudocode of ACO algorithm is given in Figure D.1.

The detailed algorithm, that is used for our problem, is explained below:

1. Initialize the pheromone trails:

   $\tau_{\text{vertex}}(s_i) = \tau_{\max}$; $\tau_{\text{clique}}(s_i, s_j) = \tau_{\max}$

2. Construct the solutions:

   for each ant $a \in [1...n\text{Ants}]$

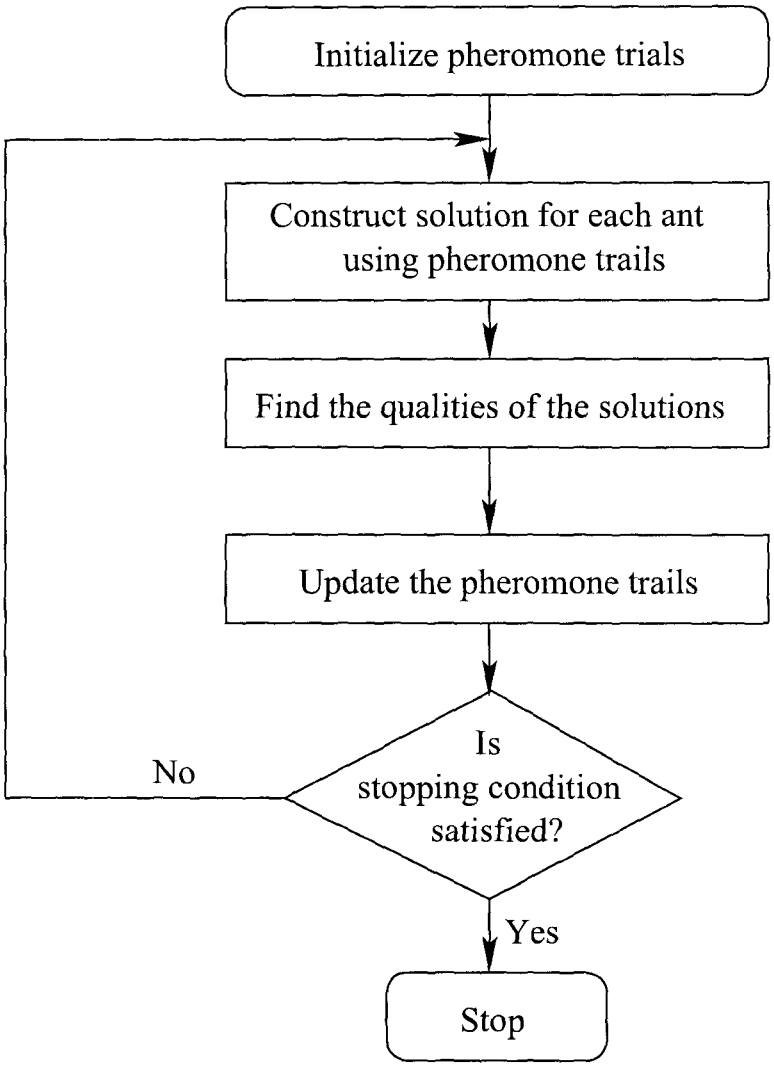


Figure D.1: ACO Algorithm

Set *Candidates* = all the sensors that belong to itself and the neighbors

Choose the first sensor s_i from *Candidates* with probability

$$p(s_i) = \frac{\tau_{\text{vertex}}(s_i)^{\alpha}}{\sum_{j \in Candidates} \tau_{\text{vertex}}(s_j)^{\alpha}} \tag{D.1}$$

Set $S_a = \{s_i\}$ and remove s_i from the *Candidates*

while *candidates* $\neq \emptyset$

Choose a sensor, s_i, from *Candidates* with probability

$$p(s_i, S_a) = \frac{\zeta(s_i, S_a)^{\beta} \sum_{j \in S_a} \tau_{\text{clique}}(s_i, s_j)^{\alpha}}{\sum_{l \in Candidates} \zeta(s_l, S_a)^{\beta} \sum_{j \in S_a} \tau_{\text{clique}}(s_l, s_j)^{\alpha}} \tag{D.2}$$

Add s_i to S_a and remove s_i from the *Candidates*

if s_i belongs to FC f, and the number of sensors belonging to this FC from S_a is equal to n_f

Remove all the sensors that belong to FC f from *Candidates*

end if

end while

Apply local search (optional)

end for

3. Trial Update:

Order the solutions according to their qualities.

Update the pheromone trails using mBest solutions as follows [9]:

$$\begin{aligned}\tau_{\text{vertex}}(s_i) &= (1-\rho)\cdot\tau_{\text{vertex}}(s_i) \\ &\quad + \rho\cdot\sum_{\substack{b=1 \\ s_i\in S_b}}^{m\text{Best}} \frac{1/(1+\text{func}(S_b)-\text{func}(S_{\text{BestSoFar}}))}{\sum_{c=1}^{m\text{Best}}(1/(1+\text{func}(S_c)-\text{func}(S_{\text{BestSoFar}})))} \qquad \text{(D.3)}\end{aligned}$$

$$\begin{aligned}\tau_{\text{clique}}(s_i,s_j) &= (1-\rho)\cdot\tau_{\text{clique}}(s_i,s_j) \\ &\quad + \rho\cdot\sum_{\substack{b=1 \\ s_i\in S_b \\ s_j\in S_b}}^{m\text{Best}} \frac{1/(1+\text{func}(S_b)-\text{func}(S_{\text{BestSoFar}}))}{\sum_{c=1}^{m\text{Best}}(1/(1+\text{func}(S_c)-\text{func}(S_{\text{BestSoFar}})))} \qquad \text{(D.4)}\end{aligned}$$

4. Termination:

   If maximum number of cycles or allocated time is reached, return $S_{\text{BestSoFar}}$. Otherwise go to step 2.

In (D.2), ζ is the heuristic factor that evaluates the promise of sensor s_i based on the sensors selected so far, S_a. This is an optional factor. It is not used always, i.e., $\zeta(s_i, S_a) = 1$, in our algorithm since it was difficult to find a computationally efficient way to define this factor.

The parameters used in ACO are: number of ants, nAnts; pheromone trails upper and lower bounds, $\tau_{\max}$ and $\tau_{\min}$, respectively; pheromone evaporation rate, ρ; pheromone factor weight, α; heuristic factor weight, β.

The quality solution from ACO can be improved by increasing the value of nAnts. However, after a certain limit, quality will not improve significantly, but computation time will increase significantly. In addition, this value must be calculated experimentally. In ACO, $\tau_{\max}$ and $\tau_{\min}$ are used to avoid premature stagnation of search, i.e, a situation where all ants construct the same solution over and over again. Diversification can be emphasized either by decreasing the value of α or by increasing the value of ρ. These values should be set according to the time available to find the solution.

Bibliography

[1] E. Aarts and J. K. Lenstra, *Local Search in Combinatorial Optimization*, John Wiley and Sons, Chichester, England, 1997.

[2] J. S. Abel, "Optimal sensor placement for passive source localization", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, pp. 2927–2930, Albuquerque, New Mexico, April 1990.

[3] M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking", *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 173–188, February 2002.

[4] Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS Publishing, 1995.

[5] Y. Bar-Shalom, X. Li and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, Wiley, 2001.

[6] R. Bejar, B. Krishnamachari, C. Gomes and B. Selman, "Distributed constraint satisfaction in a wireless sensor tracking system", *Proceedings of the Workshop on Distributed Constraint Reasoning*, pp. 81–90, Seattle, WA, 2001.

[7] P. Berman, G. Calinescu, C. Shah and A. Zelikovsky, "Power efficient monitoring

management in sensor networks", *Proceedings of the IEEE Wireless Communications and Networking Conference*, vol. 4, pp. 2329–2334, Atlanta, GA, March 2004.

[8] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*, Artech House, 1999.

[9] C. Blum and M. Dorigo, "The hyper-cube framework for ant colony optimization", *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, vol. 34, no. 2, pp. 1161–1172, April 2004.

[10] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2006.

[11] P. Brown and T. Kirby-Smit, "Operational field trials of GPS equipped sonobuoys", *Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GPS-96*, vol. 2, pp. 1553–1561, Kansas City, MO, September 1996.

[12] A. L. Buczak and V. Jamalabad, "Self organization of a heterogeneous sensor network by genetic algorithm", *Intelligent Engineering Systems Through Artificial Neural Networks*, C. H. Dagli, C. L. Chen, M. Akay (eds.), vol. 8, pp. 259–264, ASME Press, New York, NY, 1998.

[13] A. L. Buczak, Y. Jin, H. Darabi and M. Jafari, "Genetic algorithm based sensor network optimization for target tracking", *Proceedings of the Artificial Neural Networks in Engineering Conference*, vol. 9, pp. 349–354, St. Louis, MO, November 1999.

[14] N. Bulusu, D. Estrin, L. Girod, and J. Heidemann, "Scalable coordination for wireless sensor networks: self-configuring localization systems", *Proceedings of the Sixth International Symposium on Communication Theory and Applications*, Ambleside, UK, July 2001.

[15] C. Y. Chong, S. Mori, W. H. Barker and K. C. Chang, "Architectures and algorithms for track association and fusion", *IEEE Aerospace and Electronic Systems Magazine*, vol. 15, no. 1, pp. 5–13, January 2000.

[16] T. H. Chung, V. Gupta, B. Hassibi, J. Burdick and R. M. Murray, "Scheduling for distributed sensor networks with single sensor measurement per time step", *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1, pp. 187–192, New Orleans, LA, April 2004.

[17] CPLEX: *ILOG CPLEX 8.0 User's Manual*, ILOG S.A., July 2002.

[18] S. Dasika, S. Vrudhula, K. Chopra and R. Srinivasan, "A framework for battery-aware sensor management", *Proceedings of Conference on Design, Automation, and Test in Europe*, pp. 962–967, February 2004.

[19] S. Deb, M. Yeddanapudi, K. R. Pattipati and Y. A. Bar-Shalom, "A generalized S-D assignment algorithm for multisensor-multitarget state estimation", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, pp. 523–38, April 1997.

[20] M. Dorigo, "Optimization, learning and natural algorithms", *PhD thesis*, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1992. [in Italian]

[21] M. Dorigo and T. Stutzle, "Ant Colony Optimization", MIT Press, Cambridge, MA, 2004.

[22] A. Doucet, S. J. Godsill and C. Andrieu. "On sequential Monte Carlo sampling methods for Bayesian filtering", *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[23] A. Doucet, B. Vo, C. Andrieu and M. Davy, "Particle filtering for multitarget tracking and sensor management", *Proceedings of the 5th International Conference on Information Fusion*, vol. 1, pp. 474–481, Annapolis, Maryland, July 2002.

[24] O. E. Drummond, "A hybrid sensor fusion algorithm architecture and tracklets", *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, vol. 3163, pp. 485–502, San Diego, CA, July 1997.

[25] O. E. Drummond, "Track and tracklet fusion filtering using data from distributed sensors", *Proceeding of the Estimation, Tracking and Fusion: A Tribute to Yaakov Bar-Shalom*, pp. 167–186, Monterey, CA, May 2001.

[26] O. Erdinc, P. Willett and S. Coraluppi, "Multistatic sensor placement: a tracking approach", *Proceedings of the 9th International Conference on Information Fusion* Florence, Italy, July 2006.

[27] Z. Ezziane, "Solving the 0/1 knapsack problem using an adaptive genetic algorithm", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 16 , no. 1, pp. 23–30, January 2002.

[28] N. J. Gordon, D. J. Salmond and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation", *IEE Proceedings Part F: Radar and Signal Processing,* , vol. 140, no. 2, pp. 107–113, April 1993.

[29] B. Grocholsky, "Information-theoretic control of multiple sensor platforms", *PhD*

thesis, Department of Aerospace, Mechatronic and Mechanical Engineering, The University of Sydney, Australia, 2002.

[30] D. Gu, I. Postlethwaite and Y. Kim, "A comprehensive study on flight path selection algorithms" *Proceeding of the IEE Seminar on Target Tracking: Algorithms and Applications*, pp. 77–90, Austin Court, Birmingham, UK, March 2006.

[31] M. Hawkes and A. Nehorai, "Effects of sensor placement on acoustic vector-sensor array performance", *IEEE Journal of Oceanic Engineering*, vol. 24, no. 1, pp. 33–40, January 1999.

[32] M. L. Hernandez, T. Kirubarajan and Y. Bar-Shalom, "Multisensor resource deployment using posterior Cramér-Rao bounds", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, no. 2, pp. 399–416, April 2004.

[33] M. L. Hernandez, A. D. Marrs, N. J. Gordon, S. Maskell and C. M. Reed, "Cramér-Rao bounds for non-linear filtering with measurement origin uncertainty", *Proceedings of the 5th International Conference on Information Fusion*, vol. 1, pp. 18–25, Annapolis, Maryland, July 2002.

[34] C. F. Haung, "The coverage problem in a wireless sensor network", *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications*, pp. 115–121, San Diego, CA, September 2003.

[35] K. L. Hoffman, "Combinatorial optimization: current successes and directions for the future", *Journal of Computational and Applied Mathematics*, vol. 124, no. 1-2, pp. 341–360, December 2000.

[36] J. H. Holland, *Adaptation in Natural and Artijlcial Systems*, MIT Press, Cambridge, MA, 1992.

[37] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1985.

[38] P. R. Horridge and M. L. Hernandez, "Performance bounds for angle-only filtering with application to sensor network management", *Proceedings of the 6th International Conference on Information Fusion*, pp. 695–703, Cairns, Australia, July 2003.

[39] M. Howard, D. Payton and R. Estkowski, "Coalitions for distributed sensor fusion", *Proceedings of the 5th International Conference on Information Fusion*, vol. 1, pp. 636–642, Annapolis, MD, July 2002.

[40] M. Hristakeva and D. Shrestha, "Solving the 0–1 Knapsack problem with genetic algorithms", *Proceedings of the 37th Midwest Instruction and Computing Symposium*, Morris, MN, April 2004.

[41] C. Hue and J.-P. Le Cadre, "Sequential Monte Carlo methods for multiple target tracking and data fusion", *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 309–325, February 2002.

[42] C. Hue, J.-P. Le Cadre and P. Prez, "Performance analysis of two sequential Monte Carlo methods and posterior Cramér-Rao bounds for multitarget tracking", *Proceedings of the 5th International Conference on Information Fusion*, vol. 1, pp. 464–473, Annapolis, Maryland, July 2002.

[43] J. Hurink, "Solving complex optimization problems by local search", Habilitationsschrift, http://wwwhome.math.utwente.nl/~hurinkjl/papers/habil.pdf, Universität Osnabrück, 1999.

[44] B. I. Incze and S. B. Dasinger, "A Bayesian method for managing uncertainties relating to distributed multistatic sensor search", *Proceedings of the 9th International Conference on Information Fusion*, Florence, Italy, July 2006.

[45] K. Johansson, K. Jored and P. Svensson, "Submarine tracking using multi-sensor fusion and reactive planning for the positioning of passive sonobuoys", *Proceedings of Hydroakustik*, Stockholm, Sweden, September 1997.

[46] M. Kalandros and L.Y. Pao, "Randomization and super-heuristics in choosing sensor sets for target tracking applications", *Proceedings of the IEEE Conference on Decision and Control*, pp. 1803–1808, Phoenix, AZ, 1999.

[47] I. Katzela and M. Naghshineh, "Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey", *IEEE Personal Communications*, vol. 3, no. 3, pp. 10–31, June 1996.

[48] C. Kreucher, K. Kastella and A. O. Hero III, "Multitarget tracking using the joint multitarget probability density", *IEEE Transaction on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1396–1414, October 2005.

[49] V. Krishnamurthy, "Algorithms for optimal scheduling and management of hidden markov model sensors", *IEEE Transactions on Signal Processing*, vol. 50, no. 6, pp. 1382–1397, June 2002.

[50] M. E. Liggins, C. Y. Chong, I. Kadar, M. G. Alford, V. Vannicola and S. Thomopoulos, "Distributed fusion architectures and algorithms for target tracking", *Proceedings of the IEEE*, vol. 85, no. 1, pp. 95–107, January 1997.

[51] J. Liu, J. Reich, P. Cheung and F. Zhao, "Distributed group management in

sensor networks: algorithms and applications to localization and tracking", *Proceedings of Second International Workshop on Information Processing in Sensor Networks*, pp. 113–128, Palo Alto, CA, April 2003.

[52] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd edition, Springer-Verlag, Berlin, NY, 1996.

[53] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, MA, 1996.

[54] P. J. Modi, H. Jung, M. Tambe, W. M. Shen, and S. Kulkarni, "A dynamic distributed constraint satisfaction approach to resource allocation", *Lecture Notes in Computer Science*, vol. 2239, pp. 685–700, 2001.

[55] L. Y. Pao, M. Kalandros and J. Thomas, "Controlling target estimate covariance in centralized multisensor systems", *Colorado Advanced Software Institute (CASI)*, Final Report, 1997.

[56] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization Algorithms and Complexity*, Prentice Hall, 1988.

[57] K. R. Pattipati, R. L. Popp and T. Kirubarajan, "Survey of assignment techniques for multitarget tracking", In Y. Bar-Shalom and W. D. Blair, (Eds.), *Multisensor-Multitarget Tracking: Applications and Advances*, vol. 3, Boston, MA: Artech House, 2000.

[58] M. Perillo and W. Heinzelman, "Optimal sensor management under energy and reliability constraints", *Proc. IEEE Wireless Communications and Networking Conference*, vol. 3, pp. 1621–1626, March 2003.

[59] A. Petcu and B. Faltings, "A value ordering heuristic for local search in distributed resource allocation", *Lecture Notes in Computer Science*, vol. 3419, pp. 86–97, Springer Verlag, Heidelberg, Germany, 2005.

[60] R. Popoli, "The sensor management imperative", in *Multitarget-Multisensor Tracking: Applications and Advances Vol. II*, Y. Bar-Shalom (editor), Artech House, 1992.

[61] K. Punithakumar, T. Kirubarajan, and M. L. Hernandez, "Multisensor deployment using PCRLBs, incorporating sensor deployment and motion uncertainties", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 4, pp. 1474–1485, October 2006.

[62] Y. Qu, Q. Pan and J. Yan, "Flight path planning of UAV based on heuristically search and genetic algorithms", *Proceedings of the 31st Annual Conference of IEEE Industrial Electronics Society*, pp. 45–49, Raleigh, NC, November 2005.

[63] G. R. Raidl, "An improved genetic algorithm for the multiconstrained 0–1 knapsack problem", *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 207–211, Anchorage, AK, May 1998.

[64] M. G. C. Resende and R. Werneck, "On the implementation of a swap-based local search procedure for the p-median problem", Technical Report TD-5E4QKA, AT&T Labs Research, September 2002.

[65] B. Ristic, S. Zollo and S. Arulampalam, "Performance bounds for maneuvering target tracking using asynchronous multi-platform angle-only measurements", *Proceedings of the 4th International Conference on Information Fusion*, Montreal, Quebec, Aug. 2001.

[66] T. Sathyan, A. Sinha and T. Kirubarajan, "Passive geolocation and tracking of an unknown number of emitters", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 2, pp. 740–750, April 2006.

[67] J. Shin, L. J. Guibas and F. A. Zhao, "A Distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks", *Second International Workshop on Information Processing in Sensor Networks*, pp. 223–238, Palo Alto, CA, April 2003.

[68] A. Sinha, T. Kirubarajan, and Y. Bar-Shalom, "Autonomous search, tracking and classification by multiple cooperative UAVs," *Proceedings of SPIE Conference on Signal Processing, Sensor Fusion, and Target Recognition XV*, vol. 6235, Orlando, FL, April 2006.

[69] C. Solnon and D. Bridge, "An ant colony optimization meta-heuristic for subset selection problems", In *System Engineering using Particle Swarm Optimization*, Nadia Nedjah and Luiza Mourelle editors, Nova Science publisher, March 2006.

[70] A. Spyropoulos, C. S. Raghavendra and V. K. Prasanna, "A distributed algorithm for waking-up in heterogeneous sensor networks", *Second International Workshop on Information Processing in Sensor Networks*, pp. 609–624, Palo Alto, CA, April 2003.

[71] D. F. Tello Gamarra, T. F. Bastos-Filho, M. Sarcinelli-Filho, C. M. Soria and R. Carelli, "Optical flow calculation using data fusion with decentralized information filter", *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2853–2858, Barcelona, Spain, April 2005.

[72] P. Tichavsky, "Posterior Cramér-Rao bounds for adaptive harmonic retrieval", *IEEE Transactions on Signal Processing*, vol. 43, no. 5, pp. 1299–1302, May 1995.

[73] P. Tichavsky, C. H. Muravchik and A. Nehorai, "Posterior Cramér-Rao bounds for discrete-time nonlinear filtering", *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, May 1998.

[74] R. Tharmarasa, "Large-scale optimal sensor array management for multitarget tracking", *Masters Thesis*, Department of Electical and Computer Engineering, McMaster University, 2003.

[75] H. Van Trees, *Detection, Estimation and Modulation Theory*, vol. I, Wiley, New York, 1968.

[76] P. Vanheeghe, E. Duflos, P. E. Dumont and V. Nimier, "Sensor management with respect to danger level of targets", *Proceedings of the 40th IEEE Conference on Decision and Control*, vol. 5, pp. 4439–4444, December 2001.

[77] J. E. Wieselthier, G. D. Nguyen and A. Ephremides, "Resource management in energy-limited, bandwidth-limited, transceiver-limited wireless networks for session-based multicasting", *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 39, no. 5, pp. 113–131, June 2002.

[78] N. Xiong and P. Svensson, "Multi-sensor management for information fusion: issues and approaches", *Information Fusion*, vol. 3, no. 1, pp. 163–186, June 2002.

[79] H. Yang and B. Sikdar, "A protocol for tracking mobile targets sensor networks

using sensor network", *Proceedings of IEEE Workshop on Sensor Network Protocols and Applications*, pp. 71–81, Anchogare, AK, May 2003.

[80] X. Yu, K. Niyogi, S. Mehrotra and N. Venkatasubramanian, "Adaptive target tracking in sensor networks", *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference*, San Diego, CA, January 2004.

[81] F. Zhao, J. Shin and J. Reigh, "Information-driven dynamic sensor collaboration for tracking applications", *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 61–72, March 2002.

[82] H. Zhang, "Two-dimensional optimal sensor placement", *IEEE Transactions on Systems, Man and Cybernetics*, vol. **25**, no. 5, pp. 781–791, May 1995.

[83] X. Zhang and P. K. Willett, "Cramer-Rao bounds for discrete time linear filtering with measurement origin uncertainty", *Proceedings of the Workshop on Estimation, Tracking and Fusion: A Tribute to Yaakov Bar-Shalom*, pp. 546–561, Monterey, CA, May 2001.

[84] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces", *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 1293–1303, San Francisco, April 2003.