A COMPARATIVE STUDY OF PARTICLE FILTERING FOR TARGET TRACKING IN BINARY SENSOR NETWORKS

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree of

MASTER OF TECHNOLOGY

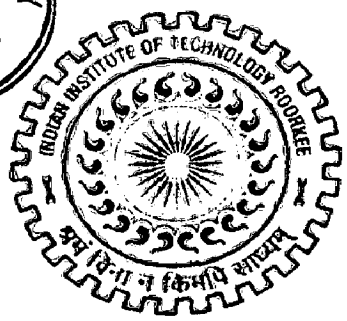
in

ELECTRONICS AND COMMUNICATION ENGINEERING

(With Specialization in Communication Systems)

By

V. S. S. N PRASAD MARNI



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE -247 667 (INDIA)

JUNE, 2010

CANDIDATE'S DECLARATION

I hereby declare that the work, which is presented in this dissertation report entitled, "**A COMPARATIVE STUDY OF PARTICLE FILTERING FOR TARGET TRACKING IN BINARY SENSOR NETWORKS**" towards the partial fulfillment of the requirements for the award of the degree of **Master of Technology** with specialization in **Communication Systems,** submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee (India) is an authentic record of my own work carried out during the period from July 2009 to June 2010, under the guidance of **Dr.D.K.MEHRA, Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee.**

I have not submitted the matter embodied in this dissertation for the award of any other Degree or Diploma.

Date: 15-06-2010
Place: Roorkee

M.V.S.S.N Prasad
V. S. S. N PRASAD MARNI

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date: 15-06-2010
Place: Roorkee

D mehra
Dr. D. K. MEHRA,
Professor, E&C Department,
IIT Roorkee,
Roorkee – 247 667 (India).

ACKNOWLEDGEMENTS

I would like to extend my heartfelt gratitude to my guide, **Dr. D. K. MEHRA** for his able guidance, valuable suggestions and constant attention. It is his constant encouragement that inspired me throughout my dissertation work. I consider myself fortunate to have my dissertation done under him.

Thanks are due to the Lab staff of Signal Processing Lab, Department of Electronics and Communication Engineering, IIT Roorkee for providing necessary facilities.

I am greatly indebted to all my course mates, who have graciously applies themselves to the task of helping me with ample morale support and valuable suggestions. Most of all I would like to thank my family. Finally, I would like to extend my gratitude to all those persons who directly or indirectly contributed towards this work.

V. S. S. N PRASAD MARNI

ABSTRACT

Many problems in signal processing requires the estimation of the state that changes over time using a sequence of noisy measurements made on the system. In reality most of the applications involve non-linear and non-Gaussian features. Estimating the state of the system in nonlinear non Gaussian environment is highly intractable. This includes the classical problem, of target tracking in wireless sensor network. In this report binary sensor networks are considered as a special case of wireless sensor networks for tracking the target. Unlike sensors considered in traditional tracking approaches, binary sensors provide only one bit of data indicating presence or absence of a target in the sensing range. The signals that reach the fusion center of these networks are therefore binary signals embedded in noise, and they pose challenging problems for recovering the sensed information by the sensors.

Particle filtering algorithm provides a numerical solution to the non-tractable recursive Bayesian estimation problem in case of non-linear and non-Gaussian systems like target tracking in binary sensor networks.

In this dissertation work, we have used the state space approach for deriving the particle filtering algorithm for non linear estimation problem. Various versions of particle filtering algorithms have been used for estimating the state of the system and have shown that the choice of auxiliary particle filter gives reasonably good results as compared to other particle filters when the process noise is equal to greater than the measurement noise.

Two particle filtering algorithms have been considered for processing of the binary data at the fusion center namely, auxiliary particle filtering and cost reference particle filtering. Unlike auxiliary particle filtering (APF), cost-reference particle filtering does not rely on any probabilistic assumptions about the dynamic system. Finally, the imperfect nature of the wireless communication channel between sensors and the fusion center is incorporated in the particle filter tracking algorithm known as channel aware particle filtering. For simulation MATLAB is used and it is demonstrated through simulation results that APF outperforms the cost reference particle filtering considerably in the presence of fading environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

APF	Auxiliary Particle Filter
ASIR	Auxiliary Sampling Importance Resampling
AWGN	Additive White Gaussian Noise
BSN	Binary Sensor Network
CRPF	Cost Reference Particle Filter
CSW	Cumulative Sum of Weights
DR	Dead-Reckoning
ED	Energy Detector
EKF	Extended Kalman Filter
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
INS	Inertial Navigation System
MAP	Maximum a Posteriori
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
MEMS	Micro Electro Mechanical Systems
NRZ	Non Return to Zero
PF	Particle Filter
PCRB	Posterior Cramer-Rao Bound
QOS	Quality of Service
RMSE	Root Mean Square Error
RPF	Regularized Particle Filter
SIR	Sampling Importance Resampling

SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
SNR	Signal to Noise Ratio
WSN	Wireless Sensor Network

Chapter 1
INTRODUCTION

In many application areas which include signal processing, statistics, communications, and econometrics, it is required to estimate the state of the system from a noisy measurements made on the system. Many practical applications which involve non-linear and non-Gaussian features namely, localization of robots, estimating noisy digital communications signals, image processing, land vehicle navigation and aircraft tracking using radar measurements etc., requires the estimation of the state which is highly intractable [1], [2], [3]. This includes the classical problem, in signal processing literature, of target tracking. Target tracking is an important element of surveillance, guidance, or obstacle avoidance systems, whose role is to determine the position, and movement of targets. The problem of target tracking in wireless sensor networks (WSN) is a typical nonlinear sequential estimation problem. The tracking problem is sometimes referred to as target motion analysis and its objective is to track the kinematics of a moving target using the noise corrupted measurements. The fundamental building block for recursively estimating the target state of a tracking system is a filter [3]. Bayesian filtering is the most commonly used framework for tracking applications. In Bayesian filtering, the tracking algorithm recursively calculates the belief in the state based on the observations, namely the posterior distribution. It is known that Kalman filter provides an optimal solution to the Bayesian sequential problem for linear/Gaussian systems. For nonlinear problems Kalman filter cannot provide the optimal solution.

In nonlinear case, the most common approach is extended Kalman filter (EKF) [2], which approximate the model by linearized version of it using Taylor series expansion and then use the optimal Kalman filter with this approximate model. This filter works well for weekly nonlinear system. For systems with high degree of non linearity further terms in Taylor series should be considered, which results in additional computational complexity [2]. The EKF assumes the Gaussian nature which is not always satisfied with the real systems. Real systems commonly include non-linear and non-Gaussian elements as well as high dimensionality. Numerical integration [1], [3] is

another approach that could be used in non-linear, non-Gaussian cases but it is computationally too expensive to be used in practical applications.

Although the idea of Monte Carlo simulation [3] originated in the late 1940s, its popularity in the field of filtering started in 1993[4]. Roughly speaking, Monte Carlo technique [1], [6] is a kind of stochastic sampling approach aiming to tackle the complex systems which are analytically intractable. The power of Monte Carlo methods is that they can approximate the solutions of difficult numerical integration problems [3]. These methods fall into two categories, namely, Markov chain Monte Carlo (MCMC) methods for batch signal processing and sequential Monte Carlo (SMC) methods for adaptive signal processing. One of the attractive merits of the sequential Monte Carlo approaches lies in the fact that they allow on-line estimation by combining the powerful Monte Carlo sampling methods with Bayesian inference at an expense of reasonable computational cost. Sequential Monte Carlo methods found limited use in the past, except for the last decade, primarily due to their very high computational complexity and the lack of adequate computing resources of the time. The fast advances of computers in the recent years and outstanding potential of particle filters have made them a very active area of research. In particular, the sequential Monte Carlo approach has been used in parameter estimation and state estimation. This SMC approach is known variously as particle filtering [2], [3], [5] boot strap filtering, the condensation algorithm, interacting particle approximations and survival of the fittest.

Particle filtering is an emerging and powerful methodology particularly useful in dealing with non linear and non-Gaussian problems based on the concept of sequential importance sampling and Bayesian theory [2], [3]. Particle filters are sequential Monte Carlo methods which can be applied to any state space model and which generalizes the Kalman filtering methods. The basic idea of particle filter is to use a number of independent random variables called particles, sampled directly from the state space, to represent the posterior probability, and update the posterior by involving the new observations; the "particle system" is properly located, weighted, and propagated recursively according to the Bayesian rule [1], [2], [3]. Particle filtering methods have the potential to use the increasing computational power available in today's technological market to push filtering theory beyond its challenges.

Particle filters have found application in many areas such as channel equalization, estimation and coding, wireless channel tracking, artificial intelligence, speech enhancement, speech recognition and machine learning, tracking in WSN, land vehicle navigation applications, GPS/INS integration etc. Some of these applications are briefly explained below.

Speech Enhancement and Recognition:

With the advent of ubiquitous computing, a significant trend in human-computer interaction is the use of a range of multimodal sensors and processing technologies to observe the user's environment. These allow users to communicate and interact naturally, both with computers and other users. Some of the applications are advanced computing environments, instrumented meeting rooms and seminar halls facilitating remote collaboration. The solution to the problem of distant speech acquisition in multiparty meetings, using multiple microphones and cameras is particle filtering. Speech Enhancement and Recognition in Meetings with an Audio-Visual Sensor Array is presented in [17].

Land vehicle navigation application:

To provide an accurate positioning, the land vehicle navigation applications are based on global positioning system (GPS). However, the GPS is not always an ideal vehicle positioning system in urban areas because the satellite coverage may be poor due to multipath reflections caused by high buildings, tunnels etc. Therefore, the GPS based systems are enhanced with a set of dead-reckoning (DR) sensors. The addition of a digital road map allows locating the vehicle continuously and helps the driver to get the best path. A hybrid filter is used for solving the fusion problem of the GPS, odometer, and digital road map measurements in the presence of GPS outages. A Hybrid Particle Approach for GNSS applications With Partial GPS outages is presented in [18].

Target tracking in WSN:

The objective of tracking in WSN is to recursively estimate the target state based on the received local sensor data, which is a highly nonlinear problem. So the best solution for tracking the target in WSN is Particle Filter. In [15], Target tracking by particle

filtering in binary sensor networks is presented, however wireless channel imperfections have not been considered as part of the tracking problem. In a target tracking scenario where a large number of wireless sensors are deployed in a particular area, we cannot always guarantee a line-of-sight between sensors and the fusion center. So we have to consider the channel imperfections between the sensors and the fusion center. Incorporating the imperfect nature of the wireless communication channels between sensors and the fusion center in the tracking algorithm is called as channel aware particle filtering [16].

1.1 Wireless Sensor Networks

Wireless sensor networks (WSN) [8]; [9] have gained worldwide attention in recent years, particularly with the proliferation in Micro-Electro-Mechanical Systems (MEMS) technology which has facilitated the development of smart sensors. Their use may span a vast range of fields, and their effectiveness is already being felt both in commercial and military applications as well as in the further development of science and engineering. Wireless sensor networks typically have little or no infrastructure. It basically consists of a number of sensor nodes (few tens to thousands) and a fusion centre. The sensor nodes will sense and measure signals that provide information about an event or events of interest based on some local decision process and send this information to the fusion centre. The fusion centre combines the received information to obtain estimates about the observed phenomenon. These sensor nodes are low power devices equipped with one or more sensors, a processor, memory, a power supply, a radio, and an actuator. Since the sensor nodes have limited memory and are typically deployed in difficult-to-access locations, a radio is implemented for wireless communication to transfer the data to a base station (e.g., a laptop, a personal handheld device, or an access point to a fixed infrastructure). Battery is the main power source in a sensor node. Secondary power supply that harvests power from the environment such as solar panels may be added to the node depending on the appropriateness of the environment where the sensor will be deployed. Depending on the application and the type of sensors used, actuators may be incorporated in the sensors.

One of the most important constraints on sensor nodes is the low power consumption requirement [10], [11]. Sensor nodes carry limited, generally

irreplaceable, power sources. Therefore, while traditional networks aim to achieve high quality of service (QoS) provisions, sensor network protocols must focus primarily on power conservation. They must have inbuilt trade-off mechanisms that give the end user the option of prolonging network lifetime at the cost of lower throughput or higher transmission delay.

Current WSNs [8] are deployed on land, underground, and underwater. Depending on the environment, a sensor network has different challenges and constraints. There are five types of WSNs: terrestrial WSN, underground WSN, underwater WSN, multi-media WSN, and mobile WSN.

Mobile WSNs: Mobile WSNs consist of a collection of sensor nodes that can move on their own and interact with the physical environment. In military surveillance and tracking, mobile sensor nodes can collaborate and make decisions based on the target.

1.1.1 Applications

Sensor network applications [8] can be classified into two categories: monitoring and tracking (see Fig.1.1). Monitoring applications include indoor/outdoor environmental monitoring, health and wellness monitoring, power monitoring, inventory location monitoring, factory and process automation, and seismic and structural monitoring. Tracking applications include tracking objects, animals, humans, and vehicles. While there are many different applications, below a few applications are given that have been deployed and tested in the real environment.

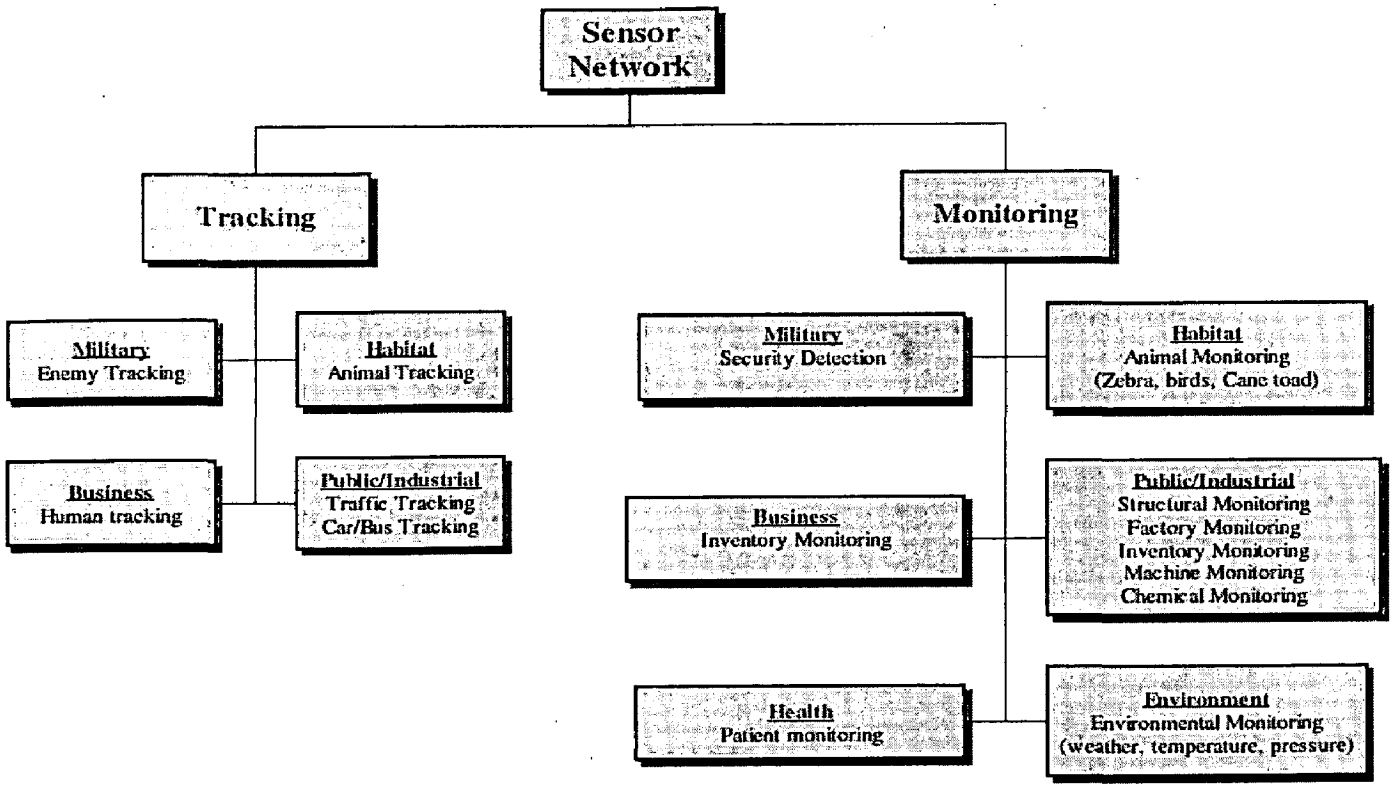


Fig.1.1. Overview of sensor applications.

1.2 Statement of the Problem

The problem studied in this work considers the target tracking in a binary sensor network by using particle filtering approach. For this purpose, we explore the feasibility of using particle filtering approach for target tracking by formulating it as a state space model i.e., state and observation equations. The tracking is based on the SMC methods for computing the a posteriori probabilities of unknown state vector. This dissertation presents the following work:

1. Study of Particle filters and its application to target tracking in binary sensor networks in the presence of AWGN noise or Mixture noise.
2. Application of Particle filtering for target tracking in binary sensor networks in the presence of fading channel.
3. Study of Posterior Cramer-Rao Bounds for discrete time nonlinear filtering and its application to target tracking scenario in binary sensor networks.

1.3 Organization of the Report

This report is organized in five chapters:

In *chapter 1,* the overview of particle filters and its tracking applications followed by the overview of wireless sensor networks and its applications are presented. Finally the statement of problem of the dissertation work is summarized.

In *chapter 2,* an overview of recursive Bayesian approach to the estimation of the system state using noisy measurements made on the system is described first. The optimal filtering technique namely Kalman filter, for the linear system and Gaussian noise is summarized. A detailed derivation of sequential importance sampling (SIS), which is the basis for the particle filtering technique is presented. The degeneracy phenomenon, resampling and choice of sampling density in particle filter are emphasized. Various versions of particle filters are also presented. Comparisons of various versions of particle filters are done based on the simulation results.

In *chapter 3,* Binary sensor network model is described by considering it as a special case of wireless sensor networks for tracking the target within a sensor field monitored by a sensor network. The particle filtering approach for tracking the target in binary sensor networks is described. The simulation results are presented by considering both AWGN and mixture noise channels. A detailed derivation of Posterior Cramer-Rao Bounds (PCRB) for discrete time nonlinear filtering is presented. The simulation results of PCRB for channel unaware particle filters are also presented.

In *chapter 4,* the channel aware particle filtering approach for tracking the target in binary sensor networks is described. Simulation results are presented by considering the fading channels between the sensors and the fusion center.

Chapter 5 concludes the report with suggestions for future work.

Chapter 2
SEQUENTIAL MONTE CARLO METHODS FOR BAYESIAN FILTERING

In this chapter, an overview of recursive Bayesian approach to the estimation of the system state using noisy measurements made on the system is described first. The concept of Monte Carlo sampling for solving the intractable integrals is discussed. A detailed derivation of sequential importance sampling (SIS), which is the basis for the particle filtering technique is presented. The degeneracy phenomenon in particle filter and the concept of resampling in particle filter is described next .The choice of sampling density with emphasis on the Gaussian optimal importance function is discussed. Various versions of particle filters like sampling importance resampling (SIR) filter, auxiliary particle filter (APF), regularized particle filter (RPF), Markov Chain Monte Carlo (MCMC) particle filters are also presented. Finally the simulation results for a nonlinear system are presented.

2.1 Recursive Bayesian Estimation

Bayesian theory is a branch of probability theory that helps to model the uncertainty about the world and the outcomes of interest by incorporating prior knowledge and observational evidence. Bayesian analysis, interpreting the probability as a conditional measure, is one of the popular methods in many cases.

In Bayesian reference, all uncertainties (including states, parameters, which are either time-varying or fixed but unknown, priors) are treated as random variables. The inference is performed with in the Bayesian framework given all of available information. The objective of Bayesian inference is to use the priors and causal knowledge, quantitatively and qualitatively, to infer the conditional probability, given finite observations. There are usually three levels of the probabilistic reasoning in Bayesian analysis. Starting with model selection given the data and assumed priors; estimate the parameters to fit the data given the model and priors; and update the hyper

parameters of the prior. There are three types of intractable problems inherently related to the evaluation of *a posteriori* density $p(\mathbf{x}/\mathbf{y})$ [1].

- **Normalization:** Given the prior $p(\mathbf{x})$ and likelihood $p(\mathbf{y}/\mathbf{x})$, the posterior $p(\mathbf{x}/\mathbf{y})$ is obtained by the product of prior and likelihood divided by a normalizing factor .The expression for the posterior $p(\mathbf{x}/\mathbf{y})$ is given by

$$p(\mathbf{x}/\mathbf{y}) = \frac{p(\mathbf{y}/\mathbf{x})p(\mathbf{x})}{\int_X p(\mathbf{y}/\mathbf{x})p(\mathbf{x})d\mathbf{x}} \tag{2.1}$$

- **Marginalization:** Given the posterior $p(\mathbf{x},\mathbf{z}/\mathbf{y})$, the marginal posterior $p(\mathbf{x}/\mathbf{y})$ is calculated by

$$p(\mathbf{x}/\mathbf{y}) = \int_Z p(\mathbf{x},\mathbf{z}/\mathbf{y})d\mathbf{z} \tag{2.2}$$

- **Expectation:** Given the conditional pdf $p(\mathbf{x}/\mathbf{y})$, the expectation of the function $f(\mathbf{x})$ can be calculated as

$$E_{p(\mathbf{x}/\mathbf{y})}[f(\mathbf{x})] = \int_X f(\mathbf{x})p(\mathbf{x}/\mathbf{y})d\mathbf{x} \tag{2.3}$$

where **x**, **y** and **z** are random variables in equations (2.1), (2.2) and (2.3).

For many problems in communications and signal processing, an estimate is required every time a measurement is received. In this case, a recursive filter is a convenient solution. A recursive filtering approach means that received data is processed sequentially rather than as a batch so that it is not necessary to store the complete data set or to reprocess existing data if a new measurement becomes available. State space model [2], [5], which is used in such situations is essentially a notational convenience used for estimation and control problems. State space model comprises of two models namely system model and measurement model. The system model describes about the evolution of the state with time and measurement model relates the noisy measurements to the state. The generalized form of state space model is given by [2].

System equation

$$\mathbf{x}_k = \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \tag{2.4}$$

where $\mathbf{f}_k : \Re^{n_x} \times \Re^{n_v} \rightarrow \Re^{n_x}$ is a possibly nonlinear evolution function.

n_x and n_v are the dimensions of the state and process noise respectively.

$\mathbf{x}_k \in \Re^{n_x}$ is state vector.

$\mathbf{v}_{k-1} \in \Re^{n_v}$ is an i.i.d process noise.

Measurement equation:

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k, \mathbf{n}_k) \tag{2.5}$$

where $\mathbf{h}_k : \Re^{n_x} \times \Re^{n_n} \rightarrow \Re^{n_z}$ is a possibly nonlinear measurement function.

n_x and n_n are the dimensions of the state and measurement noise respectively .

$\mathbf{n}_k \in \Re^{n_n}$ is an i.i.d measurement noise.

From the Bayesian perspective of dynamic state estimation, it is required to construct a posterior probability density function (pdf) of the state $p(\mathbf{x}_k / \mathbf{z}_{1:k})$ based on all the available observations $\mathbf{z}_{1:k}$ up to time k. It is assumed that the initial pdf $p(\mathbf{x}_0 / \mathbf{z}_0) \equiv p(\mathbf{x}_0)$ of the state vector, which is also known as the prior, is available. Then the pdf $p(\mathbf{x}_k / \mathbf{z}_{1:k})$ may be obtained recursively in two stages: *prediction* and *update*. Two assumptions are used to derive the recursive Bayesian filter [1].

(i) The states follow a first-order Markov process i.e.,

$$p(\mathbf{x}_k / \mathbf{x}_{0:k-1}) = p(\mathbf{x}_k / \mathbf{x}_{k-1}) \tag{2.6}$$

(ii) The observations are independent of the given states.

2.1.1 Prediction Stage

The prediction stage uses the system model to predict the state pdf forward from one measurement time to next. Since the state is usually subject to unknown disturbances (modeled as random noise), prediction generally translates, deforms, and spreads the state pdf. Specifically, given the pdf $p(\mathbf{x}_{k-1} / \mathbf{z}_{1:k-1})$ which is already available at time k-1, this stage involves the calculation of the pdf $p(\mathbf{x}_k / \mathbf{z}_{1:k-1})$.

$$p(\mathbf{x}_k / \mathbf{z}_{1:k-1}) = \left(\int_{-\infty}^{\infty} p(\mathbf{x}_k / \mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}, \mathbf{z}_{1:k-1}) d\mathbf{x}_{k-1} \right) \tag{2.7}$$

The equation (2.7) is known as the Chapman-Kolmogorov (CK) equation [3].

2.1.2 Update Stage

The update stage involves modification of the prediction pdf based on the latest measurement available at that time. Specifically, given the measurement $p(\mathbf{z}_k)$ available at time k then it is used to update the prior via Baye's rule [3].

$$p(\mathbf{x}_k / \mathbf{z}_{1:k}) = p(\mathbf{x}_k, \mathbf{z}_{1:k}) / p(\mathbf{z}_{1:k})$$

$$= \left[p(\mathbf{z}_k / \mathbf{x}_k, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k, \mathbf{z}_{1:k-1}) \right] / \left[p(\mathbf{z}_k / \mathbf{z}_{1:k-1}) p(\mathbf{z}_{1:k-1}) \right]$$

$$= p(\mathbf{z}_k / \mathbf{x}_k) p(\mathbf{x}_k / \mathbf{z}_{1:k-1}) / p(\mathbf{z}_k / \mathbf{z}_{1:k-1}) \tag{2.8}$$

where the normalizing constant is given by

$$p(\mathbf{z}_k / \mathbf{z}_{1:k-1}) = \left(\int_{-\infty}^{\infty} p(\mathbf{z}_k / \mathbf{x}_k) p(\mathbf{x}_k / \mathbf{z}_{1:k-1}) d\mathbf{x}_k \right) \tag{2.9}$$

The normalizing constant depends on the likelihood function $p(\mathbf{z}_k / \mathbf{x}_k)$ defined by the measurement model and the known statistics of observation noise $\mathbf{n}_k$. In the update stage, the measurement $\mathbf{z}_k$ is used to modify the prior density to obtain the required posterior density of the current state.

The recursive relations described above are easily solved for linear/Gaussian systems. In case the system is nonlinear/non-Gaussian in nature, the integrals are not tractable. In such cases, the approximate solution is provided by several non-linear filters.

2.2 Monte Carlo Sampling

Monte Carlo [MC] methods [1] are commonly used for approximation of intractable integrals and rely on the ability to draw a random sample from the required probability distribution Monte Carlo methods use statistical sampling and estimation techniques to evaluate the solutions to mathematical problems. Monte Carlo techniques have attracted lot of attention and have been developed in many areas. Monte Carlo methods have three categories: (i) Monte Carlo sampling, which is devoted to developing efficient sampling technique for estimation; (ii) Monte Carlo calculation, which is aimed to design various random or pseudo-random number generators; and (iii) Monte Carlo optimization, which is devoted to applying the Monte Carlo idea to optimize some non differentiable functions. In the following only Monte Carlo sampling [4] is discussed.

Consider the multidimensional integral $I = \int g(\mathbf{x})d\mathbf{x}$, where $\mathbf{x} \in R^{n_x}$ .Monte Carlo methods for numerical integration factorize $g(\mathbf{x}) = \pi(\mathbf{x})f(\mathbf{x})$ in such a way that $\pi(\mathbf{x})$ is interpreted as a probability density satisfying $\pi(\mathbf{x}) \geq 0$ and $\int \pi(\mathbf{x})d\mathbf{x} = 1$, $f(\mathbf{x})$ is an integrable function in a measurable space. The assumption is that it is possible to draw N samples $\{\mathbf{x}^i; i = 1,....N\}$ distributed according to $\pi(\mathbf{x})$. Then the pdf $\pi(\mathbf{x})$ can be approximated as [4]

$$\pi(\mathbf{x}) = \frac{1}{N}\delta(\mathbf{x} - \mathbf{x}^i) \tag{2.10}$$

The Monte Carlo estimate of the integral

$$I = \int f(\mathbf{x})\pi(\mathbf{x})d\mathbf{x} \tag{2.11}$$

is the sample mean

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i) \tag{2.12}$$

By taking large number of samples, the estimate converges to its true value. The variance of the estimate is inversely proportional to number of samples. There are several issues which are of concern in Monte Carlo sampling [1]

- **Consistency:** An estimator is consistent if the estimator converges to the true value almost surely as the number of observations approaches infinity.
- **Unbiasedness:** An estimator is unbiased if its expected value is equal to the true value.
- **Efficiency:** An estimator is efficient if it produces the smallest error covariance matrix among all unbiased estimators, it is also regarded optimally using the information in the measurements. A well-known efficiency criterion is the Cramer-Rao bound.
- **Robustness:** An estimator is robust if it is insensitive to the gross measurement errors and the uncertainties of the model.
- **Minimal variance:** Variance reduction is the central issue of various Monte Carlo approximation methods, most improvement techniques are variance reduction oriented.

In Bayesian estimation context, density $\pi(\mathbf{x})$ is the posterior density. It is not possible to sample effectively from the posterior distribution, being multivariate, nonstandard, and only known up to proportionality constant. A possible solution is to apply the importance sampling method.

2.2.1 Importance Sampling

Ideally the samples are generated from the density $\pi(\mathbf{x})$ and the integral I is evaluated by using $I_N = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)$. If the samples are easily generated from a density $q(\mathbf{x})$, which is similar to $\pi(\mathbf{x})$, then a correct weighting of the sample set still makes the

Monte Carlo estimation possible. The pdf $q(\mathbf{x})$ is referred to as importance or proposal density [3], [5]. Its similarity to $\pi(\mathbf{x})$ is interpreted by

$$\pi(\mathbf{x}) > 0 \Rightarrow \; q(\mathbf{x}) > 0 \qquad \forall \mathbf{x} \in R^{n_x} \tag{2.13}$$

This means that $q(\mathbf{x})$ and $\pi(\mathbf{x})$ has same support. The equation (2.13) is necessary for the importance sampling theory to hold and, if valid, the integral I is written as

$$I = \int f(\mathbf{x})\pi(\mathbf{x})d\mathbf{x} = \int f(\mathbf{x})\frac{\pi(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})d\mathbf{x} \tag{2.14}$$

provided that $\frac{\pi(\mathbf{x})}{q(\mathbf{x})}$ is upper bounded. A Monte Carlo estimate of I is computed by generating $N>>1$ independent samples $\{\mathbf{x}^i; i = 1,....N\}$ distributed according to $q(\mathbf{x})$ and forming the weighted sum:

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)\hat{w}(\mathbf{x}^i) \tag{2.15}$$

where,

$$\hat{w}(\mathbf{x}^i) = \frac{\pi(\mathbf{x}^i)}{q(\mathbf{x}^i)} \qquad i = 1,....., N \tag{2.16}$$

are the *importance weights*[3]. If normalizing factor of the desired density $\pi(\mathbf{x})$ is unknown, then normalization of the importance weights is carried out. Then the estimate of the integral I_N is given by

$$I_N = \frac{\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)\hat{w}(\mathbf{x}^i)}{\frac{1}{N}\sum_{j=1}^{N} \hat{w}(\mathbf{x}^j)} = \sum_{i=1}^{N} f(\mathbf{x}^i)w(\mathbf{x}^i) \tag{2.17}$$

where, the normalized importance weights are given by

$$w(\mathbf{x}^i) = \frac{\hat{w}(\mathbf{x}^i)}{\frac{1}{N}\sum_{j=1}^{N}\hat{w}(\mathbf{x}^j)} \qquad i = 1,\ldots\ldots,N \tag{2.18}$$

This technique is used in the Bayesian framework, where $\pi(\mathbf{x})$ is the posterior density.

2.3 Particle Filtering

Sequential Monte Carlo methods have found limited use in the past, except for the last decade, primarily due to their very high computational complexity and the lack of adequate computing resources. The fast advances of computers in the recent years and outstanding potential of particle filters have made them a very active area of research recently. Particle filter [1], [3], [4], [5] is a sequential Monte Carlo methodology based on the recursive computation of probability distributions. The basic idea of particle filter is to use a number of independent random variables called particles, sampled directly from the state space, to represent the posterior probability, and update the posterior by involving the new observations; the "particle system" is properly located, weighted, and propagated recursively according to the Bayesian rule. Particle filters are sequential Monte Carlo methods which can be applied to any state space model and which generalizes the Kaman filtering methods. The advantage of particle filtering over other methods is in that the exploited approximation does not involve linearization's around current estimates but rather approximations in the representation of the desired distributions by discrete random measures. Particle filter is best suited for nonlinear state-space models and non-Gaussian noises. Particle filters have found application in many areas such as channel equalization, estimation and coding, wireless channel tracking, artificial intelligence, speech enhancement, speech recognition and machine learning etc.

2.3.1 Sequential Importance Sampling (SIS)

In order to make Bayesian importance sampling more practical, it will be convenient to calculate the particle weights recursively. The sequential importance sampling (SIS) [2], [3], [4] algorithm is a Monte Carlo (MC) method that forms the basis for most sequential MC filters developed over the past decades. It is a technique for

implementing a recursive Bayesian filter by MC simulations. The key idea is to represent the required posterior density function by a set of random samples with associated weights and to compute estimates based on these samples and weights. As the number of samples becomes very large, this MC characterization becomes an equivalent representation to the usual functional description of the posterior pdf, and the SIS filter approaches the optimal Bayesian estimate.

Let $\{\mathbf{x}^i_{0:k}, w^i_k\}_{i=1}^{N_s}$ denote a random measure that characterizes the posterior pdf $p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$. Where, $\{\mathbf{x}^i_{0:k}, i = 0,....,N_s\}$ is a set of sample points with associated weights $\{w^i_k, i = 1,....,N_s\}$ and $\mathbf{x}_{0:k} = \{\mathbf{x}_j, j = 0,....,k\}$ is the set of all states up to time k. The weights are normalized such that $\sum_i w^i_k = 1$. By SIS algorithm, the set $\{\mathbf{x}^i_{0:k}, w^i_k\}_{i=1}^{N_s}$ is recursively computed from the set $\{\mathbf{x}^i_{0:k-1}, w^i_{k-1}\}_{i=1}^{N_s}$ when a new measurement $\mathbf{z}_k$ is available at time k. Specifically, suppose at time $k-1$ the posterior pdf $p(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1})$ is approximated by a random measure $\{\mathbf{x}^i_{0:k-1}, w^i_{k-1}\}_{i=1}^{N_s}$, then SIS algorithm builds a random measure by appending newly generated particles $\mathbf{x}^i_k$ to the $\mathbf{x}^i_{0:k-1}$ and updating the weights w^i_k to form $\{\mathbf{x}^i_{0:k}, w^i_k\}_{i=1}^{N_s}$ that properly represent the posterior pdf $p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$ Then, the posterior density at time k is approximated as [2]

$$p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta\left(\mathbf{x}_{0:k} - \mathbf{x}^i_{0:k}\right) \tag{2.19}$$

The above equation represents the discrete weighted approximation to the true posterior, $p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$. The weights can be chosen using the principle of importance sampling. If the samples $\mathbf{x}^i_{0:k}$ were drawn from an importance density $q(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$, the weights are given by

$$w^i_k \propto \frac{p(\mathbf{x}^i_{0:k} / \mathbf{z}_{1:k})}{q(\mathbf{x}^i_{0:k} / \mathbf{z}_{1:k})} \tag{2.20}$$

At each iteration by using the approximated $p(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1})$, and with a new set of samples; the pdf $p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$ is calculated. The importance density $q(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$ is factorized as

$$q(\mathbf{x}_{0:k} / \mathbf{z}_{1:k}) = \frac{q(\mathbf{x}_{0:k}, \mathbf{z}_{1:k})}{q(\mathbf{z}_{1:k})}$$

$$= \frac{q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) q(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})}{q(\mathbf{z}_{1:k})}$$

$$= \frac{q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) q(\mathbf{z}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) q(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}{q(\mathbf{z}_k / \mathbf{z}_{1:k-1}) q(\mathbf{z}_{1:k-1})}$$

$$= \frac{q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) q(\mathbf{z}_k / \mathbf{z}_{1:k-1}) q(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}{q(\mathbf{z}_k / \mathbf{z}_{1:k-1}) q(\mathbf{z}_{1:k-1})}$$

$$= q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) q(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1})$$

$$q(\mathbf{x}_{0:k} / \mathbf{z}_{1:k}) = q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) q(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1}) \tag{2.21}$$

By the equation (2.21), the samples $\mathbf{x}^i_{0:k} \sim q(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$ are obtained by augmenting each of the existing samples $\mathbf{x}^i_{0:k-1} \sim q(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1})$ with the new state $\mathbf{x}^i_k \sim q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})$. The pdf $p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k})$ is expressed as

$$p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k}) = \frac{p(\mathbf{x}_{0:k}, \mathbf{z}_{1:k})}{p(\mathbf{z}_{1:k})}$$

$$= \frac{p(\mathbf{z}_k / \mathbf{x}_{0:k}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_{0:k}, \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k / \mathbf{z}_{1:k-1}) p(\mathbf{z}_{1:k-1})}$$

$$= \frac{p(\mathbf{z}_k / \mathbf{x}_{0:k}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k / \mathbf{z}_{1:k-1}) p(\mathbf{z}_{1:k-1})}$$

$$= \frac{p(\mathbf{z}_k / \mathbf{x}_{0:k}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k / \mathbf{z}_{1:k-1})}$$

$$= \frac{p(\mathbf{z}_k / \mathbf{x}_k)p(\mathbf{x}_k / \mathbf{x}_{k-1})p(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k / \mathbf{z}_{1:k-1})}$$

$$p(\mathbf{x}_{0:k} / \mathbf{z}_{1:k}) \propto p(\mathbf{z}_k / \mathbf{x}_k)p(\mathbf{x}_k / \mathbf{x}_{k-1})p(\mathbf{x}_{0:k-1} / \mathbf{z}_{1:k-1}) \tag{2.22}$$

where $p(\mathbf{z}_k / \mathbf{z}_{1:k-1})$ is a normalized constant. Now substituting the equations (2.12) and (2.22) in equation (2.20), then

$$w_k^i \propto \frac{p(\mathbf{z}_k / \mathbf{x}^i{}_k)p(\mathbf{x}^i{}_k / \mathbf{x}^i{}_{k-1})p(\mathbf{x}^i{}_{0:k-1} / \mathbf{z}_{1:k-1})}{q(\mathbf{x}^i{}_k / \mathbf{x}^i{}_{0:k-1}, \mathbf{z}_{1:k})q(\mathbf{x}^i{}_{0:k-1} / \mathbf{z}_{1:k-1})}$$

$$w_k^i = w_{k-1}^i \frac{p(\mathbf{z}_k / \mathbf{x}^i{}_k)p(\mathbf{x}^i{}_k / \mathbf{x}^i{}_{k-1})}{q(\mathbf{x}^i{}_k / \mathbf{x}^i{}_{0:k-1}, \mathbf{z}_{1:k})} \tag{2.23}$$

Furthermore, if $q(\mathbf{x}_k / \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) = q(\mathbf{x}_k / \mathbf{x}_{k-1}, \mathbf{z}_k)$, then the importance density becomes only dependent on $\mathbf{x}_{k-1}$ and $\mathbf{z}_k$ This is particularly useful in the common case when only a filtered estimate of $p(\mathbf{x}_k / \mathbf{z}_{1:k})$ is required for each time step. In such situations, only $\mathbf{x}_k^i$ need to be stored and the path $\mathbf{x}_{0:k-1}^i$, the history of observations $\mathbf{z}_{1:k-1}$ can be discarded. Then the modified weight is given by

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k / \mathbf{x}_k^i)p(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i, \mathbf{z}_k)} \tag{2.24}$$

The posterior filtered density $p(\mathbf{x}_k / \mathbf{z}_{1:k})$ is given by

$$p(\mathbf{x}_k / \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta\left(\mathbf{x}_k - \mathbf{x}_k^i\right) \tag{2.25}$$

Thus the SIS algorithm consists of recursive propagation of weights and samples as each measurement is received sequentially. A pseudo-code description of the SIS algorithm is given by algorithm 2.1 [2], [3].

***Algorithm 2.1**: SIS Particle filter*

$$\left[\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = SIS\left[\left\{\mathbf{x}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{z}_k\right]$$

- *FOR* $i = 1 : N_s$

  - *Draw* $\mathbf{x}_k^i \sim q(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i, \mathbf{z}_k)$
  - *Assign each particle with the importance weight up to a normalizing constant according to*

$$\tilde{w}_k^i = w_{k-1}^i \frac{p(\mathbf{z}_k / \mathbf{x}_k^i) p(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i, \mathbf{z}_k)}$$

- *END FOR*
- *Calculate the total weight:* $t = SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i = 1 : N_s$

  - *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$

- *END FOR*

2.3.2 Degeneracy Phenomenon and Resampling in Particle Filters

In particle filters, the posterior probability is represented by a set of randomly chosen weighted samples drawn from an importance density. However a common problem with the sequential importance sampling is that after a few iterations, most particles will have negligible weight. It means that the weight is concentrated on certain particles only. This problem is called *degeneracy* problem [2], [3].The variance of the importance weights increases over time, and thus it is impossible to avoid the degeneracy problem. Effectively a large computational effect is devoted to updating particles whose contribution to approximate the posterior pdf is almost zero. A suitable measure of degeneracy of the algorithm is the *effective sample size* (N_{eff}) given by

$$N_{eff} = \frac{N_s}{1 + Var(w_k^{*i})} \tag{2.26}$$

Where, $w_k^{*i} = p(\mathbf{x}_k^i / \mathbf{z}_{1:k}) / q(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i, \mathbf{z}_k)$ .Thus the effective sample size cannot be evaluated exactly, an estimate is calculated instead which is given by

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s} (w_k^i)^2} \tag{2.27}$$

The small N_{eff}, the severe will be the degeneracy. There are three basic measures to mitigate the degeneracy problem in particle filters, (1) by increasing the number of samples N_s, (2) resampling. (3) by good choice of importance density. The simplest method to mitigate the degeneracy effect is to use a very large N_s. however it will increase the computational load on the system, which is often impractical.

Resampling

Effects of degeneracy in particle filter are reduced by using resampling [3], [6], [7] where the particles having small weights are eliminated and the particles with large weights are replicated.

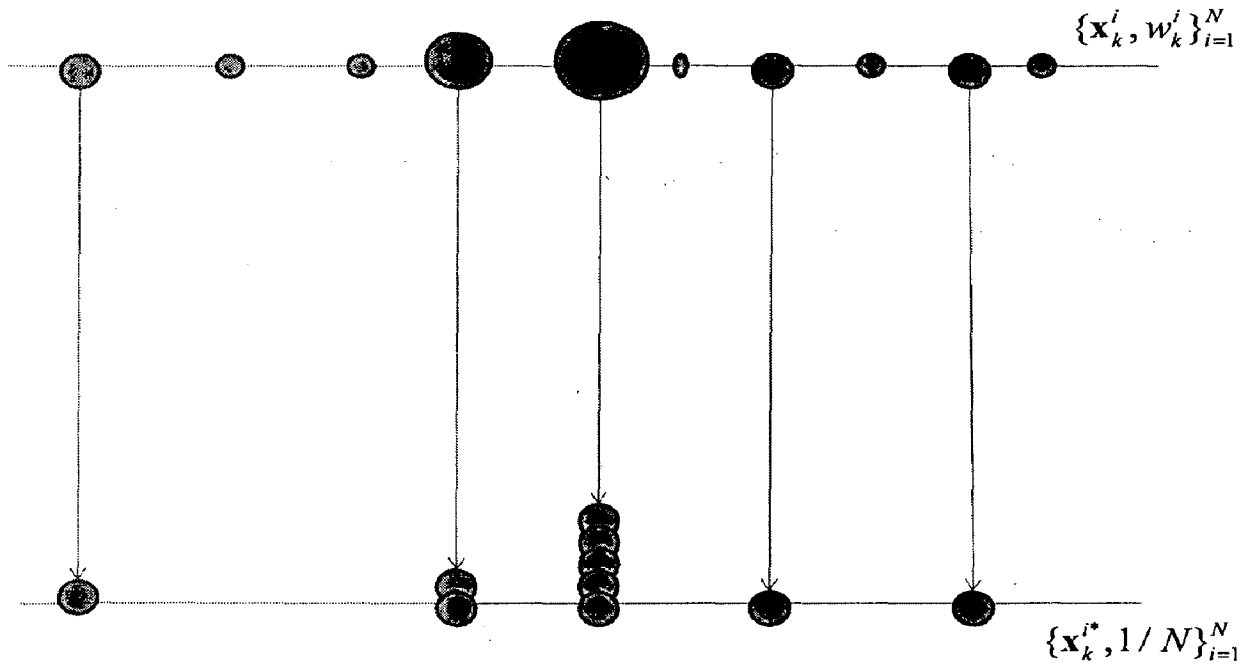


Figure 2.1 Particle resampling

The resampling stage is depicted in the above Fig.2.1. At every step the effective particle size is calculated. The calculated effective size is compared with the predefined threshold, based on that the resampling step will be carried out. The resampling stage involves drawing of N samples from the aposterior pdf with replacement. All the particles after resampling have the same weight $1/N$. By this, the particles having large weight are repeated and particles having less weight are eliminated. Thus the samples are concentrated in the region of interest. From Fig.2.1, it may be seen that the diameters of the circles are proportional to the weights of the particles and after resampling all the particles are having the same weight.

Resampling involves a mapping of random measure $\{\mathbf{x}_k^i, w_k^i\}$ into a random measure $\{\mathbf{x}_k^{i*}, 1/N\}$ with uniform weights. The set of random samples $\{\mathbf{x}_k^{i*}\}_{i=1}^N$ is generated by resampling (with replacement) N times from an approximate discrete representation of $p(\mathbf{x}_k/\mathbf{z}_{1:k})$ with the probability $p\{\mathbf{x}_k^{i*} = \mathbf{x}_k^j\} = w_k^j$. The resulting sample is an i.i.d sample from the a posterior density $p(\mathbf{x}_k/\mathbf{z}_{1:k})$, and hence new weights are uniform.. The selection of new samples is schematically shown in the Fig.2.2 [3].

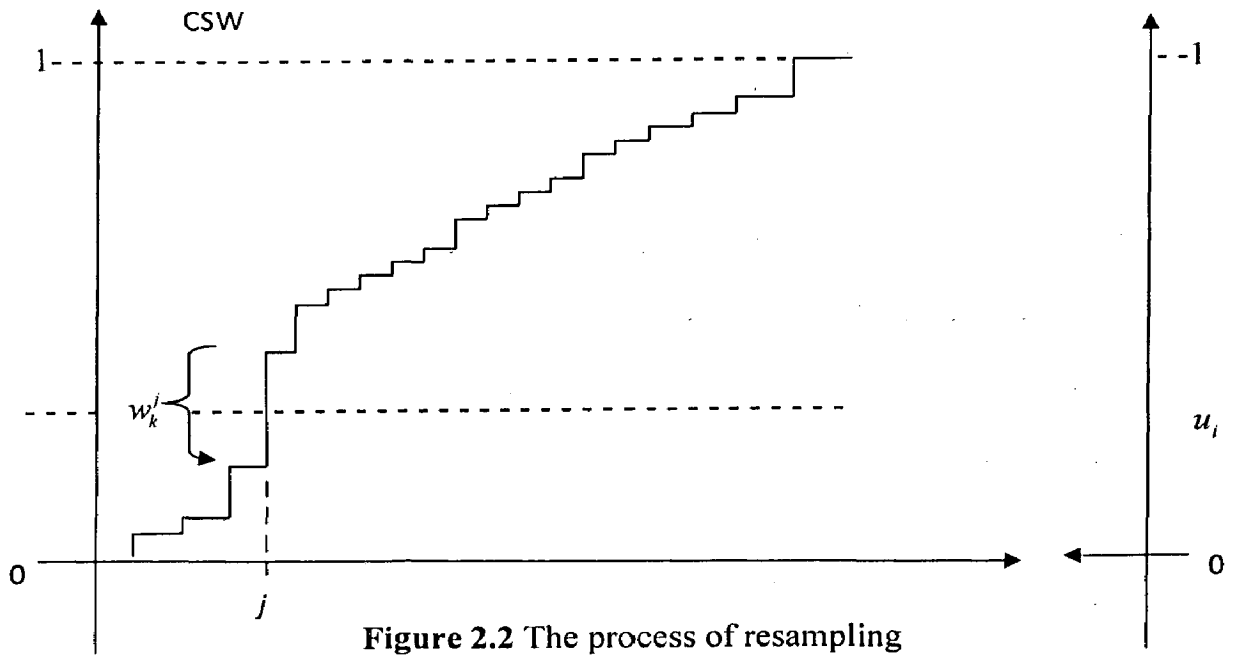


Figure 2.2 The process of resampling

In Fig.2.2, the acronym CSW stands for the cumulative sum of weights of the random measure $\{\mathbf{x}_k^j, w_k^j\}$, and random variable $u_i, i = 1,....,N$ is uniformly distributed in the interval [0,1]. From Fig.2.2, the main idea in the process of resampling is to select the new particles by comparing an ordered set of uniformly distributed random numbers $u_i, i = 1,....,N$ lies in the interval [0,1] with the cumulative sum of the normalized weights. It may be seen that from Fig.2.2, uniform random variable u_i maps into index j and the corresponding particle $\mathbf{x}_k^j$ has a good chance of being selected and multiplied because of its high value of w_k^j.This technique is mainly used in systematic resampling [3], [6], [7] and residual resampling [6], [7], which are given by algorithm 2.2 and 2.3.

Algorithm 2.2: Systematic resampling **[6], [7]**

- *Generate N uniform random numbers*

$$\tilde{u} \sim U[0,1)$$

- *Obtain the N ordered random numbers* u_k

$$u_k = \frac{(k-1)+\tilde{u}}{N}$$

- *Allocate the* n_i *copies of the particle* x_i *to the new distribution*

$$n_i = \text{the number of } u_k \in \left[\sum_{s=1}^{i-1} w_s, \sum_{s=1}^{i} w_s\right)$$

Algorithm 2.3: Residual resampling **[6], [7]**

- *Allocate* $n_i' = \lfloor Nw_i \rfloor$ *copies of the particle* x_i *to the new distribution*

- *Additionally, resample* $m = N - \sum n_i'$ *particles from* $\{x_i\}$ *by making* n_i'' *copies of particle* x_i *where the probability for selecting* x_i *is proportional to* $w_i' = Nw_i - n_i'$ *using systematic resampling.*

Now, the generic particle filter algorithm is given by algorithm 2.4.

Algorithm 2.4: Generic particle filter [2, 3]

$$\left[\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = PF\left[\left\{\mathbf{x}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{z}_k\right]$$

- *FOR* $i = 1:N_s$
  - *Draw* $\mathbf{x}_k^i \sim q(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k)$
  - *Assign each particle with the importance weight up to a normalizing constant according to*

$$\tilde{w}_k^i = w_{k-1}^i \frac{p(\mathbf{z}_k / \mathbf{x}_k^i) p(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i, \mathbf{z}_k)}$$

- *END FOR*
- *Calculate the total weight:* $t = SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i = 1:N_s$
  - *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$
- *END FOR*
- *Calculate effective sample size* $\hat{N}_{eff}$ *by*

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s} (w_k^i)^2}$$

- *IF* $\hat{N}_{eff} < N_T$

➢ *Resample using systematic resampling or residual resampling.*

- *END IF*

The general particle filtering algorithm 2.4 may be represented by Fig.2.3 [1].

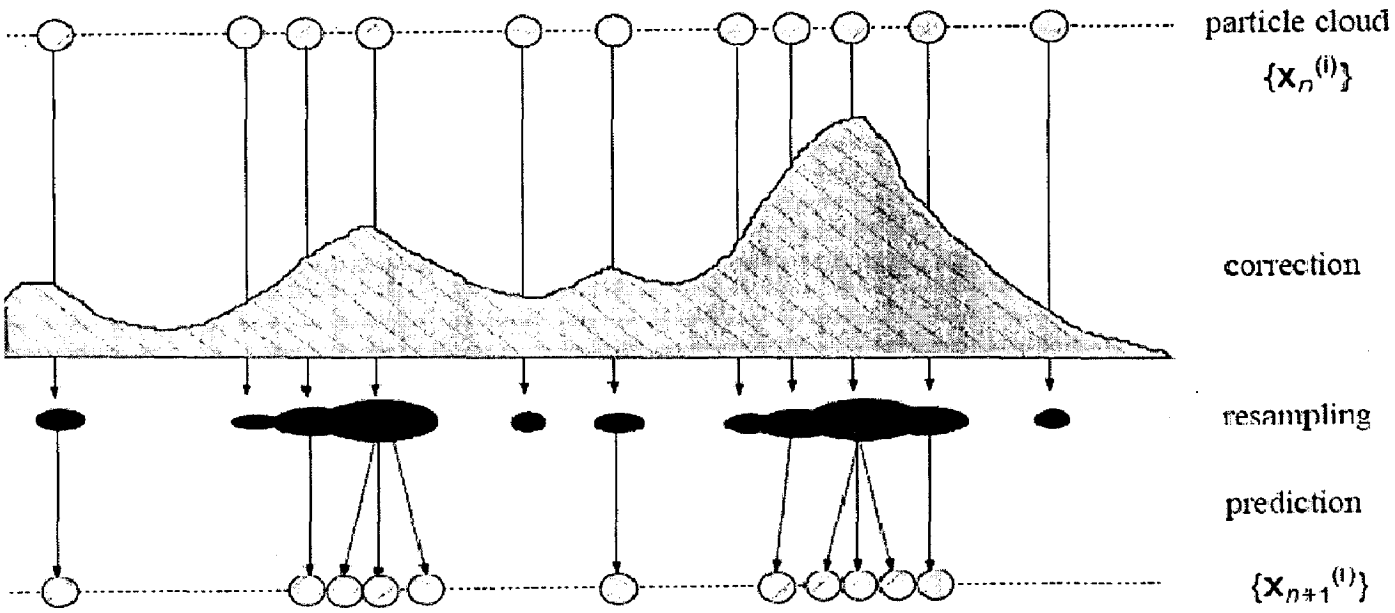


Figure 2.3 An illustration of generic Particle filter with importance sampling and resampling

From Fig.2.3, it is seen that the particles are modified by the importance density function. The higher the probability, the denser the particles are concentrated. The circle diameters are proportional to the weights of the particles. The effective size of all the particles is calculated. If the effective size is less than the predefined threshold, then the resampling step is carried out (i.e., the larger particles are repeated and the smaller particles are neglected). After resampling, the weights of all particles are same. Now these particles constitute the new set of particles. Then the whole procedure (Generic particle filter algorithm 2.4) is repeated with these new set of samples.

Although the resampling step reduces the effects of the degeneracy problem, it introduces other problems. First, it limits the opportunity to parallelize the implementation since all the particles must be combined. Second, the particles that have high weights are statistically selected many times, this lead to a loss of diversity

among the particles as the resultant sample will contain many repeated points. This problem is known as *sample impoverishment* [1], [3]. There are techniques namely *Markov chain Monte Carlo (MCMC)* [4], *regularization* [3] method to reduce the effect of sample impoverishment.

The sequential importance sampling algorithm is common for all types of particle filters. There are other versions of particle filters [2], [3], namely (1) sampling importance resampling (SIR) filter; (2) auxiliary sampling importance resampling (ASIR) filter; (3)regularized particle filter (RPF).

2.3.3 Choice of Importance Density

The choice of the sampling density affects the quality of the state estimate significantly [2], [3]. However there are number of choices for the sampling density. The sampling density must fulfill a criterion to ensure convergence of the estimates as number of samples N_s becomes large. Further, the shape of the sampling density must be as close to the true filtering pdf as possible and it should guarantee a minimum variance. The sampling density should also be as simple with respect to the weights evaluation as possible.

Optimal Sampling Density

If sampling density is chosen to minimize the variance of weights [2,3] so that effective sample size is maximized, then it is said to be optimal sampling density. This sampling density will assume the form

$$q(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k)_{opt} = p(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k)$$

$$= \frac{p(\mathbf{z}_k, \mathbf{x}_k, \mathbf{x}_{k-1}^i)}{p(\mathbf{z}_k, \mathbf{x}_{k-1}^i)}$$

$$= \frac{p(\mathbf{z}_k / \mathbf{x}_k, \mathbf{x}_{k-1}^i) p(\mathbf{x}_k, \mathbf{x}_{k-1}^i)}{p(\mathbf{z}_k / \mathbf{x}_{k-1}^i) p(\mathbf{x}_{k-1}^i)}$$

$$q(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k)_{opt} = \frac{p(\mathbf{z}_k/\mathbf{x}_k, \mathbf{x}_{k-1}^i)p(\mathbf{x}_k/\mathbf{x}_{k-1}^i)}{p(\mathbf{z}_k/\mathbf{x}_{k-1}^i)} \tag{2.28}$$

Substituting equation (2.28) in equation (2.24) we get

$$w_k^i \propto w_{k-1}^i p(\mathbf{z}_k / \mathbf{x}_{k-1}^i)$$

$$= w_{k-1}^i \int p(\mathbf{z}_k / \mathbf{x}_k^{'}) p(\mathbf{x}_k^{'} / \mathbf{x}_{k-1}^{'}) d\mathbf{x}_k^{'} \tag{2.29}$$

Interestingly, the weights do not depend on the current value of the state $\mathbf{x}_k^i$ .The above chosen optimal density has two limitations. It requires sampling from the pdf $p(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k)$ and the evolution of integral expression (2.29). Both of them cannot be done easily. When $\mathbf{x}_k$ belongs to a finite set, then the integral expression (2.29) become a sum, and sampling from the optimal importance density is possible.

Prior Sampling Density

This sampling density is frequently used due to its simplicity and easy weight computation. Here the current estimate $\mathbf{z}_k$ is ignored during drawing of samples and thus low quality estimates will be obtained. The prior sampling density takes the form [2], [3] as

$$q(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k) = p(\mathbf{x}_k / \mathbf{x}_{k-1}^i) \tag{2.30}$$

By substituting the equation (2.30) in the equation (2.21) we get

$$w_k^i \propto w_{k-1}^i p(\mathbf{z}_k / \mathbf{x}_k^i) \tag{2.31}$$

The equation (2.29) states that it is possible to calculate the importance weights before the particles are propagated to time *k*. The equation (2.31) states that this is not possible with the prior sampling density.

If the transitional prior $p(\mathbf{x}_k/\mathbf{x}_{k-1})$ is used as the importance density and is a much broader distribution than the likelihood, $p(\mathbf{z}_k/\mathbf{x}_k)$, then only a few particles will be assigned a high weight. Consequently, the particles will degenerate rapidly and the filter does not work. The particles should be in the right place (in the regions of high likelihood) by incorporating the current observation, then only efficient estimate is obtained through the particle filter algorithm.

2.4 Sampling Importance Resampling (SIR) Filter

The SIR algorithm can be easily obtained from SIS algorithm by considering the following [2].

- The importance density is chosen to be prior density

$$q(\mathbf{x}_k / \mathbf{x}_{k-1}^i, \mathbf{z}_k) = p(\mathbf{x}_k / \mathbf{x}_{k-1}^i)$$

- The resampling step is carried out at every time index.

By the choice of importance density as the prior density, the weights are given by

$$w_k^i \propto w_{k-1}^i p(\mathbf{z}_k / \mathbf{x}_k^i)$$

However, considering the fact that the resampling step is carried out at every time index, the weight update is given by

$$w_k^i \propto p(\mathbf{z}_k / \mathbf{x}_k^i) \tag{2.32}$$

The weights should be normalized before resampling step is carried out. The advantage of SIR over SIS is easy weight computation. The SIR filter algorithm 2.5 is presented in the following [2], [3].

Algorithm 2.5: SIR Particle filter

$$\left[\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = SIR\left[\left\{\mathbf{x}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{z}_k\right]$$

- *FOR* $i=1:N_s$

  - *Draw* $\mathbf{x}_k^i \sim p(\mathbf{x}_k / \mathbf{x}_{k-1}^i)$
  - *Assign each particle with the importance weight up to a normalizing constant according to*

$$\tilde{w}_k^i = p(\mathbf{z}_k / \mathbf{x}_k^i)$$

- *END FOR*
- *Calculate the total weight:* $t=SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i=1:N_s$

  - *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$

- *END FOR*
- *Resample using systematic resampling or residual resampling.*

2.5 Auxiliary Sampling Importance Resampling Filter

The ASIR filter [2], [3], [19] was introduced by Pitt and Shephard as a variant of the standard SIR filter. This filter can be derived from the SIS framework by introducing an importance density $q(\mathbf{x}_k, i/\mathbf{z}_{1:k})$, which samples the pair $\left\{\mathbf{x}_k^j, i^j\right\}_{j=1}^{M_s}$, where i^j refers to the index of the particle at k-1.

By applying Bayes' rule, the pdf $p(\mathbf{x}_k, i/\mathbf{z}_{1:k})$ can be expressed as follows:

$$p(\mathbf{x}_k, i/\mathbf{z}_{1:k}) \propto p(\mathbf{z}_k/\mathbf{x}_k)\, p(\mathbf{x}_k, i/\mathbf{z}_{1:k-1})$$

$$= \frac{p(\mathbf{z}_k/\mathbf{x}_k)\, p(\mathbf{x}_k, i, \mathbf{z}_{1:k-1})\, p(i, \mathbf{z}_{1:k-1})}{p(i, \mathbf{z}_{1:k-1})\, p(\mathbf{z}_{1:k-1})}$$

$$= p(\mathbf{z}_k/\mathbf{x}_k)\, p(\mathbf{x}_k/i, \mathbf{z}_{1:k-1})\, p(i/\mathbf{z}_{1:k-1})$$

$$= p(\mathbf{z}_k/\mathbf{x}_k)\, p(\mathbf{x}_k/\mathbf{x}_{k-1}^i)\, w_{k-1}^i \qquad (2.33)$$

The ASIR filter operates by obtaining a sample from the joint density $p(\mathbf{x}_k, i/\mathbf{z}_{1:k})$ and then omitting the indices i in the pair $(\mathbf{x}_k, i)$ to produce a sample $\{\mathbf{x}_k^j\}_{j=1}^{M_s}$ from the marginalized density $p(\mathbf{x}_k/\mathbf{z}_{1:k})$. The importance density used to draw the sample $\{\mathbf{x}_k^j, i^j\}_{j=1}^{M_s}$ is defined to satisfy the proportionality

$$q(\mathbf{x}_k, i/\mathbf{z}_{1:k}) \propto p(\mathbf{z}_k/\mu_k^i)\, p(\mathbf{x}_k/\mathbf{x}_{k-1}^i)\, w_{k-1}^i \tag{2.34}$$

where μ_k^i is some characterization of $\mathbf{x}_k$, given $\mathbf{x}_{k-1}^i$.

By writing

$$q(\mathbf{x}_k, i/\mathbf{z}_{1:k}) = q(\mathbf{x}_k/i, \mathbf{z}_{1:k})\, p(i/\mathbf{z}_{1:k}) \tag{2.35}$$

and defining

$$q(\mathbf{x}_k/i, \mathbf{z}_{1:k}) \triangleq p(\mathbf{x}_k/\mathbf{x}_{k-1}^i) \tag{2.36}$$

we have

$$q(i/\mathbf{z}_{1:k}) \propto p(\mathbf{z}_k/\mu_k^i)\, w_{k-1}^i \tag{2.37}$$

The sample $\{\mathbf{x}_k^j, i^j\}_{j=1}^{M_s}$ is then assigned a weight proportional to the ratio of the right-hand side of (2.33) to (2.34)

$$w_k^j \propto w_{k-1}^{i^j} \frac{p(\mathbf{z}_k/\mathbf{x}_k^j)\, p(\mathbf{x}_k^j/\mathbf{x}_{k-1}^{i^j})}{q(\mathbf{x}_k^j, i^j/\mathbf{z}_{1:k})} = \frac{p(\mathbf{z}_k/\mathbf{x}_k^j)}{p(\mathbf{z}_k/\mu_k^{i^j})} \tag{2.38}$$

Compared with the SIR filter, the advantage of the ASIR filter is that it naturally generates points from the sample at k-1, which, conditioned on the current measurement, are most likely to be close to the true state. If the process noise is large, a single point does not characterize $p(\mathbf{x}_k/\mathbf{x}_{k-1}^i)$ well, and ASIR resamples based on a poor approximation of $p(\mathbf{x}_k/\mathbf{x}_{k-1}^i)$. In such scenarios, the use of ASIR will degrade the performance. The ASIR filter algorithm 2.6 is presented in the following [2], [3].

Algorithm 2.6: ASIR Particle filter

$$\left[\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = ASIR\left[\left\{\mathbf{x}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{z}_k\right]$$

- *FOR* $i=1:N_s$
  - *Calculate* μ_k^i
  - *Assign each particle with the importance weight up to a normalizing constant according to* $\tilde{w}_k^i = p(\mathbf{z}_k / \mu_k^i) w_{k-1}^i$
- *END FOR*
- *Calculate the total weight:* $t=SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i=1:N_s$
  - *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$
- *END FOR*
- *Resample using systematic resampling or residual resampling.*
- *FOR* $i=1:N_s$
  - *Draw* $\mathbf{x}_k^j \sim p(\mathbf{x}_k / \mathbf{x}_{k-1}^{i^j})$
  - *Assign each particle with the importance weight up to a normalizing constant according to* $w_k^j = \dfrac{p\left(\mathbf{z}_k / \mathbf{x}_k^j\right)}{p\left(\mathbf{z}_k / \mu_k^{i^j}\right)}$
- *END FOR*
- *Calculate the total weight:* $t=SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i=1:N_s$
  - *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$
- *END FOR*

2.6 Regularized Particle Filter:

The RPF [2], [3] resamples from a continuous approximation of the posterior density $p(\mathbf{x}_k / \mathbf{z}_{1:k})$ which is given by

$$p(\mathbf{x}_k / \mathbf{z}_k) \approx \sum_{i=1}^{N} w_k^i K_h(\mathbf{x}_k - \mathbf{x}_k^i) \tag{2.39}$$

where

$K_h(\mathbf{x}) = \frac{1}{h^{n_x}} K\left(\frac{\mathbf{x}}{h}\right)$ is the kernel density, h>0 is the kernel bandwidth.

The optimal choice of the kernel is the Epanechnikov kernel, which is given by

$$K_{opt} = \begin{cases} \frac{n_x + 2}{2c_{n_x}}\left(1 - \|\mathbf{x}\|^2\right) & if \quad \|\mathbf{x}\| < 1 \\ 0 & otherwise \end{cases} \tag{2.40}$$

The regularization step which effectively jitters the resampled values is

$$\mathbf{x}_k^{i^*} = \mathbf{x}_k^i + h_{opt} D_k \varepsilon^i \tag{2.41}$$

The theoretical disadvantage of the RPF is that its samples are no longer guaranteed to asymptotically approximate those from the posterior. The RPF filter algorithm 2.7 is presented in the following [2], [3].

Algorithm 2.7: Regularized Particle filter

$$\left[\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = RPF\left[\left\{\mathbf{x}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{z}_k\right]$$

- *FOR* $i = 1 : N_s$
  - ➢ *Draw* $\mathbf{x}_k^i \sim p(\mathbf{x}_k / \mathbf{x}_{k-1}^i)$

- ➢ *Assign each particle with the importance weight up to a normalizing constant according to*

$$\tilde{w}_k^i = p(\mathbf{z}_k / \mathbf{x}_k^i)$$

- *END FOR*
- *Calculate the total weight:* $t=SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i=1:N_s$

  - ➢ *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$

- *END FOR*
- *Calculate* $\hat{N}_{eff}$
- *IF* $\hat{N}_{eff} < N_{thr}$

  - ➢ *Calculate the empirical covariance matrix* S_k *of* $\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}$
  - ➢ *Compute* D_k *such that* $D_k D_k^T = S_k$
  - ➢ *Resample using systematic resampling or residual resampling.*
  - ➢ *FOR* $i=1:N_s$
    - *Draw* $\varepsilon^i \sim K$ *from the Epanechnikov/Gaussian kernel*
    - $\mathbf{x}_k^{i^*} = \mathbf{x}_k^i + h_{opt} D_k \varepsilon^i$
  - ➢ *END FOR.*

- *END IF.*

2.7 MCMC Move Step

The MCMC move step [2], [3] is based on the Metropolis-Hastings algorithm. The key idea is that a resampled particle $\mathbf{x}_k^i$ is moved to a new state $\mathbf{x}_k^{i^*}$ according to equation (2.41), only if $u \le \alpha$ where $u \sim U[0,1]$ and α is the acceptance probability. Otherwise, the move will be rejected.

The desired density of the MCMC step is $p(\mathbf{x}_{0:k}/\mathbf{z}_{0:k})$ which can be expressed as

$$p(\mathbf{x}_{0:k}/\mathbf{z}_{0:k}) = \frac{p(\mathbf{z}_k/\mathbf{x}_k)\,p(\mathbf{x}_k/\mathbf{x}_{k-1})}{p(\mathbf{z}_k/\mathbf{z}_{0:k-1})}\,p(\mathbf{x}_{0:k-1}/\mathbf{z}_{0:k-1}) \tag{2.42}$$

The Metropolis-Hastings acceptance probability is given by

$$\alpha = \min\left\{1, \frac{p(\mathbf{x}_{0:k}^{i^*}/\mathbf{z}_{0:k})\,q(\mathbf{x}_k^i/\mathbf{x}_k^{i^*})}{p(\mathbf{x}_{0:k}^i/\mathbf{z}_{0:k})\,q(\mathbf{x}_k^{i^*}/\mathbf{x}_k^i)}\right\} \tag{2.43}$$

Let $q(./\mathbf{x}_k^i)$ correspond to sampling according to (2.41). Since in this case $q(./\mathbf{x}_k^i)$ is symmetric in its arguments, i.e., $q(\mathbf{x}_k^{i^*}/\mathbf{x}_k^i) = q(\mathbf{x}_k^i/\mathbf{x}_k^{i^*})$, the substitution of (2.42) and (2.43) yields

$$\alpha = \min\left\{1, \frac{p(\mathbf{z}_k/\mathbf{x}_k^{i^*})\,p(\mathbf{x}_k^{i^*}/\mathbf{x}_{k-1}^i)}{p(\mathbf{z}_k/\mathbf{x}_k^i)\,p(\mathbf{x}_k^i/\mathbf{x}_{k-1}^i)}\right\} \tag{2.44}$$

The theoretical advantage of the MCMC move particle filter over the RPF is that its samples are guaranteed to asymptotically approximate those from the posterior. In practical scenarios, both the RPF and the MCMC move PF perform better than the SIR in cases where sample impoverishment is severe; for example when the process noise is small. The MCMC move PF algorithm 2.8 is presented in the following [2], [3].

Algorithm 2.8: MCMC move Particle filter

$$\left[\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}\right] = MCMC\left[\left\{\mathbf{x}_{k-1}^i, w_{k-1}^i\right\}_{i=1}^{N_s}, \mathbf{z}_k\right]$$

- *FOR* $i = 1:N_s$
  - *Draw* $\mathbf{x}_k^i \sim p(\mathbf{x}_k / \mathbf{x}_{k-1}^i)$
  - *Assign each particle with the importance weight up to a normalizing constant according to*

$$\tilde{w}_k^i = p(\mathbf{z}_k / \mathbf{x}_k^i)$$

- *END FOR*
- *Calculate the total weight:* $t=SUM\left[\left\{\tilde{w}_k^i\right\}_{i=1}^{N_s}\right]$
- *FOR* $i=1:N_s$

  - *Normalize the weights:* $w_k^i = t^{-1}\tilde{w}_k^i$

- *END FOR*
- *Calculate* $\hat{N}_{eff}$
- *IF* $\hat{N}_{eff} < N_{thr}$

  - *Calculate the empirical covariance matrix* S_k *of* $\left\{\mathbf{x}_k^i, w_k^i\right\}_{i=1}^{N_s}$
  - *Compute* D_k *such that* $D_k D_k^T = S_k$
  - *Resample using systematic resampling or residual resampling.*
  - *FOR* $i=1:N_s$
    - *Draw* $\varepsilon^i \sim K$ *from the Epanechnikov/Gaussian kernel*
    - $\mathbf{x}_k^{i^*} = \mathbf{x}_k^i + h_{opt} D_k \varepsilon^i$
    - *Draw* $u \sim U[0,1]$
    - $\alpha = \min\left\{1, \dfrac{p\left(\mathbf{z}_k / \mathbf{x}_k^{i^*}\right) p\left(\mathbf{x}_k^{i^*} / \mathbf{x}_{k-1}^i\right)}{p\left(\mathbf{z}_k / \mathbf{x}_k^i\right) p\left(\mathbf{x}_k^i / \mathbf{x}_{k-1}^i\right)}\right\}$
    - *If* $u \le \alpha$
      - *Move is accepted*
      - $\mathbf{x}_k^i = \mathbf{x}_k^{i^*}$
    - *end*

  - *END FOR.*

- *END IF.*

2.8 Simulation Results

The following nonlinear state space model is considered for the simulation of various particle filters, which is given by [2]

$$x_k = f_k(x_{k-1}, k) + v_{k-1} \tag{2.45}$$

$$z_k = \frac{x_k^2}{20} + n_k \tag{2.46}$$

where

$$f_k(x_{k-1}, k) = \frac{x_{k-1}}{2} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 8\cos(1.2k) \tag{2.47}$$

From the state space model (2.45) & (2.46), the prior density $p(x_k/x_{k-1})$ and likelihood function $p(z_k/x_k)$ are respectively given by

$$p(x_k/x_{k-1}) = N(x_k; f_k(x_{k-1}, k), Q_{k-1}) \tag{2.48}$$

$$p(z_k/x_k) = N\left(z_k; \frac{x_k^2}{20}, R_k\right) \tag{2.49}$$

It is assumed that in equations (2.45) & (2.46), v_{k-1} and n_k are zero mean Gaussian random variables with variances Q_{k-1} and R_k respectively. For the simulation of various particle filters in the MATLAB environment, the following parameters are used.

- Noise variances are $Q_{k-1} = 10$ and $R_k = 1$ respectively.
- Number of states M=100
- Number of particles N=10,100
- Number of Monte Carlo runs=100

The samples $\{x_k^i\}_{i=1}^N$ and the corresponding weights $\{w_k^i\}_{i=1}^N$ are generated using algorithm 2.8. The estimate of the state x_{est} is calculated by using the set of samples

$\{x_k^i\}_{i=1}^N$ and corresponding weights $\{w_k^i\}_{i=1}^N$, which is given by the sum of products of samples and corresponding weights.

$$x_{est} = \sum_{i=1}^{N} x_k^i w_k^i \tag{2.50}$$

To obtain the performance for state estimation, the Root Mean Square Error (RMSE) between the true state and estimated state is computed, which is given by

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(x - x_{est})^2} \tag{2.51}$$

The RMSEs are obtained by estimating the state x_k over 100 Monte Carlo runs.

Fig 2.4 shows 100 true values of the state x_k as a function of time k with Q=10 and R=1. Fig 2.5 shows the measurement process z_k as a function of time k with Q=10 and R=1. Figs 2.6 and Fig 2.7 show respectively the estimated state of various versions of particle filter for comparison with Q=10 (process noise) and R=1. In Fig 2.6, 10 particles are used for estimating the state x_k. The RMSE of ASIR filter is found to be 5.9090 which is slightly low as compared to SIR, RPF and MCMC filters whose values are found to be 6.5143, 6.2187 and 5.9310 respectively. In Fig 2.7 100 particles are used for estimating the state x_k. It may be noted here that there is a close similarity between the true states and estimated states by SIR, ASIR, RPF and MCMC filters. The RMSEs of ASIR, SIR, RPF and MCMC filters are found to be 5.0133, 5.0733, 5.1536, and 5.0752 respectively. It can be seen from the Fig2.6, that the SIR filter has high RMSE when 10 particles are used, whereas from Fig2.7, we can observe that by increasing the number of particles the RMSE performance of ASIR, SIR, RPF and MCMC filters are almost equal. So, to achieve smaller errors, we have to increase the number of particles. However if we increase the number of particles the computational complexity will increase. If we consider only the computational complexity, ASIR particle filter gives reasonably good results with less number of particles when compared to other particle filters.

Fig 2.8 shows 100 true values of the state x_k as a function of time k with Q=1 and R=10. Fig 2.9 shows the measurement process z_k as a function of time k with Q=1 and R=10. In Fig 2.10 & 2.11, the simulation is carried out by assuming Q=1 and R=10 (process noise is small compared to measurement noise) in which the sample impoverishment is severe. The RMSEs of ASIR, SIR, RPF and MCMC filters are found to be 4.5212, 4.8813, 4.5837 and 4.7193 respectively for 10 particles and similarly for 100 particles the RMSEs are found to be 4.3927, 4.4690, 4.4155 and 4.5170 respectively. Fig 2.12 shows 100 true values of the state x_k as a function of time k with Q=0.1 and R=10. Fig 2.13 shows the measurement process z_k as a function of time k with Q=0.1 and R=10. In Fig 2.14 & 2.15, the simulation is carried out by using Q=0.1 and R=10 (process noise is too small compared to the measurement noise). In this case the RMSEs of ASIR, SIR, RPF and MCMC filters are found to be 5.3600, 5.3145, 4.4396 and 3.9502 respectively for 10 particles and similarly for 100 particles the RMSEs are found to be 3.7252, 3.7784, 3.7679 and 3.6732 respectively. From figures 2.10 and 2.14 we can say that RPF and MCMC PFs give reasonably good results than ASIR and SIR particle filters, when the process noise is very small. From figures 2.11 and 2.15 we can say that by increasing the number of particles the RMSE performance of ASIR, SIR, RPF and MCMC filters are almost similar. For RPF and MCMC PF smaller errors can be achieved even with less number of particles. Similarly by comparing the RMSEs of RPF and MCMC PF we can say that MCMC PF gives reasonably good results as compared to RPF, because the samples of MCMC move particle filter are guaranteed to asymptotically approximate those from the posterior whereas the samples from RPF will not be guaranteed to approximate the posterior. So if we consider the computational complexity, MCMC Particle Filter is giving reasonably good results with less number of particles when compared to other Particle Filters, when the process noise is very small compared to measurement noise.

Fig 2.16 shows 100 true values of the state x_k as a function of time k with Q=1 and R=1. Fig 2.17 shows the measurement process z_k as a function of time k with Q=1 and R=1. In Fig 2.18 & 2.19, the simulation is carried out by taking Q=1 and R=1 (process noise and measurement noise is equal). In this case the RMSEs of ASIR, SIR, RPF and MCMC filters are found to be 4.3267, 4.4594, 4.2441 and 4.5135 respectively

for 10 particles and similarly for 100 particles the RMSEs are found to be 3.5267, 3.5073, 3.7470 and 3.7738 respectively. From figures 2.18 and 2.19 we can say that the RMSE performances of all the Particle Filters are almost similar and by increasing the number of particles we can achieve the smaller errors.

Comparing all the simulations and by considering only computational complexity, we can conclude that, the choice of ASIR will give reasonably good results with less number of particles when compared to other Particle Filters, when the process noise is equal to or greater than the measurement noise. Similarly when the process noise is very small compared to the measurement noise where the sample impoverishment problem will be severe we can say that MCMC move particle filter will give reasonably good results compared to the other versions of particle filters.

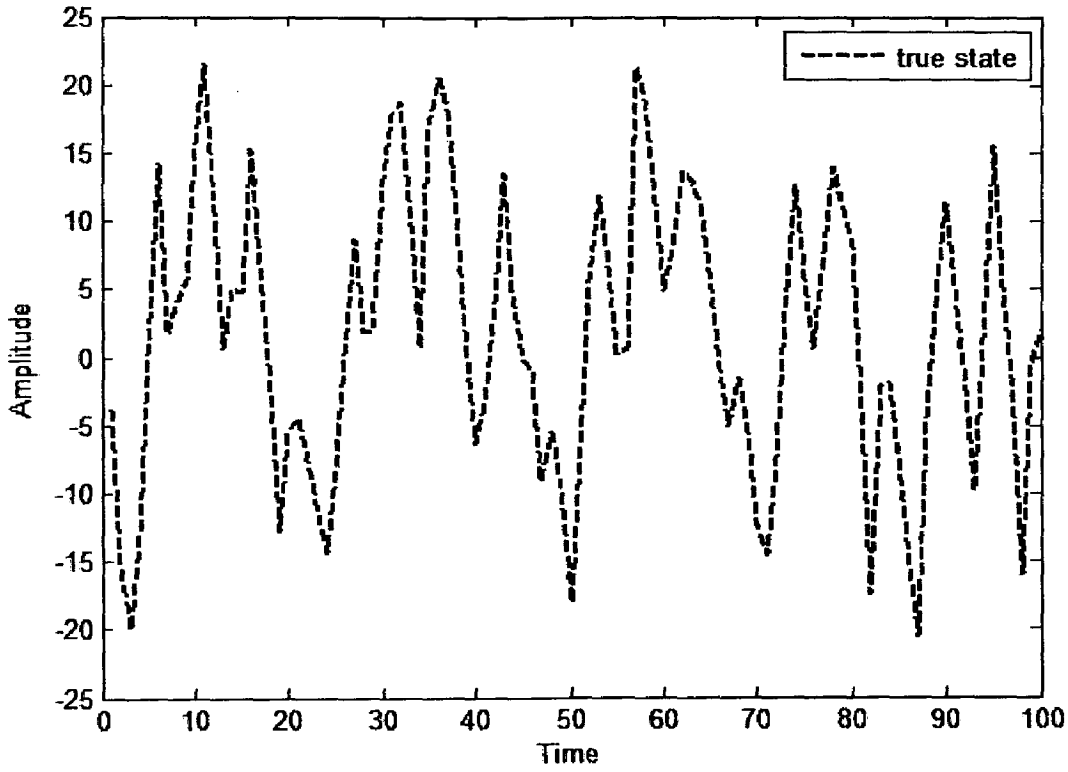


Figure 2.4 True values of the state x_k as a function of time k with Q=10 and R=1.

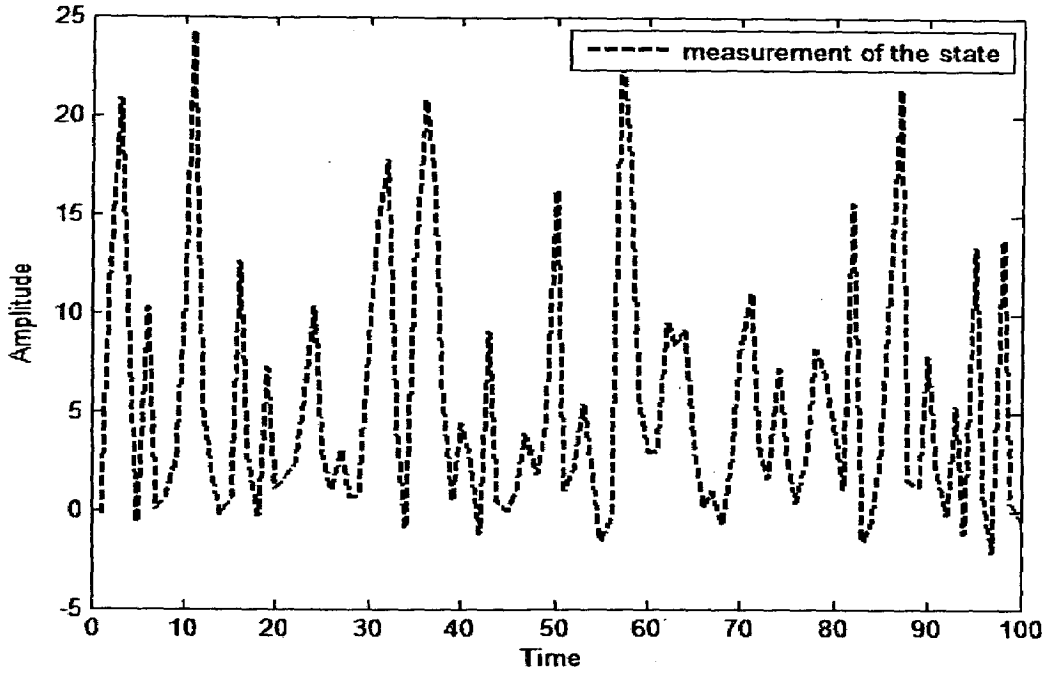


Figure 2.5 Measurement process z_k of the state x_k as a function of time k with Q=10 and R=1.

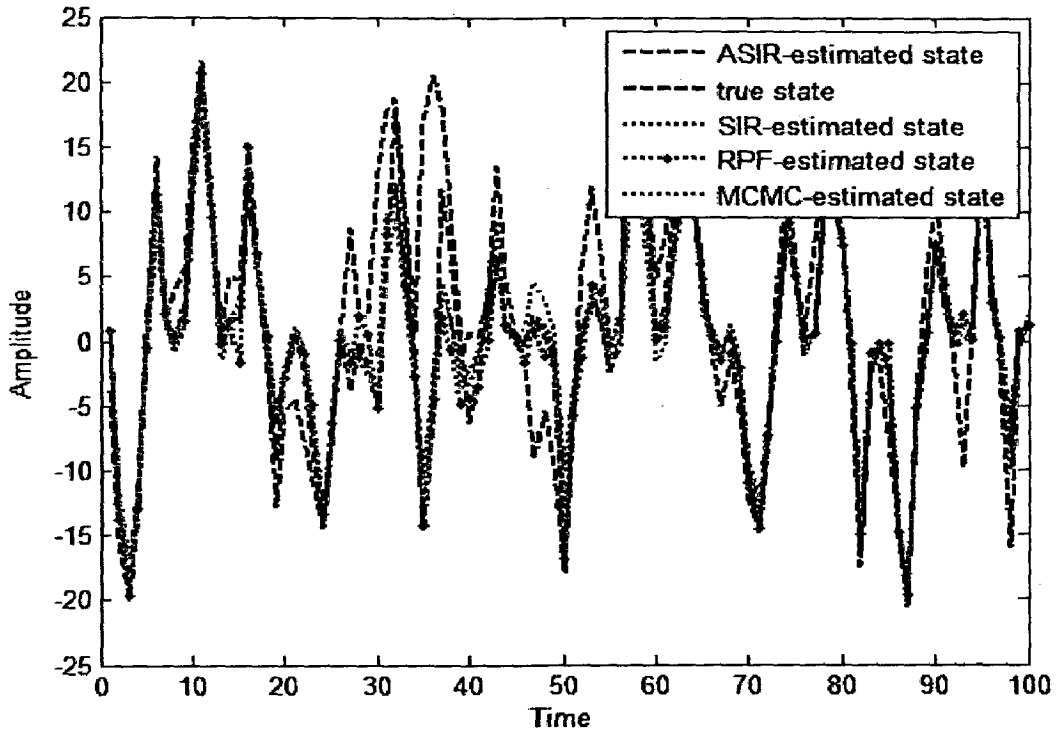


Figure 2.6 True and estimated values of the state x_k as a function of time k for 10 particles with Q=10 and R=1.

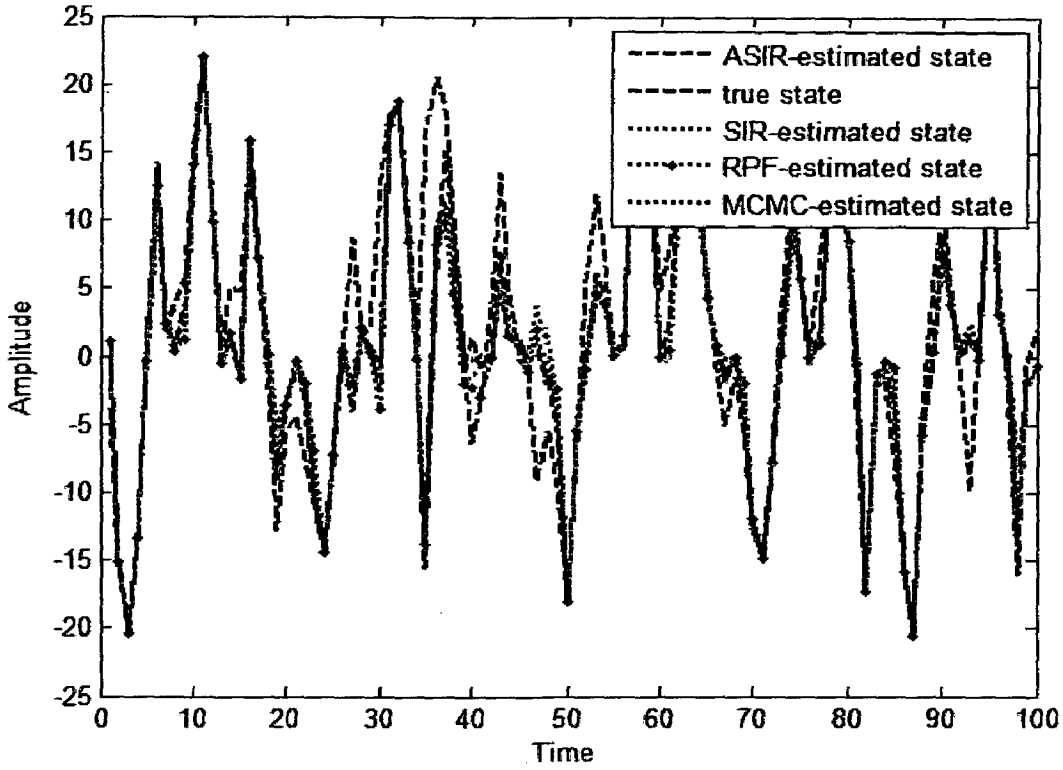


Figure 2.7 True and estimated values of the state x_k as function of time k for 100 particles with Q=10 and R=1.

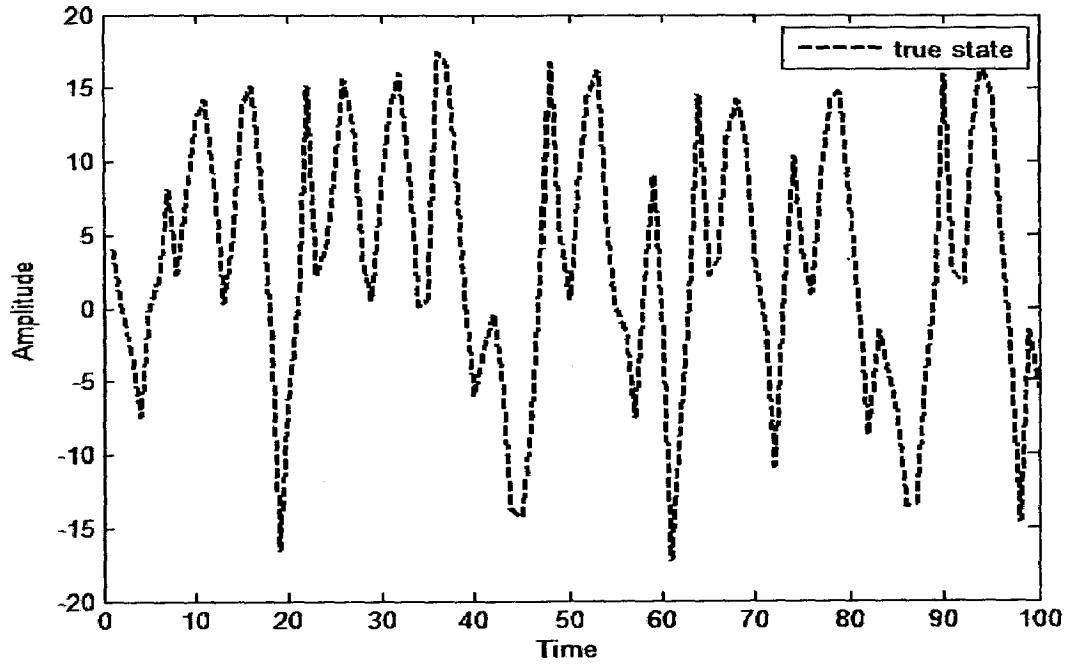


Figure 2.8 True values of the state x_k as a function of time k with Q=1 and R=10.

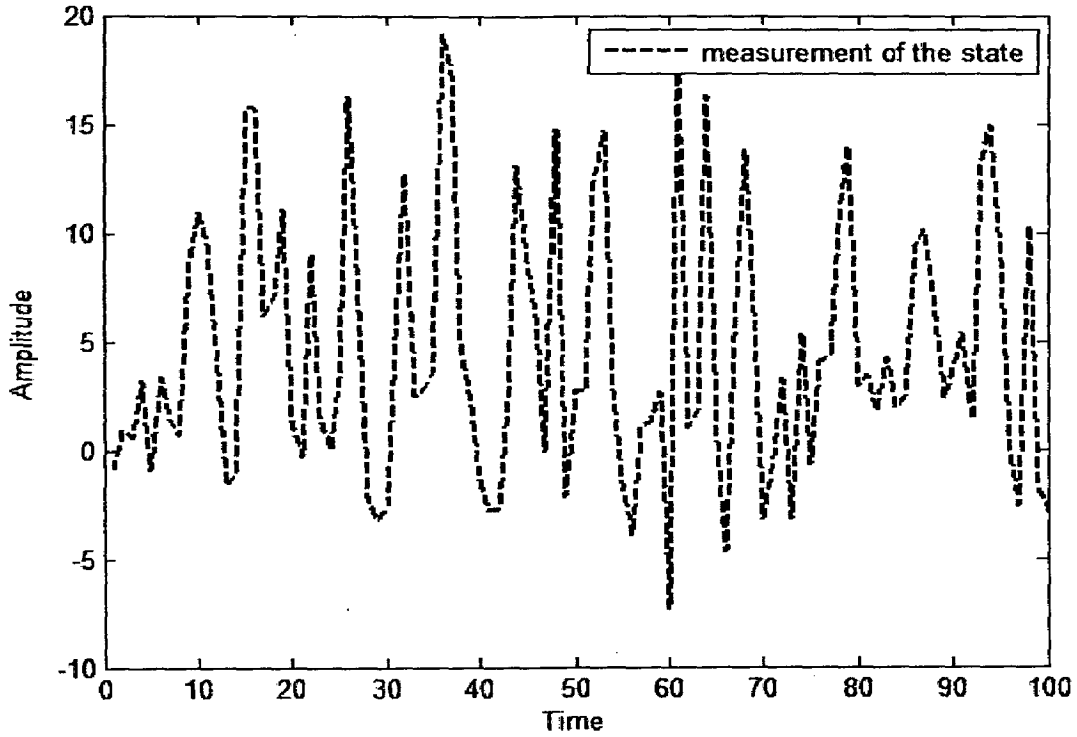


Figure 2.9 Measurement process z_k of the state x_k as a function of time k with Q=1 and R=10.

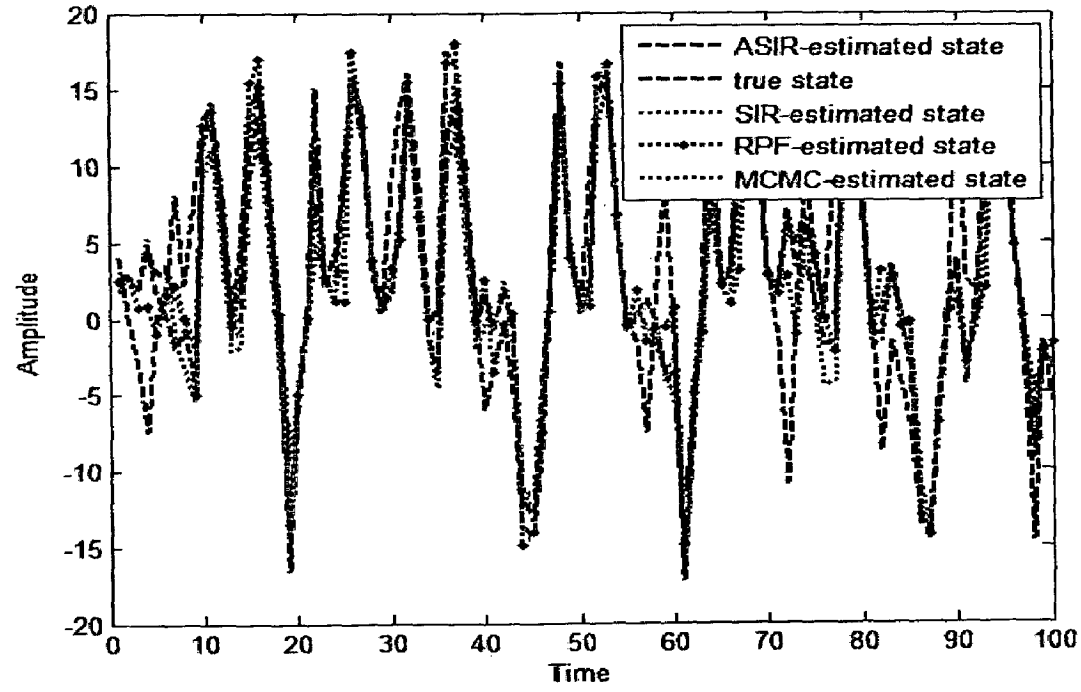


Figure 2.10 True and estimated values of the state x_k as a function of time k for 10 particles with Q=1 and R=10.

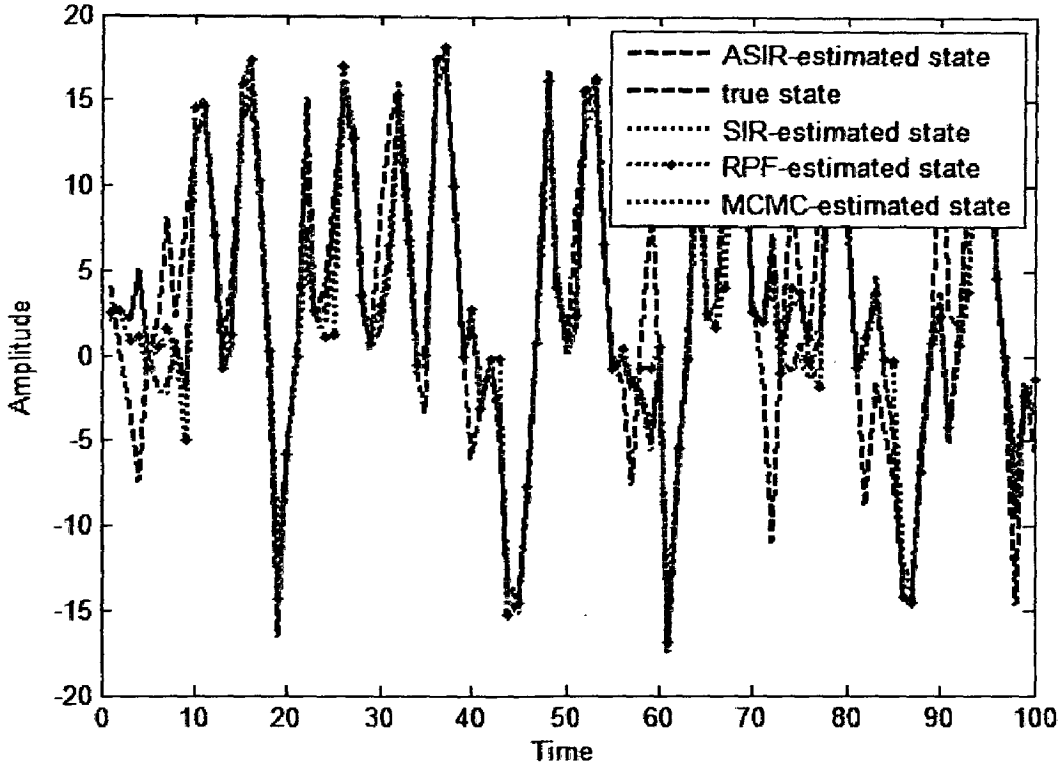


Figure 2.11 True and estimated values of the state x_k as a function of time k for 100 particles with Q=1 and R=10.

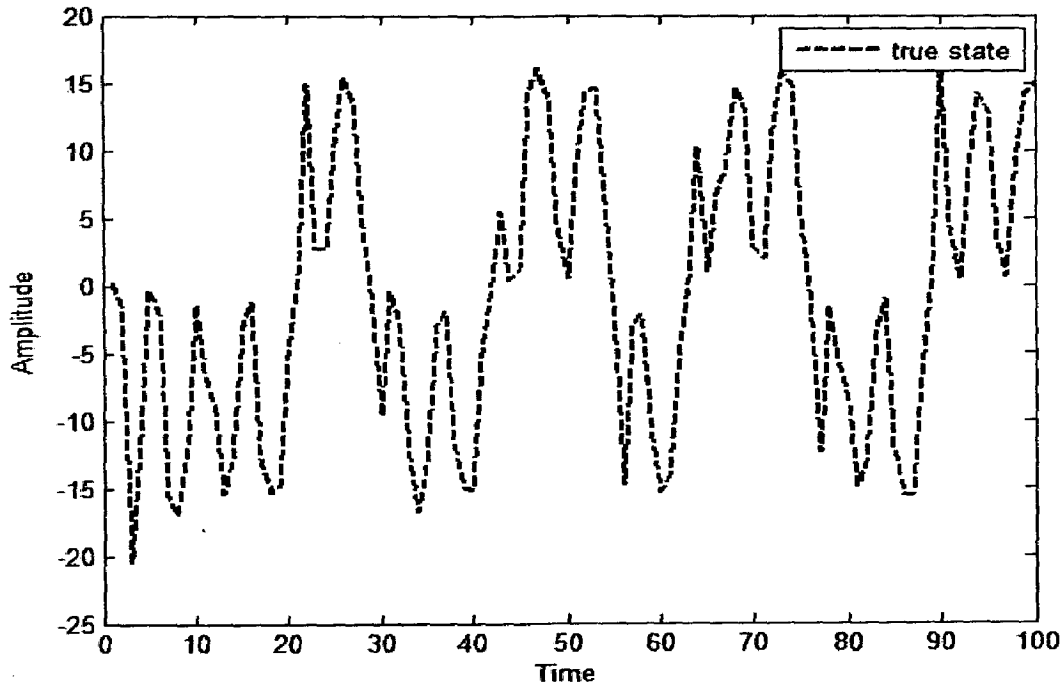


Figure 2.12 True values of the state x_k as a function of time k with Q=0.1 and R=10.

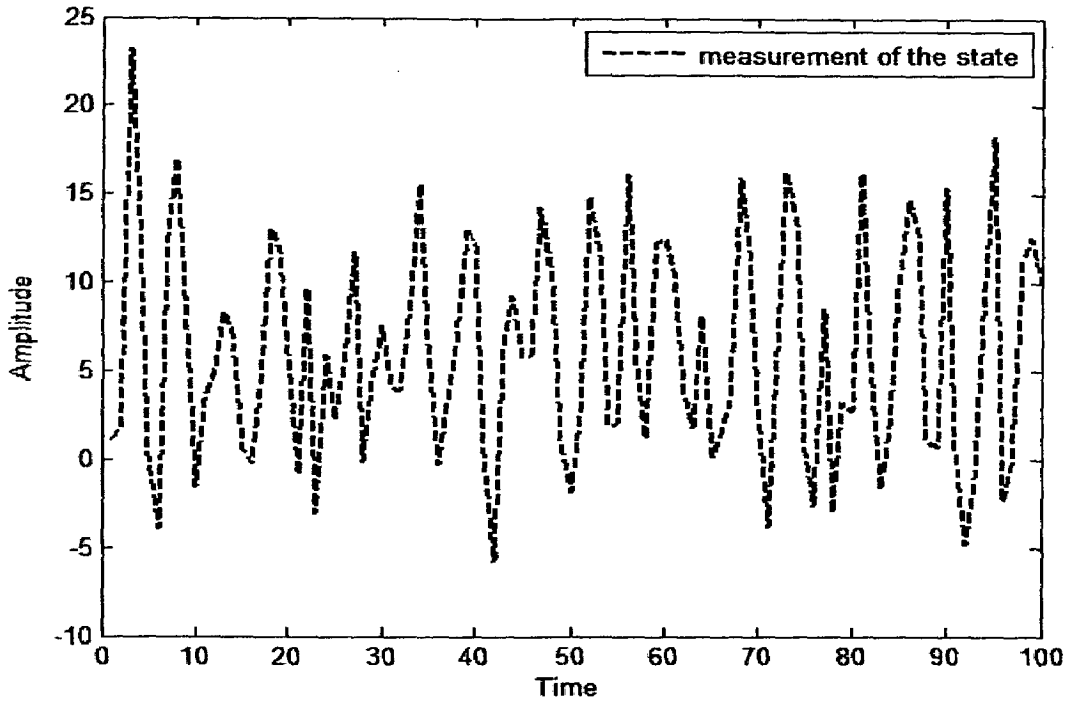


Figure 2.13 Measurement process z_k of the state x_k as a function of time k with Q=0.1 and R=10.

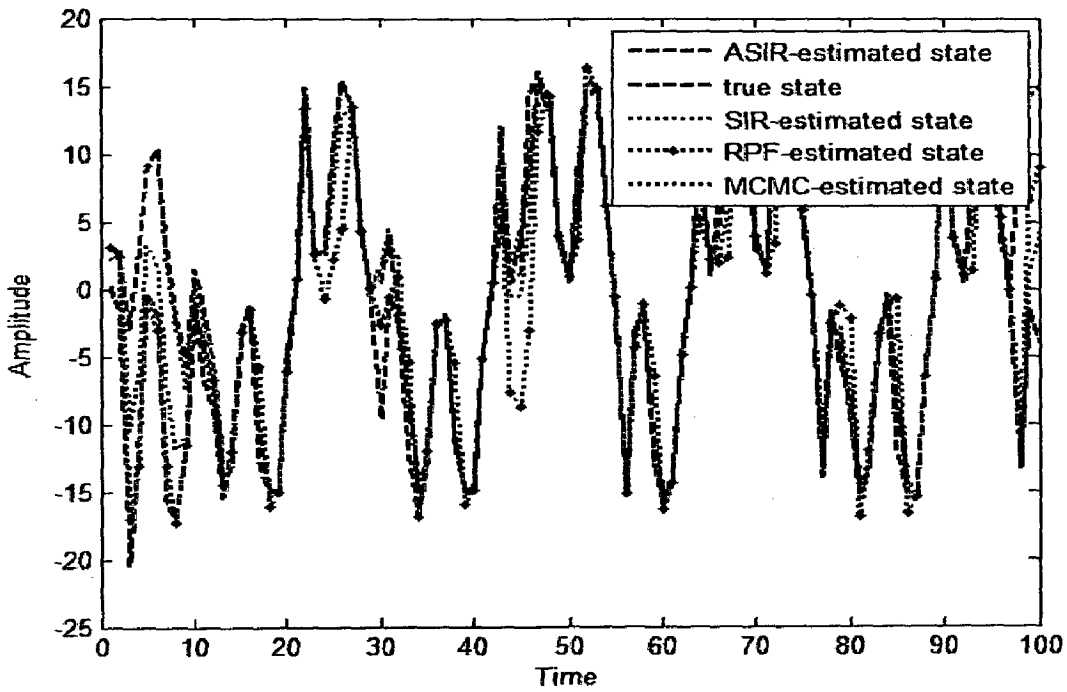


Figure 2.14 True and estimated values of the state x_k as a function of time k for 10 particles with Q=0.1 and R=10.

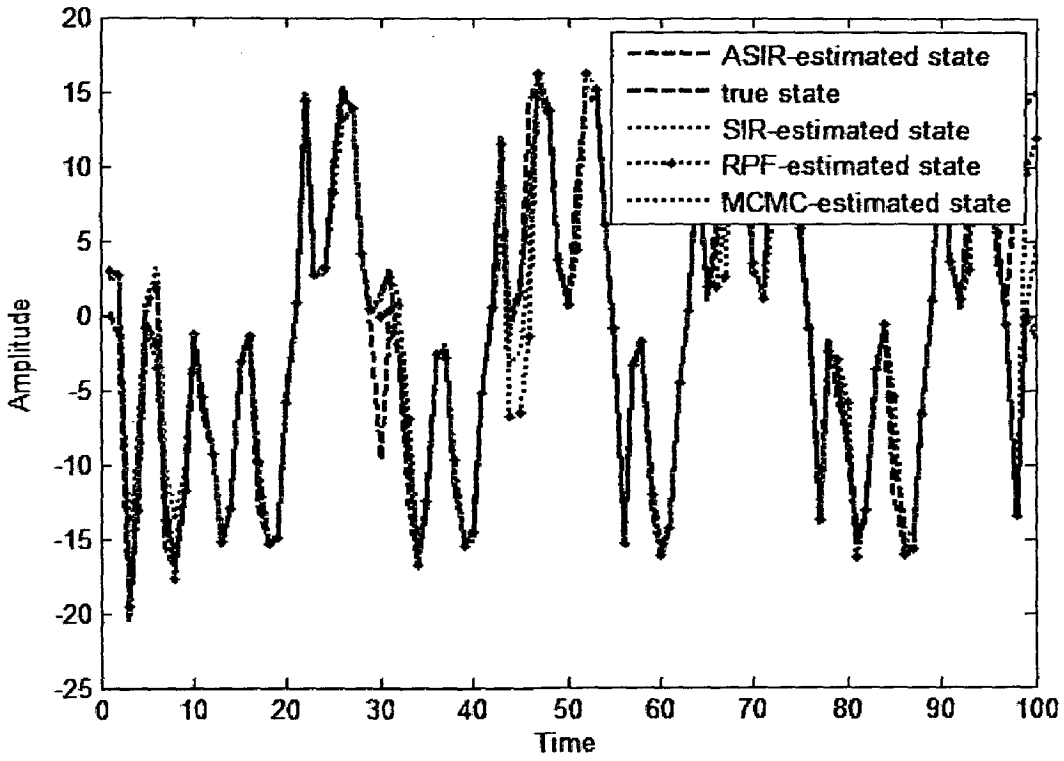


Figure 2.15 True and estimated values of the state x_k as a function of time k for 100 particles with Q=0.1 and R=10.

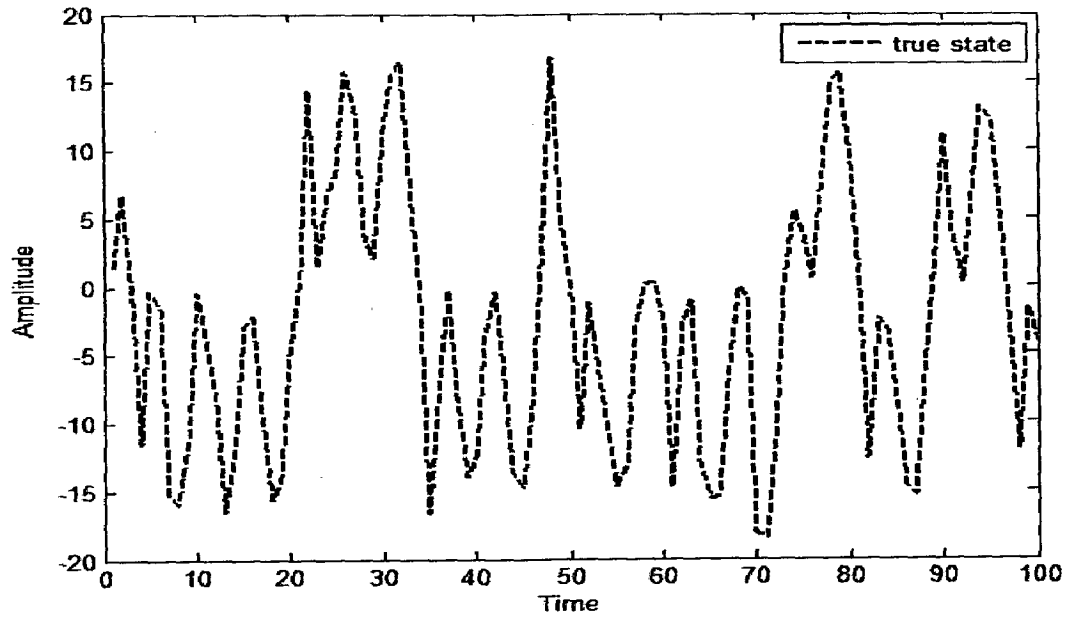


Figure 2.16 True values of the state x_k as a function of time k with Q=1 and R=1.

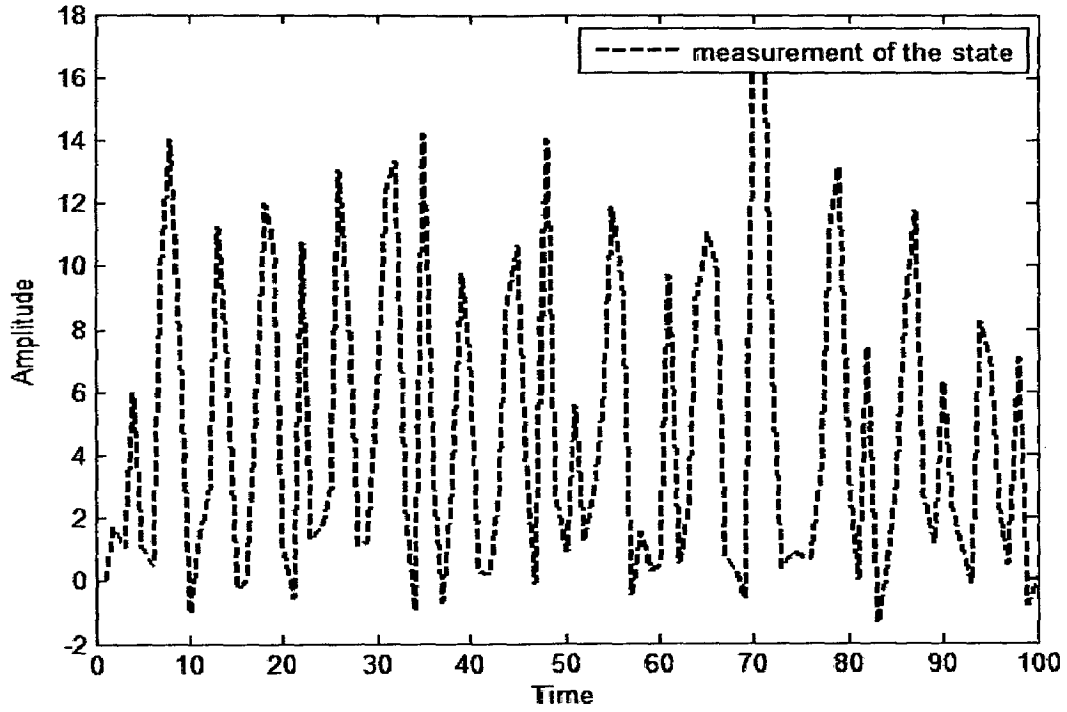


Figure 2.17 Measurement process z_k of the state x_k as a function of time k with Q=1 and R=1.

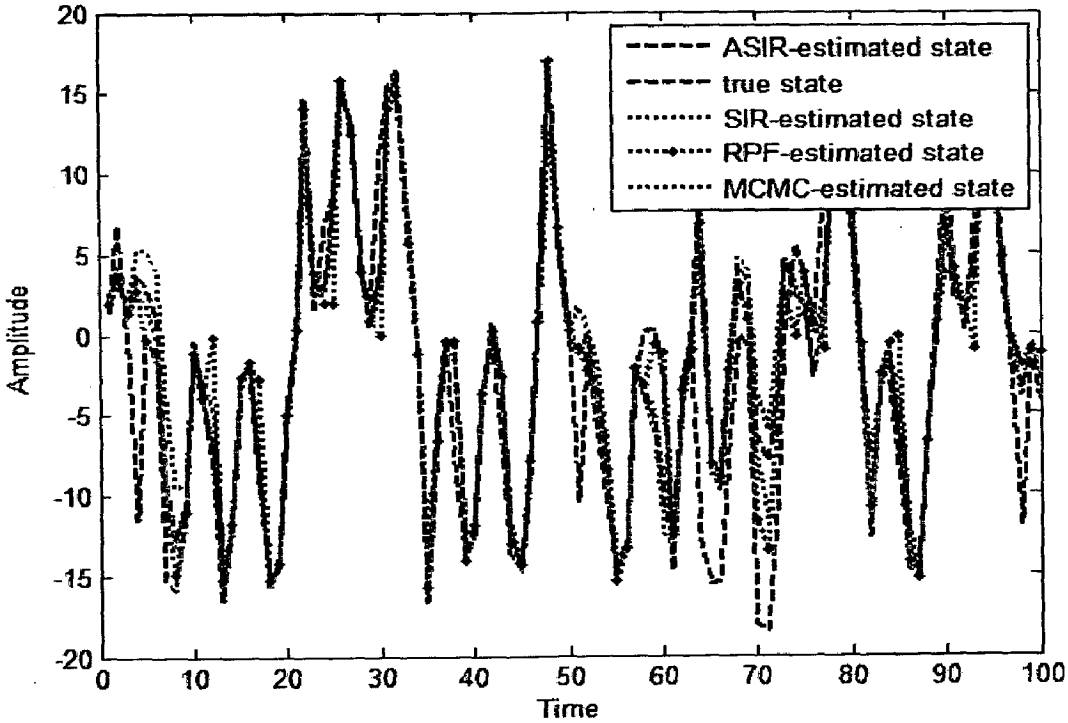


Figure 2.18 True and estimated values of the state x_k as a function of time k for 10 particles with Q=1 and R=1.

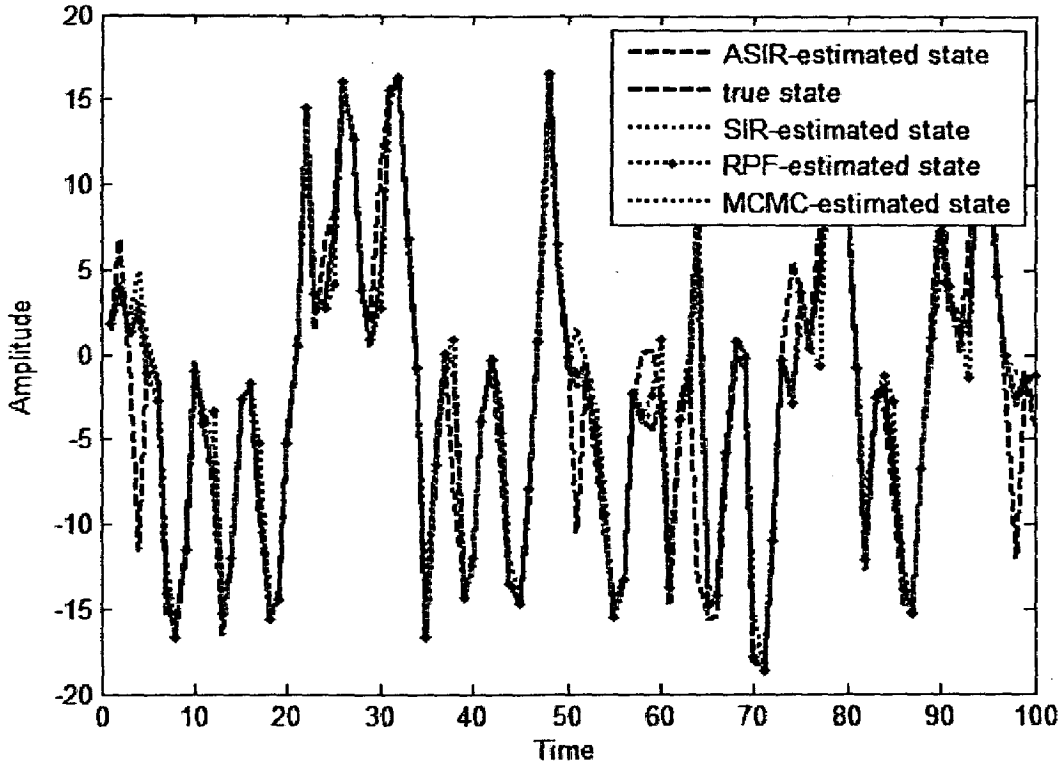


Figure 2.19 True and estimated values of the state x_k as a function of time k for 100 particles with Q=1 and R=1.

Chapter 3

TRACKING IN BINARY SENSOR NETWORKS

In this chapter, binary sensor network (BSN) model is introduced first, by considering it as a special case of wireless sensor networks for tracking the target within a sensor field monitored by a sensor network. Unlike sensors considered in traditional tracking approaches, binary sensors provide only one bit of data indicating presence or absence of a target in the sensing range. They are incapable of producing any other information. The signals that reach the fusion center of these networks are therefore binary signals embedded in noise, and they pose challenging problems for recovering the sensed information by the sensors. The central unit uses a model for the target movement in the sensor field and estimates the target's trajectory, velocity, and power using the received data. The mathematical formulation of the tracking problem in binary sensor networks is described and two particle filtering algorithms are presented namely, auxiliary particle filtering (APF) and cost reference particle filtering (CRPF) for processing of the binary data. A detailed derivation of Posterior Cramer-Rao Bound (PCRB) for discrete time nonlinear filtering is presented. Finally the simulation results of APF, CRPF and PCRB are presented.

3.1 Binary Sensor Networks

Sensor networks have two major requirements. [12]

- Efficient networking and energy-saving techniques are required, as the sensors have to communicate with one another or with a "base" to transmit readings or results of the local computation.
- The fusion center should be efficient in processing the information gathered by sensors.

It is not practical to rely on sophisticated sensors with large power supply and communication demands. Simple, inexpensive individual devices deployed in large numbers are likely to be the source of the battlefield awareness in the future. For example [12], if the sensors are obtaining sound levels, instead of using the sophisticated sensors for detecting the absolute sound level (which may cause confusion between loud near objects and quieter close objects), the sensor may simply report whether the sound is getting louder or quieter. Similarly for the seismic sensor, an increase or decrease in intensity can be used. In these systems, using a

single bit of information allows for inexpensive sensing as well as minimal communication. This minimalist approach to extracting information from sensor networks lead to a binary model of sensor networks. Binary sensors [15] are the sensors that transmit only binary information about sensed events (the event is sensed or is not sensed). The signals that reach the fusion center of these networks are therefore binary signals embedded in noise, and they pose challenging problems for recovering the sensed information by the sensors.

3.1.1 The Binary Sensor Network Model

Binary sensor network [15] consists of sensors that measure the signal. Unlike sensors considered in traditional tracking approaches, binary sensors provide only one bit of data indicating presence or absence of a target in the sensing range. They are incapable of producing any other information. If the level of the measured signal is above a predefined threshold, they report to the fusion center with a signal that identifies them; otherwise they are silent. The binary sensor network is shown in Fig 3.1.

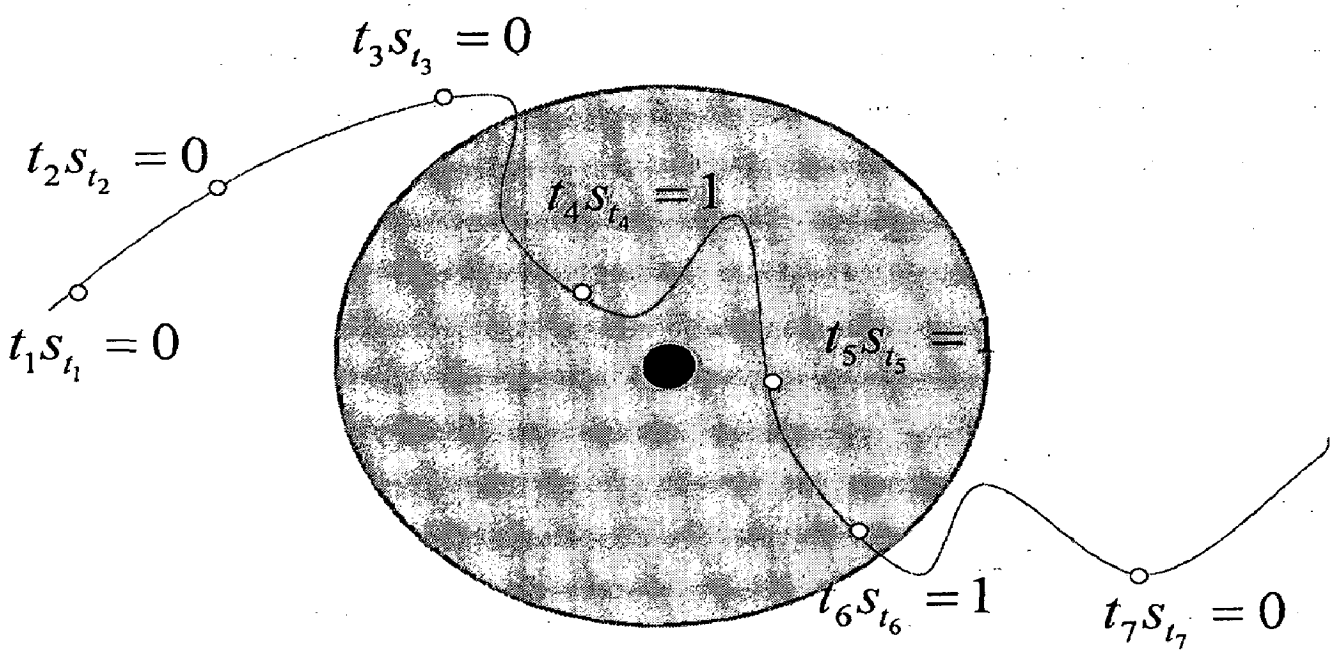


Fig.3.1. Binary sensor network with a target passing nearby

In the Fig 3.1, a binary sensor is represented by the small circle and its range by the larger circle. When the target is outside the range of the sensor, the received signal is below the set threshold, and the sensor does not transmit anything. During the time when the target is inside the range of the sensor, the received signal is above the threshold, and the sensor transmits a "one" to the fusion center. When at a given time the fusion center does not receive

a signal from a particular sensor, this implies that the sensor transmits a "zero." The network consists of binary sensors that may be deployed randomly, deterministically, or both. In all cases, the fusion center is assumed to know the locations of all the sensors and that the locations remain fixed for all time.

3.2 Mathematical formulation of tracking problem

The objective of tracking is to recursively estimate the target state, i.e., the position and the velocity of the target, based on received local sensor data. The sensors are assumed to be stationary and the fusion center has perfect information about the locations of sensors. Now the tracking problem can be formulated [15] based on the target dynamic model and the measurement model.

Target Dynamic Model:

The movement of the target as described in [13] is its position (p_t), velocity (v_t), and acceleration (a_t). From the differential equations $\dot{p}_t = v_t$ and $\dot{v}_t = a_t$, the expressions for the position and velocity are given by

$$p_t = p_0 + v_0 t + a_0 \frac{t^2}{2} \tag{3.1}$$

$$v_t = v_0 + a_0 t \tag{3.2}$$

If we substitute the sample period (T_s) in place of t, then the discrete time model for motion between two consecutive measurements is obtained. The general model [15] for the target movement is given by

$$\mathbf{x}_t = \mathbf{G}_x \mathbf{x}_{t-1} + \mathbf{G}_u \mathbf{u}_t \tag{3.3}$$

Where $\mathbf{x}_t = [x_{1,t}, x_{2,t}, \dot{x}_{1,t}, \dot{x}_{2,t}]^T \in \Re^4$ is the state vector which indicates the position and velocity of the target in a two dimensional Cartesian coordinate system.

$\mathbf{G}_x = \begin{pmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ and $\mathbf{G}_u = \begin{pmatrix} \frac{T_s^2}{2} & 0 \\ 0 & \frac{T_s^2}{2} \\ T_s & 0 \\ 0 & T_s \end{pmatrix}$ are known matrices.

$\mathbf{u}_t$ is 2×1 vector representing the state space noise process(acceleration), and it is assumed to be a Gaussian vector with a covariance matrix $\mathbf{c}_u = \text{diag}\left(\sigma_{u_1}^2, \sigma_{u_2}^2\right)$.

Measurement Model:

The power measured by the n th sensor [14] is given by

$$y_{n,t} = g_n(\mathbf{x}_t) + v_{n,t} \tag{3.4}$$

$$= \frac{\psi d_0^{\alpha}}{\| \mathbf{r}_n - \mathbf{l}_t \|^{\alpha}} + v_{n,t} \qquad n = 1, 2, \ldots, N$$

Where

$g_n(.)$ is a function that models the received signal power by the nth sensor and,

$v_{n,t}$ is noise process independent from $\mathbf{u}_t$ and independent from noise samples of other sensors..

$\mathbf{r}_n \in \Re^2$ is the position of the nth sensor.

$\mathbf{l}_t = \begin{bmatrix} x_{1,t} & x_{2,t} \end{bmatrix}^T$ is the location of the target at time t.

$\| \mathbf{r}_n - \mathbf{l}_t \|$ denotes the Euclidean distance between $\mathbf{r}_n$ and $\mathbf{l}_t$;

ψ is the emitted power of the target measured at a reference distance d_0;

α is an attenuation parameter that depends on the transmission medium and is considered to be known and the same for all sensors.

$v_{n,t}$ can be approximated well with a normal distribution, namely, $v_{n,t} \sim N(\mu_v, \sigma_v^2)$ where $\mu_v = \sigma^2$ with σ^2 being the known power of the background measurement noise of one sample and $\sigma_v^2 = 2\sigma^4 / L$, with L being the number of samples used to obtain the measured power.

The nth sensor measures the received power [15] $y_{n,t}$ and compares it with the threshold γ. If the measured value is below the threshold γ, it does not transmit anything and if the measured power is greater than the threshold γ, the sensor transmits its identification code to the fusion center. Therefore, the sensors in the network send signals to the fusion center only if the received power is greater than the sensor thresholds. The received signal from the n^{th} sensor at the fusion center is given by

$$z_{n,t} = \beta_n s_{n,t} + \varepsilon_{n,t} \tag{3.5}$$

where

$$s_{n,t} = \begin{cases} 1 & if \quad y_{n,t} > \gamma \\ 0 & if \quad y_{n,t} \le \gamma \end{cases} \tag{3.6}$$

$\varepsilon_{n,t}$ is the observation noise and is modelled as AWGN with zero mean and variance σ_ε^2.

β_n is a known attenuation coefficient associated with the n^{th} sensor.

In summary, the measurements made by the sensors are complete and are modelled by (3.4). The sensors, however, always transmit binary signals constructed according to (3.6), and the fusion center receives them as quantified by (3.5). Now the objective is to track the evolving state $\mathbf{x}_{0:t} = (\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_t)$ using the observations $\mathbf{z}_{1:t} = (z_{1,1:t}, ...z_{N,1:t})$, that is, the observations up to time instant t of the first sensor, $z_{1,1:t}$, the second sensor, $z_{2,1:t}$, as well as the remaining $N-2$ sensors, $z_{3,1:t}, ...z_{N,1:t}$. Therefore by the use of the observations $\mathbf{z}_{1:t}$, the state vector $\mathbf{x}_{0:t}$ can be estimated by the particle filter. The APF uses probabilistic assumptions about all the noise processes in the model and about the prior of the states. The CRPF only needs knowledge of the first moments of the noise process.

3.3 Particle Filtering approach for target tracking in Binary Sensor Networks

In [15] two types of particle filters have been considered for tracking the target in binary sensor network, these are

1) The auxiliary particle filter (APF)

2) The cost-reference particle filter (CRPF).

The two methods are sequential statistical signal processing procedures with distinct features but with similar algorithmic outline. Tracking algorithm based on APF is discussed in section -3.3.1 followed by the CRPF algorithm in section-3.3.2.

3.3.1 APF Algorithm: [15], [19]

According to the theory of particle filtering, *a posterior* distribution of $\mathbf{x}_{0:t}$, $p(\mathbf{x}_{0:t} / \mathbf{z}_{1:t})$, can be tracked by approximating it with a random measure χ_t which can be defined as $\chi_t = \left\{\mathbf{x}_{0:t}^{(m)}, w_t^m\right\}_{m=1}^{M}$ where m is an index and M being the number of particles. At every time instant t, the particle filter carries out the following operations:

(1) Selection of most promising particle streams

(2) Particle propagation

(3) Computation of particle weights

(4) State estimation.

The APF draw samples from an importance function which is as close as possible to the optimal one. The selection of most promising particles is carried out by sampling from a multinomial distribution where the number of possible outcomes is M and the probabilities of the respective outcomes are $\tilde{w}_t^m$, $m = 1, 2, ..., M$, and Recalling the equation (2.37), we have

$$\tilde{w}_t^m \propto p\left(\mathbf{z}_t / \mu_t^{(m)}\right) w_{t-1}^{(m)} \tag{3.7}$$

where $\mu_t^{(m)}$ is some parameter that characterizes $\mathbf{x}_t^{(m)}$ given $\mathbf{x}_{t-1}^{(m)}$

Since the noise samples $\varepsilon_{n,t}$ in (3.5) are assumed independent, we have

$$p\left(\mathbf{z}_t / \mu_t^{(m)}\right) = \prod_{n=1}^{N} p\left(z_{n,t} / \mu_t^{(m)}\right) \tag{3.8}$$

where $p\left(z_{n,t} / \mu_t^{(m)}\right)$, can be written as

$$\begin{aligned} p\left(z_{n,t} / \mu_t^{(m)}\right) &= p\left(z_{n,t} / s_{n,t} = 0, \mu_t^{(m)}\right) P\left(s_{n,t} = 0 / \mu_t^{(m)}\right) \\ &\quad + p\left(z_{n,t} / s_{n,t} = 1, \mu_t^{(m)}\right) P\left(s_{n,t} = 1 / \mu_t^{(m)}\right) \\ &= p\left(z_{n,t} / s_{n,t} = 0\right) P\left(s_{n,t} = 0 / \mu_t^{(m)}\right) \\ &\quad + p\left(z_{n,t} / s_{n,t} = 1\right) P\left(s_{n,t} = 1 / \mu_t^{(m)}\right) \end{aligned} \tag{3.9}$$

where

$$p\left(z_{n,t} / s_{n,t}\right) = N\left(\beta_n s_{n,t}, \sigma_\varepsilon^2\right) \tag{3.10}$$

and

$$P\left(s_{n,t} = 1 / \mu_t^{(m)}\right) = Q\left(\frac{\gamma - g_n\left(\mu_t^{(m)}\right) - \mu_v}{\sigma_v}\right) \tag{3.11}$$

$$P\left(s_{n,t} = 0 / \mu_t^{(m)}\right) = 1 - Q\left(\frac{\gamma - g_n\left(\mu_t^{(m)}\right) - \mu_v}{\sigma_v}\right) \tag{3.12}$$

where $Q(.)$ denotes the complement of the standard normal cumulative distribution function.

At the beginning, the initial set of particles $\mathbf{x}_0^{(m)}$, $m = 1, 2, ..., M$, are drawn from a prior distribution $\pi(\mathbf{x}_0)$, and the weights of the particles are set to $1/M$. Suppose now that at

time instant t-1, we have the random measure $\chi_{t-1} = \left\{ \mathbf{x}_{0:t-1}^{(m)}, w_{t-1}^{m} \right\}_{m=1}^{M}$. Then the steps of a particle filter recursion can be implemented as follows.

1) Selection of Most Promising Particle Streams: For selection of the most promising particles, the conditional mean of $\mathbf{x}_t^{(m)}$ given $\mathbf{x}_{t-1}^{(m)}$ is used as a characterizing parameter of every stream, i.e.

$$\mu_t^{(m)} = E\left(\mathbf{x}_t / \mathbf{x}_{t-1}^{(m)} \right) \tag{3.13}$$

Now by applying conditional mean to equation (3.3), we have

$$\mu_t^{(m)} = \mathbf{G}_x \mathbf{x}_{t-1}^{(m)} \tag{3.14}$$

From the above equation we can readily compute the conditional mean. This is followed by computation of the weights according to (3.7) and their normalization. Finally, a set of indices $\{i_m\}$ are drawn from the probability mass function (pmf) represented by the normalized weights.

2) New Particle Generation: The first two elements of the four-dimensional state $\mathbf{x}_t$ represent the location of the target in a two-dimensional space, and the remaining elements are the components of the velocity in this space. That implies that the generation of $\mathbf{x}_t$ requires drawing only two dimensional random variables. The generation can be carried out by first, propagating the velocity components one step ahead using the joint distribution $p\left(\dot{x}_{1,t}, \dot{x}_{2,t} / \dot{x}_{1,t-1}, \dot{x}_{2,t-1} \right)$ or $p\left(\dot{x}_{1,t}, \dot{x}_{2,t} / \dot{x}_{1,t-1}, \dot{x}_{2,t-1}, \mathbf{z}_t \right)$ and second, computing the locations according to

$$x_{1,t}^{(m)} = x_{1,t-1}^{(i_m)} + \frac{T_s}{2}\left(\dot{x}_{1,t}^{(m)} + \dot{x}_{1,t-1}^{(i_m)} \right) \tag{3.15}$$

$$x_{2,t}^{(m)} = x_{2,t-1}^{(i_m)} + \frac{T_s}{2}\left(\dot{x}_{2,t}^{(m)} + \dot{x}_{2,t-1}^{(i_m)} \right) \tag{3.16}$$

The above equations are obtained from (3.3).

3) Weight Computation: The newly generated particles are assigned weights according to

$$w_t^m \propto \frac{p\left(\mathbf{z}_t / \mathbf{x}_t^{(m)} \right)}{p\left(\mathbf{z}_t / \mu_t^{(i_m)} \right)}$$

The likelihood terms of the numerator and denominator are calculated as in the APF using (3.8)–(3.13).

4) State Estimation: Once the weights are normalized, one can use χ_t to compute estimates of the unknown states. If the estimation is minimum mean square error (MMSE) estimation, then it can be obtained from

$$\hat{\mathbf{x}}_t = \sum_{m=1}^{M} w_t^m \mathbf{x}_t^{(m)} \tag{3.17}$$

3.3.2 CRPF Algorithm

The objective of CRPF [15], [20] is to estimate sequentially the evolution of the unknown state $\mathbf{x}_{0:t}$ from $\mathbf{z}_{1:t}$ without assumptions about the probability distributions of noise processes in the model. It is similar in structure to that of the APF because CRPF also uses the same discrete random measure. This random measure is composed of particles and costs associated to them, where the costs are user-defined. The random measure can be denoted by $\zeta_t = \left\{\mathbf{x}_{0:t}^{(m)}, C_t^{(m)}\right\}_{m=1}^{M}$, where $\mathbf{x}_{0:t}^{(m)}$ has the same meaning as before and $C_t^{(m)}$ are the associated costs to $\mathbf{x}_{0:t}^{(m)}$. It is clear that the costs will play the role of the weights in APF. With appropriate choices of the costs, the CRPF can be made equivalent to the APF or the standard particle filter.

In general, the costs are updated according to

$$\begin{aligned} C_t^{(m)} &= C\left(\mathbf{x}_{0:t}^{(m)}/\mathbf{z}_{1:t}\right) \\ &= \lambda C\left(\mathbf{x}_{0:t-1}^{(m)}/\mathbf{z}_{1:t-1}\right) + \Delta C\left(\mathbf{x}_t^{(m)}/\mathbf{z}_t\right) \end{aligned} \tag{3.18}$$

Where λ is a forgetting factor $(0 \le \lambda \le 1)$, and $\Delta C\left(\mathbf{x}_t^{(m)}/\mathbf{z}_t\right)$ is an incremental cost. Obviously, the value of λ controls how fast the state estimates can adapt to new values of the states. The incremental cost contributes to the total cost at time instant t and is a function of the particle value and the observation at that instant. A typical incremental cost has the form

$$\Delta C\left(\mathbf{x}_t^{(m)}/\mathbf{z}_t\right) = \| \mathbf{z}_t - \hat{\mathbf{z}}_t^{(m)} \|^q \tag{3.19}$$

Where $\hat{\mathbf{z}}_t^{(m)}$ is a function of $\mathbf{x}_t^{(m)}$, and $q > 0$. So, CRPF proceeds analogously to the APF; with the vector of observations $\mathbf{z}_t$, the discrete random measure at time instant $t-1$,

$\zeta_{t-1} = \left\{\mathbf{x}_{0:t-1}^{(m)}, C_{t-1}^{(m)}\right\}_{m=1}^{M}$, is updated to $\zeta_t = \left\{\mathbf{x}_{0:t}^{(m)}, C_t^{(m)}\right\}_{m=1}^{M}$ to reflect accurately the possible value of the unknown state at time instant t, $\mathbf{x}_t$. The procedure has four steps:

(1) Selection of most promising particle streams

(2) Propagation of particles

(3) Cost update

(4) State estimation.

The initialization of the method is carried out by randomly drawing initial particles from some probability density function (pdf) $\pi(\mathbf{x}_0)$ whose support includes the space of $\mathbf{x}_0$. This method can be implemented as follows.

1) Selection of Most Promising Particle Streams: This step is reminiscent of the main idea of the APF, where resampling at time instant $t-1$ takes place by using measurements from time instant t. For CRPF, a risk function $R\left(\mathbf{x}_{t-1}^{(m)}/\mathbf{z}_t\right)$ is defined, which quantifies the quality of the particle $\mathbf{x}_{t-1}^{(m)}$ given the next set of observations, $\mathbf{z}_t$. The incremental cost can be used as a risk function which is given by

$$R\left(\mathbf{x}_{t-1}^{(m)}/\mathbf{z}_t\right) = \Delta C\left(\mathbf{x}_t^{(m)}/\mathbf{z}_t\right)$$

with

$$\hat{\mathbf{z}}_t^{(m)} = h\left(\hat{\mathbf{y}}_t^{(m)}\right) \tag{3.20}$$

$$\hat{\mathbf{y}}_t^{(m)} = g\left(\hat{\mathbf{x}}_t^{(m)}\right) + \boldsymbol{\mu}_v \tag{3.21}$$

$$\hat{\mathbf{x}}_t^{(m)} = G_x \hat{\mathbf{x}}_{t-1}^{(m)} \tag{3.22}$$

where the elements of $g(.)$ are defined by (3.4) and those of $h(.)$ by (3.5) and (3.6), i.e., the elements of $h(.)$ are given by

$$h\left(\hat{y}_{n,t}^{(m)}\right) = \begin{cases} \beta_n, & \hat{y}_{n,t} > \gamma \\ 0, & \hat{y}_{n,t} \le \gamma \end{cases}$$

Once the risks are computed, they are added to their costs at $t-1$ to obtain the predicted costs, i.e. $\hat{C}_t^{(m)} = \lambda C\left(\mathbf{x}_{0:t-1}^{(m)}/\mathbf{z}_{1:t-1}\right) + R\left(\mathbf{x}_{t-1}^{(m)}/\mathbf{z}_t\right)$ (3.23)

The particles are then sorted according to their predicted costs $\hat{C}_t^{(m)}$ in descending order and the first L of the M particles are replicated $J = M/L$ times (where J is an integer). In other words, each of the surviving particles will have J children at time instant t. With this sorting scheme, the usage of functions and classical resampling methods are avoided and proceeded directly with removing "bad" particles without sacrificing the performance.

2) Particle Propagation: For particle propagation a Gaussian proposal density can be used in a similar way as is done with APF, but without using the functional form of the Gaussian for computing the costs of the particles. Also, the use of a Gaussian is not strictly required, instead of that, any other density that is centered around the particle and that produces random variables with appropriate variance, like a uniform, or a Laplace, or a Cauchy density can be used. If the proposal density is Gaussian, then the velocities of the target can be drawn with mean $\dot{\mathbf{x}}_{t-1}^{(i_m)} = \begin{bmatrix} \dot{\mathbf{x}}_{1,t-1}^{(i_m)} & \dot{\mathbf{x}}_{2,t-1}^{(i_m)} \end{bmatrix}$ and with covariance matrix $\sigma_{t-1}^{2,(i_m)}\mathbf{I}_{2\times 2}$, where the i_m s denote indexes of sorted particles, and $\dot{\mathbf{x}}_{t-1}^{(i_m)}$'s are surviving particles from step 1.The variance, $\sigma_{t-1}^{2,(i_m)}$, is recursively updated by

$$\sigma_{t-1}^{2,(i_m)} = \frac{t-2}{t-1}\sigma_{t-2}^{2,(i_m)} + \frac{\| \dot{\mathbf{x}}_{t-1}^{(i_m)} - \dot{\mathbf{x}}_{t-2}^{(i_m)} \|^2}{2(t-1)}$$

The locations are then obtained by (3.15) and (3.16).

3) Cost Update: The cost update is performed by using (3.18).

4) State Estimation: The state is estimated by using the particles and the associated costs. One way of carrying out the estimation is by creating a pmf from the costs. This can be done by defining a monotonically decreasing function $\eta(.)$ that converts the set of costs into probability masses $\pi_{c_t}^{(m)}$, which is given by

$$\pi_{c_t}^{(m)} \propto \eta\left(C_t^{(m)}\right)$$

One function that has worked well for different problems is

$$\eta\left(C_t^{(m)}\right) = \frac{1}{\left(C_t^{(m)} - \min\left(C_t\right) + 1/M\right)^2}$$

Once the $\pi_{c_t}^{(m)}, m = 1,2,....,M$ are computed, the estimation of $\mathbf{x}_t$ can be carried out readily.

3.4 Posterior Cramer Rao Bound

In time invariant statistical models, a commonly used lower bound is the Cramer Rao Bound (CRB), given by the inverse of the Fisher information matrix. In the time varying systems we have considered, the estimated parameter vector is random and it corresponds to an underlying nonlinear, randomly driven model. Lower bounds for nonlinear dynamical systems have appeared in [22]. The continuous time case has received more emphasis as compared to discrete time case, which is of greater practical importance. In [21], a novel and simple derivation of the posterior CRB for the discrete time multidimensional non linear filtering problem that avoids any Gaussian assumptions is presented. In [15], this lower bound is extended for a frequently occurring case of nonlinear filtering, where the conditional distribution of the state one step ahead, given the current state, is singular but the implementation procedure for the computation of PCRB is not presented. So, we have done a detailed derivation of the implementation procedure for the computation of PCRB.

Let $\mathbf{x}$ represent a vector of measured data and $\boldsymbol{\theta}$ be an r-dimensional estimated random parameter, let $p_{x,\theta}(X,\Theta)$ be the joint probability density of the pair $(\mathbf{x},\boldsymbol{\theta})$ and let $g(\mathbf{x})$ be a function of $\mathbf{x}$ which is an estimate of $\boldsymbol{\theta}$. The PCRB on the estimation error has the form

$$P \triangleq E\left\{\left[g(\mathbf{x})-\boldsymbol{\theta}\right]\left[g(\mathbf{x})-\boldsymbol{\theta}\right]^{T}\right\} \geq J^{-1} \tag{3.23}$$

Where J is the $r \times r$ (Fisher) information matrix with the elements

$$J_{ij} = E\left[-\frac{\partial^2 \log p_{x,\theta}(X,\Theta)}{\partial\Theta_i \partial\Theta_j}\right] \qquad i,j = 1,\ldots\ldots,r \tag{3.24}$$

The inequality in (3.23) means that the difference $P - J^{-1}$ is a positive semi definite matrix. Let ∇ and Δ be operators of the first and second order partial derivatives respectively

$$\nabla_{\Theta} = \left[\frac{\partial}{\partial\Theta_1},\ldots\ldots,\frac{\partial}{\partial\Theta_r}\right]^{T}$$

$$\Delta_{\psi}^{\theta} = \nabla_{\psi}\nabla_{\Theta}^{T}$$

Using this notation, (3.24) can be written as

$$J = E\left[-\Delta_{\theta}^{\theta} \log p_{x,\theta}(X,\Theta)\right] \tag{3.25}$$

Since $p_{x,\theta}(X,\Theta) = p_{x/\theta}(X/\Theta).p_{\theta}(\Theta)$, it can easily be seen that J can be decomposed into two additive parts:

$$J = J_D + J_P$$

where J_D represents the information obtained from the data, and J_P represents the a priori information.

$$J_D = E\left\{-\Delta_{\theta}^{\theta} \log p_{x/\theta}(X/\Theta)\right\} \qquad (r \times r)$$

$$J_P = E\left\{-\Delta_{\theta}^{\theta} \log p_{\theta}(\Theta)\right\} \qquad (r \times r)$$

3.4.1 LOWER BOUND FOR THE NONLINEAR FILTERING PROBLEM

Consider the nonlinear filtering problem [15]

$$\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, w_t) \tag{3.26}$$

$$\mathbf{z}_t = h_t(\mathbf{x}_t, v_t) \tag{3.27}$$

where

$\mathbf{x}_t$ is system state at time t;

$\mathbf{z}_t$ is the measurement process;

w_t and v_t are independent white noise processes.

f_t and h_t are nonlinear functions.

The joint probability distribution of $\mathbf{x}_{0:t} = (\mathbf{x}_0, ..., \mathbf{x}_t)$ and $\mathbf{z}_{1:t} = (\mathbf{z}_1, ..., \mathbf{z}_t)$ which is denoted by p_t is given by

$$p_t = p(\mathbf{x}_{0:t}, \mathbf{z}_{1:t}) = p(\mathbf{x}_0)\prod_{j=1}^{t} p(\mathbf{z}_j/\mathbf{x}_j)\prod_{k=1}^{t} p(\mathbf{x}_k/\mathbf{x}_{k-1}) \tag{3.28}$$

Decompose $\mathbf{x}_{0:t}$ as $\mathbf{x}_{0:t} = \left[\mathbf{x}_{0:t-1}^T, \mathbf{x}_t^T\right]^T$ and correspondingly the information matrix $J(\mathbf{x}_{0:t})$ can be decomposed into blocks as

$$J(\mathbf{x}_{0:t}) = \begin{bmatrix} A_t & B_t \\ B_t^T & C_t \end{bmatrix}$$

$$\triangleq \begin{bmatrix} E\left\{-\Delta_{\mathbf{x}_{0:t-1}}^{\mathbf{x}_{0:t-1}} \log p_t\right\} & E\left\{-\Delta_{\mathbf{x}_{0:t-1}}^{\mathbf{x}_t} \log p_t\right\} \\ E\left\{-\Delta_{\mathbf{x}_t}^{\mathbf{x}_{0:t-1}} \log p_t\right\} & E\left\{-\Delta_{\mathbf{x}_t}^{\mathbf{x}_t} \log p_t\right\} \end{bmatrix} \tag{3.29}$$

it can be easily shown that the mean square error of estimation of $\mathbf{x}_t$ is lower bounded by the right lower block of $J(\mathbf{x}_{0:t})^{-1}$ and is given by J_t^{-1} .

where $J_t = C_t - B_t^T A_t^{-1} B_t$ (3.30)

Thus computation of the $(r \times r)$ matrix J_t involves calculation of the inverse of the matrix. $J(\mathbf{x}_{0:t})$. Recursive relation for the computation of Fisher information matrix is necessary for reducing the complexity.

The joint probability distribution of $\mathbf{x}_{0:t+1} = (\mathbf{x}_0, \ldots\ldots, \mathbf{x}_{t+1})$ and $\mathbf{z}_{1:t+1} = (\mathbf{z}_1, \ldots\ldots, \mathbf{z}_{t+1})$ which is denoted by p_{t+1} is given by

$$p_{t+1} \triangleq p(\mathbf{x}_{0:t+1}, \mathbf{z}_{1:t+1})$$

$$= p(\mathbf{x}_{0:t}, \mathbf{z}_{1:t}).p(\mathbf{x}_{t+1}/\mathbf{x}_{0:t}, \mathbf{z}_{1:t}).p(\mathbf{z}_{t+1}/\mathbf{x}_{t+1}, \mathbf{x}_{0:t}, \mathbf{z}_{1:t})$$

$$= p_t.p(\mathbf{x}_{t+1}/\mathbf{x}_t).p(\mathbf{z}_{t+1}/\mathbf{x}_{t+1}) \tag{3.31}$$

Let

$$D_t^{11} = E\left\{-\Delta_{\mathbf{x}_t}^{\mathbf{x}_t} \log p(\mathbf{x}_{t+1}/\mathbf{x}_t)\right\}$$

$$D_t^{12} = E\left\{-\Delta_{\mathbf{x}_t}^{\mathbf{x}_{t+1}} \log p(\mathbf{x}_{t+1}/\mathbf{x}_t)\right\}$$

$$D_t^{21} = E\left\{-\Delta_{\mathbf{x}_{t+1}}^{\mathbf{x}_t} \log p(\mathbf{x}_{t+1}/\mathbf{x}_t)\right\} = \left[D_t^{12}\right]^T$$

$$D_t^{22} = E\left\{-\Delta_{\mathbf{x}_{t+1}}^{\mathbf{x}_{t+1}} \log p(\mathbf{x}_{t+1}/\mathbf{x}_t)\right\} + E\left\{-\Delta_{\mathbf{x}_{t+1}}^{\mathbf{x}_{t+1}} \log p(\mathbf{z}_{t+1}/\mathbf{x}_{t+1})\right\}$$

Using (3.31) and the above equations the posterior information matrix for $\mathbf{x}_{0:t+1}$ can be written in block form as

$$J(\mathbf{x}_{0:t+1}) = \begin{bmatrix} A_t & B_t & 0 \\ B_t^T & C_t + D_t^{11} & D_t^{12} \\ 0 & D_t^{21} & D_t^{22} \end{bmatrix}$$

The information submatrix J_{t+1} can be found as an inverse of the right lower submatrix of $J(\mathbf{x}_{0:t+1})^{-1}$

$$J_{t+1} = D_t^{22} - \begin{bmatrix} 0 & D_t^{21} \end{bmatrix} \begin{bmatrix} A_t & B_t \\ B_t^T & C_t + D_t^{11} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ D_t^{12} \end{bmatrix}$$

$$= D_t^{22} - D_t^{21} \left[C_t + D_t^{11} - B_t^T A_t^{-1} B_t \right]^{-1} D_t^{12}$$

Using equation (3.30) the above equation can be rewritten as

$$J_{t+1} = D_t^{22} - D_t^{21} \left[J_t + D_t^{11} \right]^{-1} D_t^{12} \tag{3.32}$$

The above equation is the required recursive relation for the computation of information matrix and its inverse will give the lower bound.

The initial information submatrix J_0 can be calculated from the a priori probability function $p(x_0)$ and is given by

$$J_0 = E\left\{ -\Delta_{x_0}^{x_0} \log p(x_0) \right\} \tag{3.33}$$

Computation of the information submatrix, as described above will fails if the conditional distribution of $\mathbf{x}_{t+1}$ given $\mathbf{x}_t$ is singular. The state equation given by (3.3), which is used for the generation of the target movement, is singular so the above method cannot be used for computing the PCRB in this case. We next describe the method which can be used to find PCRB in the above situation.

3.4.2 LOWER BOUND FOR THE NONLINEAR FILTERING PROBLEM WHEN THE STATE EQUATION IS SINGULAR:

Let $\xi_t = \left[\mathbf{v}_t \quad \tilde{\mathbf{l}}_t\right]^T$ be the state vector

Where $\tilde{\mathbf{l}}_t = \left[x_{1,t} \quad x_{2,t} \quad \psi_t\right]^T$ and

$\mathbf{v}_t = \left[\dot{x}_{1,t} \quad \dot{x}_{2,t}\right]^T$

The state evolution as given by (3.3) can also be expressed in block vector notation [15] as

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \mathbf{F}u_t$$

$$\tilde{\mathbf{l}}_{t+1} = \tilde{\mathbf{l}}_t + \mathbf{G}\left(\mathbf{v}_{t+1} + \mathbf{v}_t\right) \tag{3.34}$$

Where

$$\mathbf{F} = \begin{bmatrix} T_s & 0 \\ 0 & T_s \end{bmatrix}, \qquad \mathbf{G} = \begin{bmatrix} \frac{T_s}{2} & 0 \\ 0 & \frac{T_s}{2} \\ 0 & 0 \end{bmatrix}$$

The derivation involves the following lemma [21].

Lemma 1: Consider the problem of estimating a random vector $\mathbf{x}$ from an observation vector $\mathbf{z}$. Let $p(\mathbf{x},\mathbf{z})$ be the joint probability density of $(\mathbf{x},\mathbf{z})$, and assume that information matrix $J(\mathbf{x}) = E\left\{-\Delta_{\mathbf{x}}^{\mathbf{x}} \log p(\mathbf{x},\mathbf{z})\right\}$ exists. Let, $\mathbf{y} = M\mathbf{x}$, where M is a constant invertible matrix. Then, the probability density $p(\mathbf{y},\mathbf{z})$ exists, and the corresponding information matrix for estimating $\mathbf{y}$ is given by

$$J(\mathbf{y}) = M^{-T} J(\mathbf{x}) M^{-1} \tag{3.35}$$

Let p_t denote the probability density of the triplet $\left[\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{z}_{1:t}\right]$ and is given by

$$p_t \triangleq p\left(\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{z}_{1:t}\right) = p\left(\mathbf{v}_{0:t-1}, \mathbf{v}_t, \mathbf{l}_t, \mathbf{z}_{1:t}\right) \tag{3.36}$$

The information matrix that corresponds to the triplet $\left[\mathbf{v}_{0:t-1}, \mathbf{v}_t, \mathbf{l}_t\right]$ can be written in block form as

$$J(\mathbf{v}_{0:t-1}, \mathbf{v}_t, \mathbf{l}_t) = \begin{bmatrix} A_t & B_t & C_t \\ B_t^T & D_t & E_t \\ C_t^T & E_t^T & F_t \end{bmatrix}$$

where the blocks $A_t, B_t, \ldots\ldots F_t$ are obtained as expectations of the second-order derivatives of $-\log p_t$ with respect $\mathbf{v}_{0:t-1}$, $\mathbf{v}_t$ and $\mathbf{l}_t$.

The information submatrix for the state vector ξ_t can be obtained as the inverse of the right-lower submatrix of $\left\{J(\mathbf{v}_{0:t-1}, \mathbf{v}_t, \mathbf{l}_t)^{-1}\right\}$, i.e.,

$$\begin{aligned} J_t &= \begin{bmatrix} J_t^{11} & J_t^{12} \\ J_t^{21} & J_t^{22} \end{bmatrix} \\ &= \begin{bmatrix} D_t & E_t \\ E_t^T & F_t \end{bmatrix} - \begin{bmatrix} B_t^T \\ C_t^T \end{bmatrix} [A_t]^{-1} [B_t \quad C_t] \\ &= \begin{bmatrix} D_t - B_t^T A_t^{-1} B_t & E_t - B_t^T A_t^{-1} C_t \\ E_t^T - C_t^T A_t^{-1} B_t & F_t - C_t^T A_t^{-1} C_t \end{bmatrix} \end{aligned} \tag{3.37}$$

The probability density of the quartet $[\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{v}_{t+1}, \mathbf{z}_{t+1}]$, denoted by p_{t+1} is given by

$$\begin{aligned} p_{t+1} &\triangleq p(\mathbf{v}_{0:t+1}, \mathbf{l}_t, \mathbf{z}_{1:t+1}) \\ &= p(\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{z}_{1:t}) . p(\mathbf{v}_{t+1}/\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{z}_{1:t}) . p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}, \mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{z}_{1:t}) \\ &= p_t . p(\mathbf{v}_{t+1}/\xi_t) . p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}, \xi_t) \end{aligned} \tag{3.38}$$

Applying log on both sides we have

$$-\log p_{t+1} = -\log p_t - \log\left(p(\mathbf{v}_{t+1}/\xi_t) . p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}, \xi_t)\right) = -\log p_t - \log \tilde{p}_t$$

where

$$\tilde{p}_t = p(\mathbf{v}_{t+1}/\xi_t) . p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}, \xi_t)$$

The information matrix that corresponds to the quartet $[\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{v}_{t+1}, \mathbf{z}_{t+1}]$ can be written in block form as

$$J(\mathbf{v}_{0:t}, \mathbf{l}_t, \mathbf{v}_{t+1}) = \begin{bmatrix} A_t & B_t & C_t & 0 \\ B_t^T & D_t + H_t^{11} & E_t + H_t^{12} & H_t^{13} \\ C_t^T & \left(E_t + H_t^{12}\right)^T & F_t + H_t^{22} & H_t^{23} \\ 0 & \left(H_t^{13}\right)^T & \left(H_t^{23}\right)^T & H_t^{33} \end{bmatrix}$$

where

$$\begin{aligned} H_t^{11} &= E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log \tilde{p}_t\right\} & H_t^{12} &= E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{l}_t} \log \tilde{p}_t\right\} & H_t^{13} &= E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_{t+1}} \log \tilde{p}_t\right\} \\ H_t^{22} &= E\left\{-\Delta_{\mathbf{l}_t}^{\mathbf{l}_t} \log \tilde{p}_t\right\} & H_t^{23} &= E\left\{-\Delta_{\mathbf{l}_t}^{\mathbf{v}_{t+1}} \log \tilde{p}_t\right\} & H_t^{33} &= E\left\{-\Delta_{\mathbf{v}_{t+1}}^{\mathbf{v}_{t+1}} \log \tilde{p}_t\right\} \end{aligned} \tag{3.39}$$

The information submatrix for $[\mathbf{v}_t, \mathbf{l}_t, \mathbf{v}_{t+1}]$ then equals

$$J(\mathbf{v}_t, \mathbf{l}_t, \mathbf{v}_{t+1}) = \begin{bmatrix} D_t + H_t^{11} & E_t + H_t^{12} & H_t^{13} \\ (E_t + H_t^{12})^T & F_t + H_t^{22} & H_t^{23} \\ (H_t^{13})^T & (H_t^{23})^T & H_t^{33} \end{bmatrix} - \begin{bmatrix} B_t^T \\ C_t^T \\ 0 \end{bmatrix} A_t^{-1} [B_t \quad C_t \quad 0]$$

This can be rewritten using equation (3.37) as

$$J(\mathbf{v}_t, \mathbf{l}_t, \mathbf{v}_{t+1}) = \begin{bmatrix} J_t^{11} + H_t^{11} & J_t^{12} + H_t^{12} & H_t^{13} \\ (J_t^{12} + H_t^{12})^T & J_t^{22} + H_t^{22} & H_t^{23} \\ (H_t^{13})^T & (H_t^{23})^T & H_t^{33} \end{bmatrix} \tag{3.40}$$

Now using equation (3.34) we can write

$$\begin{bmatrix} \mathbf{v}_t \\ \mathbf{v}_{t+1} \\ \mathbf{l}_{t+1} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} \\ \mathbf{G} & \mathbf{I} & \mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{v}_t \\ \mathbf{l}_t \\ \mathbf{v}_{t+1} \end{bmatrix} \triangleq M_t \begin{bmatrix} \mathbf{v}_t \\ \mathbf{l}_t \\ \mathbf{v}_{t+1} \end{bmatrix} \tag{3.41}$$

By using above lemma and equations (3.40) and (3.41) we can write,

$$J(\mathbf{v}_t, \mathbf{v}_{t+1}, \mathbf{l}_{t+1}) = M_t^{-T} J(\mathbf{v}_t, \mathbf{l}_t, \mathbf{v}_{t+1}) M_t^{-1}$$

$$= M_t^{-T} \begin{bmatrix} J_t^{11} + H_t^{11} & J_t^{12} + H_t^{12} & H_t^{13} \\ (J_t^{12} + H_t^{12})^T & J_t^{22} + H_t^{22} & H_t^{23} \\ (H_t^{13})^T & (H_t^{23})^T & H_t^{33} \end{bmatrix} M_t^{-1} \tag{3.42}$$

Let S_{t+1} be the information matrix which is given by

$$S_{t+1} = J(\mathbf{v}_t, \mathbf{v}_{t+1}, \mathbf{l}_{t+1})$$

this implies that

$$S_{t+1} \triangleq \begin{bmatrix} S_{t+1}^{11} & S_{t+1}^{12} & S_{t+1}^{13} \\ S_{t+1}^{21} & S_{t+1}^{22} & S_{t+1}^{23} \\ S_{t+1}^{31} & S_{t+1}^{32} & S_{t+1}^{33} \end{bmatrix}$$

$$= M_t^{-T} \begin{bmatrix} J_t^{11} + H_t^{11} & J_t^{12} + H_t^{12} & H_t^{13} \\ (J_t^{12} + H_t^{12})^T & J_t^{22} + H_t^{22} & H_t^{23} \\ (H_t^{13})^T & (H_t^{23})^T & H_t^{33} \end{bmatrix} M_t^{-1} \tag{3.43}$$

The information submatrix for $[\mathbf{v}_{t+1}, \mathbf{l}_{t+1}]$ then equals

$$J_{t+1} = \begin{bmatrix} J_{t+1}^{11} & J_{t+1}^{12} \\ J_{t+1}^{21} & J_{t+1}^{22} \end{bmatrix}$$

$$= \begin{bmatrix} S_{t+1}^{22} & S_{t+1}^{23} \\ S_{t+1}^{32} & S_{t+1}^{33} \end{bmatrix} - \begin{bmatrix} S_{t+1}^{21} \\ S_{t+1}^{31} \end{bmatrix} \left[S_{t+1}^{11} \right]^{-1} \begin{bmatrix} S_{t+1}^{12} & S_{t+1}^{13} \end{bmatrix} \tag{3.44}$$

The above equation is the recursive relation for estimating the information matrix from which we can easily find the Posterior Cramer Rao Lower Bound of the state vector by taking its inverse.

The obtained PCRBs do not have analytical expressions in closed form but they can be computed using Monte Carlo simulation methods.

3.4.3 IMPLEMENTATION PROCEDURE FOR THE ESTIMATION OF PCRB:

The variables which we have to compute for the estimation of PCRB are H_t^{11}, H_t^{12}, H_t^{13}, H_t^{22}, H_t^{23} and H_t^{33}.

Recalling the equations (3.39) we have

$$\tilde{p}_t = p(\mathbf{v}_{t+1}/\xi_t).p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t)$$

$$H_t^{11} = E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log \tilde{p}_t\right\} = E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p(\mathbf{v}_{t+1}/\mathbf{v}_t).p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t)\right)\right\}$$

$$= E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p(\mathbf{v}_{t+1}/\mathbf{v}_t)\right)\right\} + E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t)\right)\right\}$$

$$= H_{t,a}^{11} + H_{t,b}^{11} \tag{3.45}$$

From the state equation we can say that $p(\mathbf{v}_{t+1}/\mathbf{v}_t)$ is a Gaussian distribution with mean $\mathbf{v}_t$ and variance Q.

$$Q = E\left[(\mathbf{v}_{t+1} - \mathbf{v}_t)(\mathbf{v}_{t+1} - \mathbf{v}_t)^T\right] = E\left[(\mathbf{F}u_t)(\mathbf{F}u_t)^T\right]$$

$$= E\left[\mathbf{F}u_t u_t^T \mathbf{F}^T\right] = \mathbf{F}E\left[u_t u_t^T\right]\mathbf{F}^T = \mathbf{F}C_u\mathbf{F}^T$$

$$H_{t,a}^{11} = E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p(\mathbf{v}_{t+1}/\mathbf{v}_t)\right)\right\} = E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(k.\exp\left(-\frac{1}{2}(\mathbf{v}_{t+1} - \mathbf{v}_t)Q^{-1}(\mathbf{v}_{t+1} - \mathbf{v}_t)^T\right)\right)\right\}$$

$$= E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log(k)\right\} + E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t}\left(-\frac{1}{2}(\mathbf{v}_{t+1}-\mathbf{v}_t)Q^{-1}(\mathbf{v}_{t+1}-\mathbf{v}_t)^T\right)\right\} = Q^{-1}.$$

Therefore $H_{t,a}^{11} = Q^{-1} = \left(\mathbf{F}C_u\mathbf{F}^T\right)^{-1}$ (3.46)

$$H_{t,b}^{11} = E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

Note that the calculation of $H_{t,b}^{11}$ requires the exact knowledge of the observation likelihood function $p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)$ and for most of the real world scenarios including this problem,

$H_{t,b}^{11}$ does not have a closed form solution. However, similar to the nonlinear filtering problem, Monte Carlo techniques can again be applied to solve this problem.

In the above equations the expectation is taken over $p\left(\xi_{0:t+1},\mathbf{z}_{1:t+1}\right)$.

$$p\left(\xi_{0:t+1},\mathbf{z}_{1:t+1}\right) = p\left(\xi_{0:t},\mathbf{z}_{1:t}\right)p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right).p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)$$

$$H_{t,b}^{11} = E_{p(\xi_t)}\left\{\Lambda'\right\} \tag{3.47}$$

Where $\Lambda' \in R^5 \times R^5$ and its elements are defined as

$$\Lambda' = E_{p(\mathbf{v}_{t+1}/\mathbf{v}_t).p(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t)}\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \tag{3.48}$$

The outer integrations in (3.48) can be approximately evaluated by converting them into summations using Monte Carlo integration methodology. In order to do this a set of samples are generated from $\mathbf{v}_{t+1}^j = p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right)$ with identical weights $w_{t+1}^j = M^{-1}$

where $j = 1,\ldots\ldots,M$. Then, the above expectations can be approximated as follows.

$$\Lambda' \approx -\frac{1}{M}\sum_{j=1}^{M} E_{p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}^j,\xi_t\right)}\left\{\Delta_{\mathbf{v}_t}^{\mathbf{v}_t} \log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \tag{3.49}$$

The final expectations with respect to $p\left(\xi_t\right)$ in (3.47) can be obtained by averaging the above approximations over a number of Monte Carlo trials, i.e., over a number of sample tracks.

$$\Lambda' \approx -\frac{1}{M}\sum_{j=1}^{M} E_{p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}^j,\xi_t\right)}\left\{\nabla_{\mathbf{v}_t}\nabla_{\mathbf{v}_t}^T \log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$= -\frac{1}{M}\sum_{j=1}^{M} E_{p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)}\left\{\nabla_{\mathbf{v}_t}\left(\frac{\nabla_{\mathbf{v}_t}^{T} p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)}{p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)}\right)\right\}$$

$$= -\frac{1}{M}\sum_{j=1}^{M} E_{p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)}\left\{\frac{p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\nabla_{\mathbf{v}_t}\nabla_{\mathbf{v}_t}^{T} p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right) - \nabla_{\mathbf{v}_t} p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\nabla_{\mathbf{v}_t}^{T} p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)}{\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)^2}\right\}$$

In the above equation $\nabla_{\mathbf{v}_t}\nabla_{\mathbf{v}_t}^{T} p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right) = 0$, so the above equation can be modified as follows

$$\Lambda' \approx \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{N}\int_{0}^{\infty}\left\{\frac{\nabla_{\mathbf{v}_t} p\left(z_{i,t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)\nabla_{\mathbf{v}_t}^{T} p\left(z_{i,t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)}{p\left(z_{i,t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)}\right\} dz_{i,t+1} \tag{3.50}$$

The expressions for partial derivative terms in the above equation is given as

$$\begin{aligned}\nabla_{\mathbf{v}_t} p\left(z_{i,t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right) &= \sum_{s_{n,t+1}\in\{0,1\}} p\left(z_{n,t+1}/s_{n,t+1}\right)\nabla_{\mathbf{v}_t} p\left(s_{n,t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)\\ &= p\left(z_{n,t+1}/s_{n,t+1}=0\right)\nabla_{\mathbf{v}_t} p\left(s_{n,t+1}=0/\mathbf{v}_{t+1}^{j},\xi_t\right)\\ &\quad + p\left(z_{n,t+1}/s_{n,t+1}=1\right)\nabla_{\mathbf{v}_t} p\left(s_{n,t+1}=1/\mathbf{v}_{t+1}^{j},\xi_t\right)\end{aligned} \tag{3.51}$$

where

$$\nabla_{\mathbf{v}_t} p\left(s_{n,t+1}=1/\mathbf{v}_{t+1}^{j},\xi_t\right) = \nabla_{\mathbf{v}_t}\left[Q\left(\frac{\gamma - g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)\right]$$

$$= \begin{bmatrix} \frac{1}{\sqrt{2\pi}}\frac{\partial}{\partial \dot{x}_{1,t}}\int_{\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}}^{\infty} e^{-\frac{k^2}{2}}dk \\ \frac{1}{\sqrt{2\pi}}\frac{\partial}{\partial \dot{x}_{2,t}}\int_{\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}}^{\infty} e^{-\frac{k^2}{2}}dk \end{bmatrix} \tag{3.52}$$

In the above equation the partial differentiation can be done using Leibniz rule, which is given by

$$\frac{d}{d\alpha}\int_{a(\alpha)}^{b(\alpha)} f(x,\alpha)dx = \frac{db(\alpha)}{d\alpha} f\left(b(\alpha),\alpha\right) - \frac{da(\alpha)}{d\alpha} f\left(a(\alpha),\alpha\right) + \int_{a(\alpha)}^{b(\alpha)}\frac{\partial}{\partial\alpha} f(x,\alpha)dx \tag{3.53}$$

$$\frac{\partial}{\partial \dot{x}_{1,t}} \int_{\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}}^{\infty} e^{-\frac{k^2}{2}} dk = 0 - \frac{d}{d\dot{x}_{1,t}} \left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v} \right) e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} + 0$$

$$= -\frac{\alpha \psi d_0^{\alpha} \left(r_n - x_{1,t}\right)}{\sigma_v \left\| r_n - l_t \right\|^{\alpha+2}} e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} \tag{3.54}$$

Similarly,

$$\frac{\partial}{\partial \dot{x}_{2,t}} \int_{\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}}^{\infty} e^{-\frac{k^2}{2}} dk = -\frac{\alpha \psi d_0^{\alpha} \left(r_n - x_{2,t}\right)}{\sigma_v \left\| r_n - l_t \right\|^{\alpha+2}} e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} \tag{3.55}$$

Substituting the equations (3.54) and (3.55) in (3.52) we have,

$$\nabla_{\mathbf{v}_t} p\left(s_{n,t+1} = 1 / \mathbf{v}_{t+1}^j, \xi_t\right) = \begin{bmatrix} \frac{\alpha \psi d_0^{\alpha} \left(r_n - x_{1,t}\right)}{\sqrt{2\pi}\sigma_v \left\| r_n - l_t \right\|^{\alpha+2}} e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} \\ \frac{\alpha \psi d_0^{\alpha} \left(r_n - x_{2,t}\right)}{\sqrt{2\pi}\sigma_v \left\| r_n - l_t \right\|^{\alpha+2}} e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} \end{bmatrix} \tag{3.56}$$

Similarly,

$$\nabla_{\mathbf{v}_t} p\left(s_{n,t+1} = 0 / \mathbf{v}_{t+1}^j, \xi_t\right) = \begin{bmatrix} -\frac{\alpha \psi d_0^{\alpha} \left(r_n - x_{1,t}\right)}{\sqrt{2\pi}\sigma_v \left\| r_n - l_t \right\|^{\alpha+2}} e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} \\ -\frac{\alpha \psi d_0^{\alpha} \left(r_n - x_{2,t}\right)}{\sqrt{2\pi}\sigma_v \left\| r_n - l_t \right\|^{\alpha+2}} e^{-\frac{\left(\frac{\gamma - g_n\left(\mathbf{l}_t^j\right) - \mu_v}{\sigma_v}\right)^2}{2}} \end{bmatrix} \tag{3.57}$$

Substituting the equations (3.56) and (3.57) in (3.51) we have,

$$\nabla_{\mathbf{v}_t} p\left(z_{i,t+1}/\mathbf{v}_{t+1}^j,\xi_t\right)=\begin{bmatrix}\dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{1,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{-\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ \dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{2,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{-\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}\right)^2}{2}}\end{bmatrix}\left\{p\left(z_{n,t+1}/s_{n,t+1}=1\right)-p\left(z_{n,t+1}/s_{n,t+1}=0\right)\right\} \quad (3.58)$$

Similarly,

$$\nabla_{\mathbf{v}_t}^T p\left(z_{i,t+1}/\mathbf{v}_{t+1}^j,\xi_t\right)=\begin{bmatrix}\dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{1,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{-\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}\right)^2}{2}} & \dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{2,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{-\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}\right)^2}{2}}\end{bmatrix}\left\{p\left(z_{n,t+1}/s_{n,t+1}=1\right)-p\left(z_{n,t+1}/s_{n,t+1}=0\right)\right\}$$

(3.59)

By substituting the equations (3.58) and (3.59) in (3.50) we can get $H_{t,b}^{11}$.

$$H_t^{12}=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{l}_t}\log\tilde{p}_t\right\}=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{l}_t}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right).p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{l}_t}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right)\right)\right\}+E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{l}_t}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{l}_t}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \quad (3.60)$$

Again we have to apply Monte Carlo techniques for solving H_t^{12} . The only difference in calculating H_t^{12} is, the second partial derivative should be with respect to $\mathbf{l}_t$.

$$\nabla_{\mathbf{l}_t} p\left(s_{n,t+1}=1/\mathbf{v}_{t+1}^j,\xi_t\right)=\nabla_{\mathbf{l}_t}\left[Q\left(\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}\right)\right]$$

$$=\begin{bmatrix}\dfrac{1}{\sqrt{2\pi}}\dfrac{\partial}{\partial x_{1,t}}\displaystyle\int_{\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}}^{\infty}e^{-\frac{k^2}{2}}dk \\ \dfrac{1}{\sqrt{2\pi}}\dfrac{\partial}{\partial x_{2,t}}\displaystyle\int_{\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}}^{\infty}e^{-\frac{k^2}{2}}dk \\ \dfrac{1}{\sqrt{2\pi}}\dfrac{\partial}{\partial \psi}\displaystyle\int_{\frac{\gamma-g_n\left(\mathbf{l}_t^j\right)-\mu_v}{\sigma_v}}^{\infty}e^{-\frac{k^2}{2}}dk\end{bmatrix}$$

$$\nabla_{l_t} p\left(s_{n,t+1}=1/\mathbf{v}_{t+1}^{j},\xi_t\right)=\begin{bmatrix}\dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{1,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ \dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{2,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ \dfrac{d_0^{\alpha}}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}}\end{bmatrix} \tag{3.61}$$

$$\nabla_{l_t} p\left(s_{n,t+1}=0/\mathbf{v}_{t+1}^{j},\xi_t\right)=\begin{bmatrix}-\dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{1,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ -\dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{2,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ -\dfrac{d_0^{\alpha}}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}}\end{bmatrix} \tag{3.62}$$

By using equations (3.61) and (3.62) we have,

$$\nabla_{l_t} p\left(z_{i,t+1}/\mathbf{v}_{t+1}^{j},\xi_t\right)=\begin{bmatrix}\dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{1,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ \dfrac{\alpha\psi d_0^{\alpha}\left(r_n-x_{2,t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}} \\ \dfrac{d_0^{\alpha}}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha}}e^{\frac{\left(\frac{\gamma-g_n\left(\mathbf{l}_t^{j}\right)-\mu_v}{\sigma_v}\right)^2}{2}}\end{bmatrix}\left\{p\left(z_{n,t+1}/s_{n,t+1}=1\right)-p\left(z_{n,t+1}/s_{n,t+1}=0\right)\right\} \tag{3.63}$$

$$\nabla_{v_t}^{T} p\left(z_{t+1}/v_{t+1}^{i},\xi\right)=\left[\frac{\alpha\mu c_0^{\alpha}\left(r_n-x_{1t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{r-g_1(l_t')-\mu}{\sigma_v}\right)^2}{2}}\quad\frac{\alpha\mu c_0^{\alpha}\left(r_n-x_{2t}\right)}{\sqrt{2\pi}\sigma_v\left\|r_n-l_t\right\|^{\alpha+2}}e^{\frac{\left(\frac{r-g_2(l_t')-\mu}{\sigma_v}\right)^2}{2}}\right]\left\{p\left(z_{n,t+1}/s_{n,t+1}=1\right)-p\left(z_{n,t+1}/s_{n,t+1}=0\right)\right\} \quad (3.64)$$

again by substituting the equations (3.63) and (3.64) in (3.50) we can get H_t^{12}.

Similarly we can find the remaining variables H_t^{13}, H_t^{22}, H_t^{23} and H_t^{33} by using Monte Carlo methods.

$$H_t^{22}=E\left\{-\Delta_{l_t'}^{l_t'}\log\tilde{p}_t\right\}=E\left\{-\Delta_{l_t'}^{l_t'}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right).p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{l_t'}^{l_t'}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right)\right)\right\}+E\left\{-\Delta_{l_t'}^{l_t'}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{l_t'}^{l_t'}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \quad (3.65)$$

$$H_t^{13}=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_{t+1}}\log\tilde{p}_t\right\}=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right).p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right)\right)\right\}+E\left\{-\Delta_{\mathbf{v}_t}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \quad (3.66)$$

$$H_t^{23}=E\left\{-\Delta_{\mathbf{l}_t'}^{\mathbf{v}_{t+1}}\log\tilde{p}_t\right\}=E\left\{-\Delta_{\mathbf{l}_t'}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right).p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{\mathbf{l}_t'}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right)\right)\right\}+E\left\{-\Delta_{\mathbf{l}_t'}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{\mathbf{l}_t'}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \quad (3.67)$$

$$H_t^{33}=E\left\{-\Delta_{\mathbf{v}_{t+1}}^{\mathbf{v}_{t+1}}\log\tilde{p}_t\right\}=E\left\{-\Delta_{\mathbf{v}_{t+1}}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right).p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\}$$

$$=E\left\{-\Delta_{\mathbf{v}_{t+1}}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{v}_{t+1}/\mathbf{v}_t\right)\right)\right\}+E\left\{-\Delta_{\mathbf{v}_{t+1}}^{\mathbf{v}_{t+1}}\log\left(p\left(\mathbf{z}_{t+1}/\mathbf{v}_{t+1},\xi_t\right)\right)\right\} \quad (3.68)$$

3.5 SIMULATION RESULTS:

For target tracking in binary sensor networks using particle filtering algorithm in MATLAB environment, the following parameters have been used in simulations.

- Number of sensors in the network N =264
- The sensors are deployed in a field with dimensions $800\times 500\, m^2$.
- The attenuation parameter α =2.5
- Reference power parameter ψ =5000 at reference distance d_0 =1 m.
- The sensing radius of the sensor is 25 m.
- The threshold power γ =2.
- Covariance matrix of the state noise process is $\mathbf{c}_u = diag(0.05, 0.01)$.
- Mean and variance of the measurement noise in (2.4) is μ_v =1 & σ_v^2 =0.01.
- The observation noise variance at the fusion center is σ_ε^2 =0.01.
- Sampling interval T_s =1 sec.
- Number of particles M =1000.
- The prior for the target's location and velocity is a Gaussian distribution with mean $\bar{\mathbf{x}}_0 = [0 \quad 0 \quad 0.01 \quad 0.01]^T$ and covariance matrix Ξ =diag$\{10, 10, 0.1, 0.1\}$.
- Initial particles of ψ were drawn from a uniform distribution on $[10^3, 10^4]$.
- The cost function of the CRPF is defined using (3.18) and (3.19) with forgetting factor λ =0 and q =2.

To obtain the performance of state estimation, the Root Mean Square Error (RMSE) between the true state and estimated state is computed, which is given by

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}-\hat{\mathbf{x}})^2} \tag{3.69}$$

The noise variance σ_ε^2 and the parameters β_n, unless otherwise stated, were chosen to yield signal-to-noise ratio (SNR) of 20 dB at the fusion center. For simulation, the deterministically deployed sensor network is considered.

Steps carried out for the simulation of auxiliary particle filtering algorithm for target tracking in WSN are:

1) Generate the target trajectory using the state equation (3.3) and the prior information.
2) Generate sensor measurements using the measurement equation (3.4) and compare the measurements with the threshold using equation (3.6), to send binary information to the fusion center indicating the presence or absence of the target.
3) Repeat step-2 for each and every sensor and at each time instant t.
4) Generate the observations at the fusion center using equation (3.5) for each and every sensor and at each time instant t.
5) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.
6) Compute the characterizing parameter using equation (3.14) and compute the weights using equation (3.7). Now based on these weights select the most promising particle streams.
7) Generate the new particles using equations (3.15) and (3.16).
8) Calculate the weights of new generated particle streams and then estimate the target state using equation (3.17)
9) Repeat steps from 6 to 8 for each and every time instant t.
10) Compute the RMSE between the true state and the estimated state using equation (3.69).

Steps from 1 to 9 are repeated for each independent trial and RMSE is averaged over all independent trials.

Steps carried out for the simulation of cost reference particle filtering algorithm for target tracking in WSN are:

1) Generate the target trajectory using the state equation (3.3) and the prior information.
2) Generate sensor measurements using the measurement equation (3.4) and compare the measurements with the threshold using equation (3.6), to send binary information to the fusion center indicating the presence or absence of the target.
3) Repeat step-2 for each and every sensor and at each time instant t.
4) Generate the observations at the fusion center using equation (3.5) for each and every sensor and at each time instant t.
5) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.

3) Estimate the present state samples based on previous state samples using equation (3.22).
4) Substitute the present state samples in equation (3.21) for getting the measurement information at the sensors.
5) Take the decision at the sensors using equation (3.6) and estimate the observation information at the fusion center using equation (3.20).
6) Compute the risk function using equations (3.19) and compute the costs using equation (3.18). Now based on these costs select the most promising particle streams and directly remove the bad particles.
7) Generate the new particles using equations (3.15) and (3.16).
8) Calculate the costs of new generated particle streams using equation (3.18) and then estimate the target state using equation (3.17)
9) Repeat steps from 6 to 8 for each and every time instant t.
10) Compute the RMSE between the true state and the estimated state using equation (3.69).

Steps from 1 to 9 are repeated for each independent trial and RMSE is averaged over all independent trials.

Steps carried out for the simulation of Posterior Cramer-Rao lower bound for target tracking in WSN are:

1) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.
2) Compute the equations (3.58) and (3.59) and substitute in equation (3.50).
3) Now compute the integration term in equation (3.50) using Simpsons rule for the estimation of parameter $H_{t,b}^{11}$.
4) Similarly compute the parameter $H_{t,a}^{11}$ using equation (3.46).
5) Now substitute both $H_{t,a}^{11}$ and $H_{t,b}^{11}$ in equation (3.45) for computing the parameter H_t^{11}.
6) Similarly find the remaining variables H_t^{12}, H_t^{13}, H_t^{22}, H_t^{23} and H_t^{33} by using equations (3.65), (3.66), (3.67) and (3.68).
7) Calculate the information matrix using equations (3.43) and (3.44).

8) The diagonal elements of inverse of the information matrix will give the RMSEs of position, velocity and the target emitted power.

Steps from 1 to 8 are repeated for each independent trial and RMSE is averaged over all independent trials.

In Fig. 3.2, we can see the realization of a trajectory for the target and the obtained estimates using APF and CRPF, denoted by APF-bin, and CRPF-Bin, respectively. It can be seen that the algorithms track the target's trajectory closely.

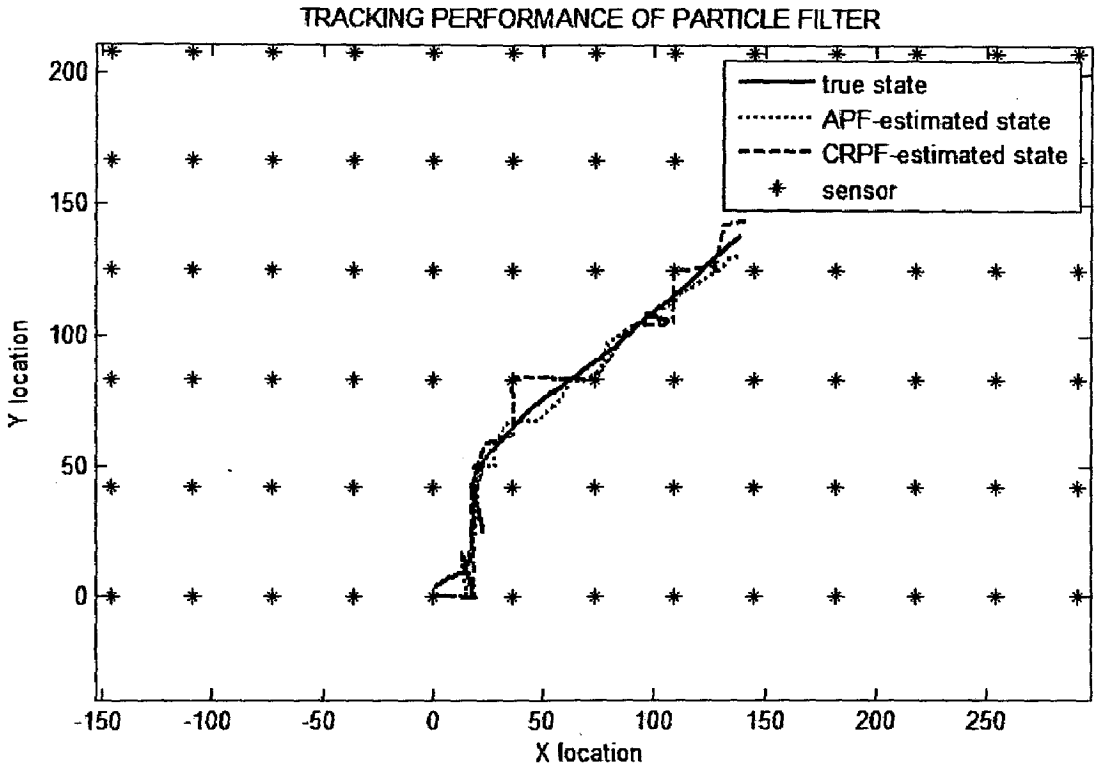


Fig3.2. A realization of a target trajectory and its estimates by APF and CRPF for deterministically deployed sensor network.

In Fig. 3.3, we display the root mean square errors (RMSEs) of the location estimate of the target obtained by the APF and CRPF algorithms that use binary sensor measurements, denoted by APF-bin, and CRPF-Bin, respectively. From the fig.3.3 we can observe that the CRPF does not have much degraded performance with respect to the APF even though it does not use probabilistic information and we can also observe that the RMSEs of APF and CRPF are very close to PCRB. The RMSEs were obtained by averaging over 100 different realizations.

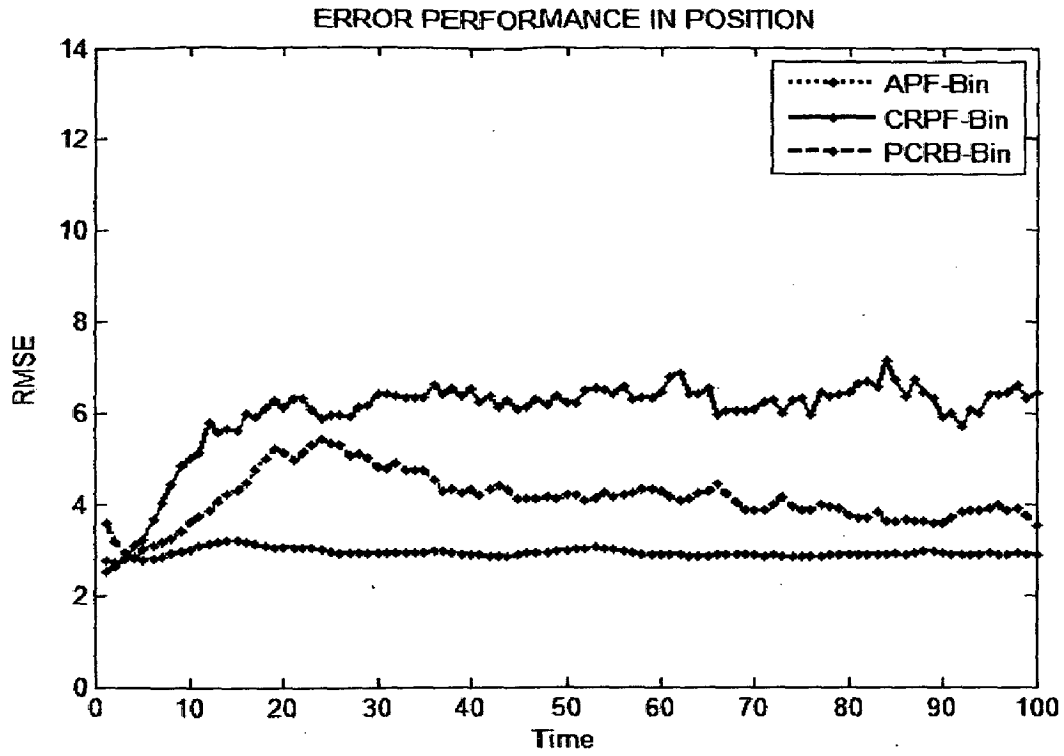


Fig3.3 RMSEs of the location estimates of the target obtained by the APF and CRPF algorithms with binary measurements and its PCRB.

In Fig. 3.4, the RMSEs of the constant parameter ψ are displayed for both APF and CRPF algorithms and its PCRB is also presented.

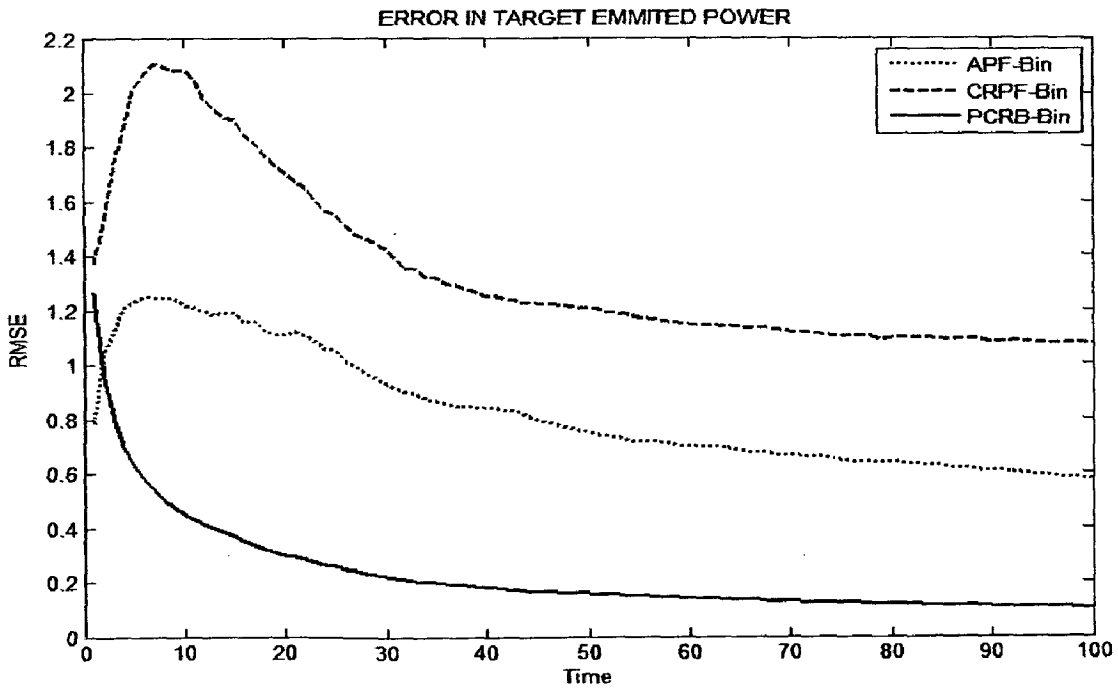


Fig3.4. RMSEs of ψ as a function of time

In Fig. 3.5, we display the root mean square errors (RMSEs) of the location estimate of the target obtained by the APF and CRPF algorithms by assuming the target emitted power as a known constant and as an unknown constant. From this figure, we can observe that there is no significant degradation in performance even though the emitted power by the target is assumed to be unknown. Again the RMSEs were obtained by averaging over 100 different realizations.

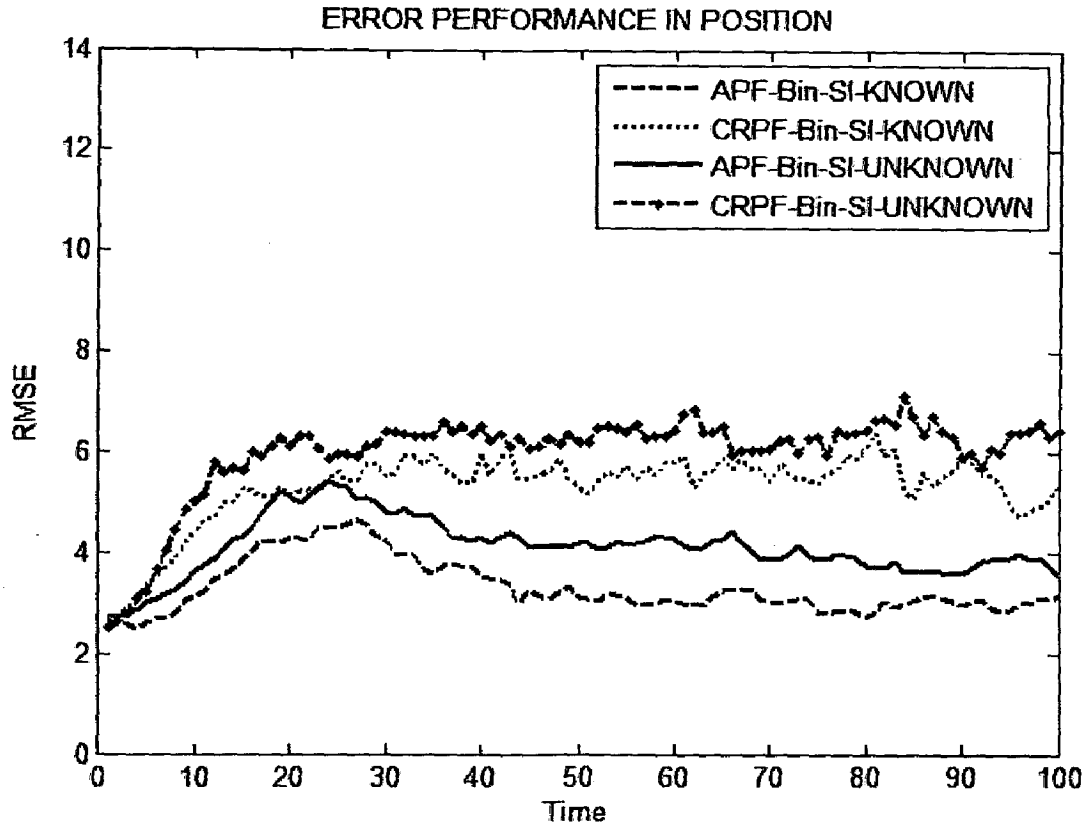


Fig3.5. RMSEs of location estimates of the target obtained by the APF and CRPF algorithms by assuming the target emitted power as a known constant and as an unknown constant.

We also performed the simulations by assuming the mixture noise processes. The state noise process is generated using a Gaussian mixture model and is given by

$$\mathbf{u}_t \sim 0.6N\left(0,\mathbf{C}_{u,1}\right)+0.4N\left(0,\mathbf{C}_{u,2}\right)$$

with $\mathbf{C}_{u,1}=diag\{0.05,0.02\}$ and $\mathbf{C}_{u,2}=diag\{0.5,0.2\}$.

The transmission measurement noise using a Gaussian mixture model is as follows

$$\varepsilon_{n,t} \sim 0.5N\left(\mu_\varepsilon,\sigma_\varepsilon^2\right)+0.5N\left(-\mu_\varepsilon,\sigma_\varepsilon^2\right)$$

with $\mu_\varepsilon=0.1$ and $\sigma_\varepsilon^2=0.001$

Instead of the accurate pdf, the APF assumes a Gaussian pdf $\mathbf{u}_t \sim N\left(0, \tilde{\mathbf{C}}_u\right)$ for generating new particles. Similarly APF assumed the measurement noise to be zero mean and with a variance of $\tilde{\sigma}_\varepsilon^2 = \mu_\varepsilon^2 + \sigma_\varepsilon^2 = 0.0081$.

In Fig. 3.6, we plot the cumulative distribution functions of the RMSE of APF-Bin and CRPF-Bin. From the graph we can say that 85% of the times the RMSE accumulated by the CRPF-Bin is less than 12m while 53% of the times the RMSE accumulated by the APF-bin is less than 12m. Clearly CRPF-Bin outperforms the APF-Bin considerably in the presence of mixture noise. The RMSEs of 100 different trajectories were computed and summed over the entire time period.

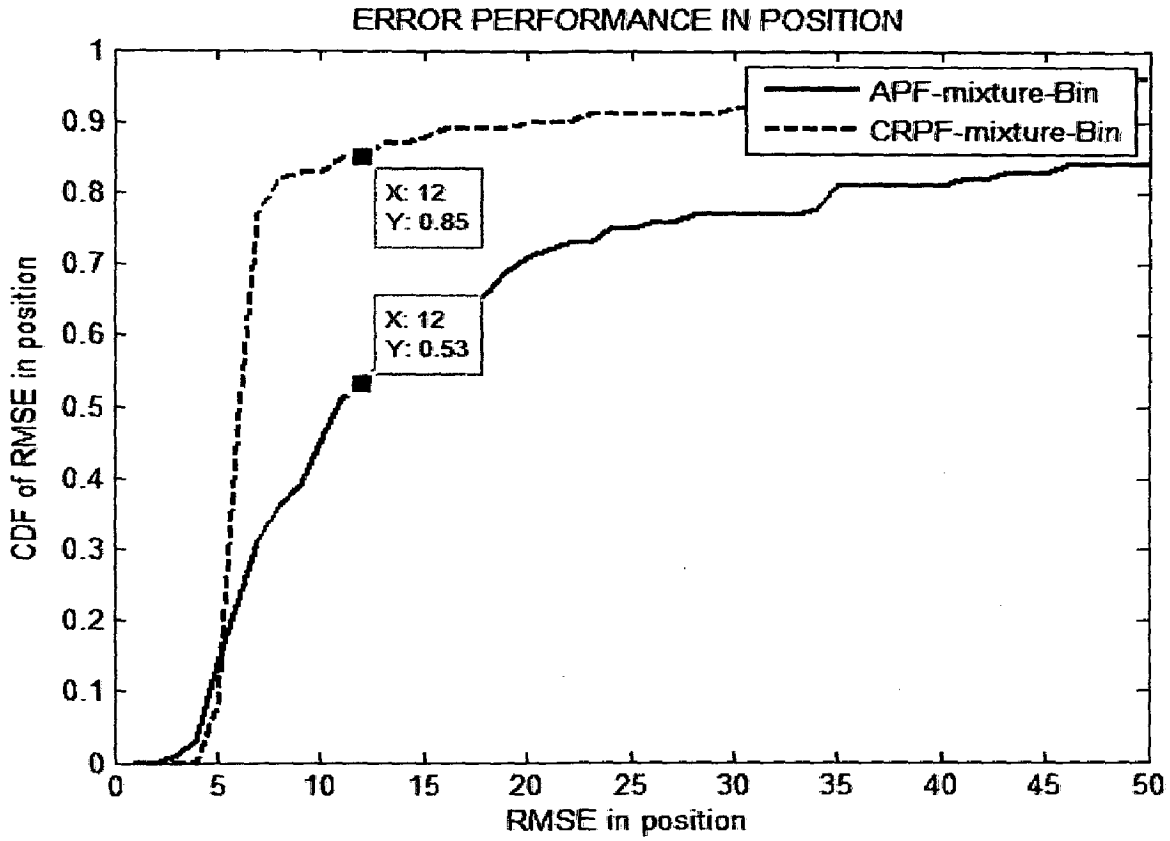


Fig 3.6 CDF of the RMSEs of APF-Bin and CRPF-Bin.

Chapter 4

TRACKING IN BINARY SENSOR NETWORKS USING CHANNEL AWARE PARTICLE FILTERING

In this chapter, particle filtering approach for target tracking in binary sensor network (BSN) is described, by considering the imperfect nature of the wireless communication channels between sensors and the fusion center. In a target tracking scenario where a large number of wireless sensors are deployed in a particular area, we cannot always guarantee a line of sight between sensors and the fusion center [16]. The signals that reach the fusion center of these networks will face challenging problems for recovering the sensed information by the sensors. The imperfect nature of the wireless communication channel between sensors and the fusion center is incorporated in the particle filter tracking algorithm known as channel aware particle filtering, which is an efficient tracking algorithm for recovering the target information based on the received information at the fusion center. The imperfect nature of the wireless communication channel between sensors and the fusion center is modelled as Rayleigh fading channel [26]. We consider phase coherent and phase noncoherent reception. Two particle filtering algorithms namely, auxiliary particle filtering (APF) and cost reference particle filtering (CRPF) are presented by considering the Rayleigh fading channel with coherent and noncoherent reception. Finally the simulation results of APF and CRPF are presented.

4.1 Mathematical formulation of tracking problem

The state and measurement equations as defined in previous chapter using equations (3.3) and (3.4) will remain same, whereas the information received at the fusion center which is defined by equation (3.5) will change and it involves the imperfect wireless channel between the sensor and fusion center. The modified observation equation [16] is given by

$$r_{n,t} = h_{n,t} e^{j\phi_{n,t}} s_{n,t} + \varepsilon_{n,t} \tag{4.1}$$

where

$$s_{n,t} = \begin{cases} 1 & if \ \ y_{n,t} > \gamma \\ 0 & if \ \ y_{n,t} \le \gamma \end{cases}$$

$\varepsilon_{n,t}$ is a zero mean complex Gaussian observation noise ($\mathcal{CN}$) with independent real and

imaginary parts having identical variance σ_ε^2. i.e., $\varepsilon_{n,t} \sim \mathcal{CN}\left(0, 2\sigma_\varepsilon^2\right)$

$h_{n,t}e^{j\phi_{n,t}}$ denote the complex gain of the discrete time Rayleigh fading channel between n^{th} sensor and the fusion center and it is assumed that the channel gain is stationary and ergodic.

Now the objective is to track the evolving state $\mathbf{x}_{0:t} = (\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_t)$ using the observations $\mathbf{r}_{1:t} = (r_{1,1:t}, ...r_{N,1:t})$, that is, the observations up to time instant t of the first sensor, $r_{1,1:t}$, the second sensor, $r_{2,1:t}$, as well as the remaining $N-2$ sensors, $r_{3,1:t}, ...r_{N,1:t}$. The observations are obtained by using equation (4.1). Therefore by the use of observations $\mathbf{r}_{1:t}$, the state vector $\mathbf{x}_{0:t}$ can be estimated by the particle filter.

4.2 APF algorithm with coherent reception

In coherent reception [25], the binary signalling is replaced by NRZ signalling so that the effect of the fading channel reduces to a real scalar multiplication [23]. i.e., (0, 1) is replaced by (-1, 1). The received measurement $\tilde{s}_{n,t}$ with the knowledge of the channel phase at the receiver can be expressed as

$$\tilde{s}_{n,t} = \mathrm{Re}\left\{r_{n,t}e^{-j\phi_{n,t}}\right\} = h_{n,t}s_{n,t} + \mathrm{Re}\left\{\varepsilon_{n,t}e^{-j\phi_{n,t}}\right\} \tag{4.2}$$

The noise term in the above equation is real WGN with variance σ_ε^2,

$$\text{i.e., } \mathrm{Re}\left\{\varepsilon_{n,t}e^{-j\phi_{n,t}}\right\} \sim \mathcal{N}\left(0, \sigma_\varepsilon^2\right) \tag{4.3}$$

After multiplying with the phase information, using equations (4.2) and (4.3) the observation equation can be rewritten as

$$z_{n,t} = h_{n,t}s_{n,t} + \mathcal{N}\left(0, \sigma_\varepsilon^2\right) \tag{4.4}$$

Let us assume that the channel has unit power, then the probability density function of $h_{n,t}$ is given as

$$p\left(h_{n,t}\right) = 2h_{n,t}e^{-h_{n,t}^2}, \qquad h_{n,t} \geq 0 \tag{4.5}$$

The APF algorithm which is used for estimating the target state as described in the previous chapter will remains same except in evaluating the pdf $p\left(z_{n,t}/\mathbf{x}_t^{(m)}\right)$.

Recalling the equation (3.9) we have

$$p\left(z_{n,t}/\mathbf{x}_t^{(m)}\right)=p\left(z_{n,t}/s_{n,t}=-1\right)P\left(s_{n,t}=-1/\mathbf{x}_t^{(m)}\right)$$
$$+p\left(z_{n,t}/s_{n,t}=1\right)P\left(s_{n,t}=1/\mathbf{x}_t^{(m)}\right) \tag{4.6}$$

where

$$P\left(s_{n,t}=1/\mu_t^{(m)}\right)=Q\left(\frac{\gamma-g_n\left(\mu_t^{(m)}\right)-\mu_v}{\sigma_v}\right) \tag{4.7}$$

$$P\left(s_{n,t}=-1/\mu_t^{(m)}\right)=1-Q\left(\frac{\gamma-g_n\left(\mu_t^{(m)}\right)-\mu_v}{\sigma_v}\right) \tag{4.8}$$

where $Q(.)$ denotes the complement of the standard normal cumulative distribution function. For evaluating the above equation (4.6) we have to compute the conditional pdf $p\left(z_{n,t}/s_{n,t}\right)$, which is given by

$$p\left(z_{n,t}/s_{n,t}\right)=\int_0^\infty p\left(z_{n,t}/h_{n,t},s_{n,t}\right)p\left(h_{n,t}\right)dh_{n,t} \tag{4.9}$$

Substituting the equation (4.5) in the above equation (4.9) the conditional pdf $p\left(z_{n,t}/s_{n,t}\right)$ in equation (4.9) simplifies to [24]

$$p\left(z_{n,t}/s_{n,t}\right)=\frac{2\sigma_\varepsilon}{\sqrt{2\pi}\left(1+2\sigma_\varepsilon^2\right)}e^{-\frac{z_{n,t}^2}{2\sigma_\varepsilon^2}}\left[1+s_{n,t}\sqrt{2\pi}\alpha z_{n,t}e^{\frac{\alpha z_{n,t}^2}{2}}Q\left(-\alpha z_{n,t}s_{n,t}\right)\right] \tag{4.10}$$

where $\alpha=1/\left(\sigma_\varepsilon\sqrt{1+2\sigma_\varepsilon^2}\right)$. using equation (4.10) we have

$$p\left(z_{n,t}/s_{n,t}=1\right)=\frac{2\sigma_\varepsilon}{\sqrt{2\pi}\left(1+2\sigma_\varepsilon^2\right)}e^{-\frac{z_{n,t}^2}{2\sigma_\varepsilon^2}}\left[1+\sqrt{2\pi}\alpha z_{n,t}e^{\frac{\alpha z_{n,t}^2}{2}}Q\left(-\alpha z_{n,t}\right)\right]$$

$$p\left(z_{n,t}/s_{n,t}=-1\right)=\frac{2\sigma_\varepsilon}{\sqrt{2\pi}\left(1+2\sigma_\varepsilon^2\right)}e^{-\frac{z_{n,t}^2}{2\sigma_\varepsilon^2}}\left[1-\sqrt{2\pi}\alpha z_{n,t}e^{\frac{\alpha z_{n,t}^2}{2}}Q\left(\alpha z_{n,t}\right)\right]$$

So finally by substituting the above equations in equation (4.6) and then following the same procedure as in APF we can estimate the target state.

4.3 APF algorithm with noncoherent reception

In noncoherent reception [25], we employ the energy detection (ED) strategy at the fusion center and makes it possible to use sensor censoring, i.e., ON/OFF signalling at the sensors [24]. The sensor censoring enables WSN to save energy thereby increasing the network lifetime. The received signal from the n^{th} sensor before ED at the fusion center is given by

$$r_{n,t} = \begin{cases} \varepsilon_{n,t} & s_{n,t} = 0 \\ h_{n,t} e^{j\phi_{n,t}} + \varepsilon_{n,t}, & s_{n,t} = 1 \end{cases} \quad (4.11)$$

where $\varepsilon_{n,t} \sim \mathcal{CN}\left(0, 2\sigma_\varepsilon^2\right)$, and $h_{n,t} e^{j\phi_{n,t}} \sim \mathcal{CN}(0,1)$

After energy detection, the observation model at the fusion center for the nth sensor is given as

$$z_{n,t} = \left| r_{n,t} \right|^2 \quad (4.12)$$

where the notation $|\cdot|$ indicates the magnitude of a complex number.

The APF algorithm which is used for estimating the target state as described in the previous chapter will remains same except in evaluating the pdf $p\left(z_{n,t} / \mathbf{x}_t^{(m)}\right)$.

Recalling the equation (3.9) we have

$$p\left(z_{n,t} / \mathbf{x}_t^{(m)}\right) = p\left(z_{n,t} / s_{n,t} = 0\right) P\left(s_{n,t} = 0 / \mathbf{x}_t^{(m)}\right) + p\left(z_{n,t} / s_{n,t} = 1\right) P\left(s_{n,t} = 1 / \mathbf{x}_t^{(m)}\right) \quad (4.13)$$

where

$$P\left(s_{n,t} = 1 / \mu_t^{(m)}\right) = Q\left(\frac{\gamma - g_n\left(\mu_t^{(m)}\right) - \mu_v}{\sigma_v}\right)$$

$$P\left(s_{n,t} = 0 / \mu_t^{(m)}\right) = 1 - Q\left(\frac{\gamma - g_n\left(\mu_t^{(m)}\right) - \mu_v}{\sigma_v}\right)$$

where $Q(.)$ denotes the complement of the standard normal cumulative distribution function.

Again for evaluating the equation (4.13) we have to compute the conditional pdf $p\left(z_{n,t} / s_{n,t}\right)$.

Since $|r_{n,t}|$ is a Rayleigh distributed random variable, $\sum_{i=1}^{k}|r_{n,t}|^2$ has a gamma distribution with parameters k and σ^2. From equation (4.12) we can say that $z_{n,t}$ is gamma distributed with shaping parameter k as 1 and scaling parameter σ^2 as the variance of $|r_{n,t}|$.

The equation defining the probability density function of a gamma-distributed random variable $z_{n,t}$ which is $p(z_{n,t}/s_{n,t})$, can be written as

$$p(z_{n,t}/s_{n,t}=0)=\frac{1}{2\sigma_\varepsilon^2}e^{-\frac{z_{n,t}}{2\sigma_\varepsilon^2}}, \qquad z_{n,t}>0 \tag{4.14}$$

$$p(z_{n,t}/s_{n,t}=1)=\frac{1}{1+2\sigma_\varepsilon^2}e^{-\frac{z_{n,t}}{1+2\sigma_\varepsilon^2}}, \qquad z_{n,t}>0 \tag{4.15}$$

So finally by substituting the above equations (4.14) and (4.15) in equation (4.13) and then following the same procedure as in APF we can estimate the target state.

4.4 CRPF algorithm with coherent reception

In coherent reception the observation model given in equation (4.4) will remains same.

Recalling the observation equation (4.4) we have

$$z_{n,t}=h_{n,t}s_{n,t}+\mathcal{N}(0,\sigma_\varepsilon^2) \tag{4.16}$$

where

$$s_{n,t}=\begin{cases}1 & if\ y_{n,t}>\gamma\\ -1 & if\ y_{n,t}\le\gamma\end{cases} \tag{4.17}$$

From this observation information we have to estimate the target state. The CRPF algorithm which is used for estimating the target state as described in the previous chapter will remains same except in estimating the observation information $\hat{\mathbf{z}}_t^{(m)}$ which is used for calculating the cost function.

Recalling the equations (3.20), (3.21) and (3.22) we have

$$\hat{\mathbf{z}}_t^{(m)}=h(\hat{\mathbf{y}}_t^{(m)}) \tag{4.18}$$

$$\hat{\mathbf{y}}_t^{(m)}=g(\hat{\mathbf{x}}_t^{(m)})+\mu_v \tag{4.19}$$

$$\hat{\mathbf{x}}_t^{(m)}=G_x\hat{\mathbf{x}}_{t-1}^{(m)} \tag{4.20}$$

where the elements of $g(.)$ are defined by (3.4) and those of $h(.)$ by (4.16) and (4.17), i.e., the elements of $h(.)$ are given by

$$h\left(\hat{y}_{n,t}^{(m)}\right)=\begin{cases} E\left(h_{n,t}\right), & \hat{y}_{n,t}>\gamma \\ -E\left(h_{n,t}\right), & \hat{y}_{n,t}\le\gamma \end{cases} \tag{4.21}$$

Let us assume that the channel has unit power, then the probability density function of $h_{n,t}$ is given as

$$p\left(h_{n,t}\right)=2h_{n,t}e^{-h_{n,t}^2}, \qquad h_{n,t}\ge 0 \tag{4.22}$$

we know that

$$E\left(h_{n,t}\right)=\int_0^\infty h_{n,t}p\left(h_{n,t}\right)dh_{n,t} \tag{4.23}$$

By substituting the equation (4.22) in equation (4.23) we have

$$E\left(h_{n,t}\right)=\int_0^\infty 2h_{n,t}^2e^{-h_{n,t}^2}dh_{n,t} \tag{4.24}$$

Let $h_{n,t}^2=x$ then the above equation (4.24) can be rewritten as

$$E\left(h_{n,t}\right)=\int_0^\infty \sqrt{x}e^{-x}dx \tag{4.25}$$

The Gamma function is defined as

$$\Gamma(n)=\int_0^\infty t^{n-1}e^{-t}dt \tag{4.26}$$

Comparing equations (4.25) and (4.26) we can say, $E\left(h_{n,t}\right)$ is a gamma function with n=1.5. Therefore,

$$E\left(h_{n,t}\right)=\Gamma(1.5)=\frac{\sqrt{\pi}}{2} \tag{4.27}$$

By substituting the above equation in (4.21), we get $h\left(\hat{y}_{n,t}^{(m)}\right)$ as

$$h\left(\hat{y}_{n,t}^{(m)}\right)=\begin{cases} \dfrac{\sqrt{\pi}}{2}, & \hat{y}_{n,t}>\gamma \\ -\dfrac{\sqrt{\pi}}{2}, & \hat{y}_{n,t}\le\gamma \end{cases} \tag{4.28}$$

The observation information $\hat{\mathbf{z}}_t^{(m)}$ can be estimated by substituting the above equation (4.28) in equation (4.18) for all the sensors. Now following the same procedure as in CRPF by using the above calculated observation information $\hat{\mathbf{z}}_t^{(m)}$ we can estimate the target state.

4.5 CRPF algorithm with noncoherent reception

In noncoherent reception the observation model given in equation (4.12) will remains same. Recalling the observation equation (4.12) we have

$$z_{n,t} = \left| r_{n,t} \right|^2 \tag{4.29}$$

where the notation $|\cdot|$ indicates the magnitude of a complex number.

From this observation information we have to estimate the target state. The CRPF algorithm which is used for estimating the target state as described in the previous chapter will remains same except in estimating the observation information $\hat{\mathbf{z}}_t^{(m)}$ which is used for calculating the cost function.

Recalling the equations (3.20), (3.21) and (3.22) we have

$$\hat{\mathbf{r}}_t^{(m)} = h\left(\hat{\mathbf{y}}_t^{(m)}\right) \tag{4.30}$$

$$\hat{\mathbf{y}}_t^{(m)} = g\left(\hat{\mathbf{x}}_t^{(m)}\right) + \mu_v \tag{4.31}$$

$$\hat{\mathbf{x}}_t^{(m)} = G_x \hat{\mathbf{x}}_{t-1}^{(m)} \tag{4.32}$$

where the elements of $g(.)$ are defined by (3.4) and those of $h(.)$ by (4.16), i.e., the elements of $h(.)$ are given by

$$h\left(\hat{y}_{n,t}^{(m)}\right) = \begin{cases} E\left(h_{n,t}\right), & \hat{y}_{n,t} > \gamma \\ 0, & \hat{y}_{n,t} \le \gamma \end{cases} \tag{4.33}$$

using equation (4.27), we get $h\left(\hat{y}_{n,t}^{(m)}\right)$ as

$$h\left(\hat{y}_{n,t}^{(m)}\right) = \begin{cases} \dfrac{\sqrt{\pi}}{2}, & \hat{y}_{n,t} > \gamma \\ 0, & \hat{y}_{n,t} \le \gamma \end{cases} \tag{4.34}$$

The observation information $\hat{\mathbf{r}}_t^{(m)}$ can be estimated by substituting the above equation (4.34) in equation (4.30) for all the sensors. The ED output of the estimated $\hat{\mathbf{r}}_t^{(m)}$ is

$$\hat{\mathbf{z}}_t^{(m)} = \left(\hat{\mathbf{r}}_t^{(m)}\right)^2 \tag{4.35}$$

Now following the same procedure as in CRPF by using the above calculated ED observation information $\hat{\mathbf{z}}_t^{(m)}$ we can estimate the target state.

4.6 SIMULATION RESULTS:

In simulations the channel aware particle filtering algorithm is applied for the same environment which is created in the previous chapter. So the parameters used in the simulations for target tracking in binary sensor networks will remains same except in the creation of fading channel between sensors and the fusion centre. For cresting the fading channel in MATLAB environment, the following parameters have been used in simulations.

- Variance of the Rayleigh fading Channel is unity. i.e., the fading channels between all the sensors and the fusion center have unit power.
- The cost function of the CRPF is defined using (3.18) and (3.19) with forgetting factor λ =0.25 and q =2.

To obtain the performance of state estimation, the Root Mean Square Error (RMSE) between the true state and estimated state is computed, which is given by

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(\mathbf{x}-\hat{\mathbf{x}}\right)^2} \tag{4.36}$$

For simulation, the deterministically deployed sensor network is considered.

Steps carried out for the simulation of target state generation in a WSN and observation information generation at the fusion center are:

1) Generate the target trajectory using the state equation (3.3) and the prior information.
2) Generate sensor measurements using the measurement equation (3.4) and compare the measurements with the threshold using equation (3.6), to send binary information to the fusion center indicating the presence or absence of the target.
3) The binary information should be in the form of NRZ signaling for coherent reception and it should be in the form of ON/OFF signaling for non coherent reception.
4) Repeat step-2 for each and every sensor and at each time instant t.

5) Create Rayleigh fading channel by generating the fading coefficients between all the sensors and the fusion center.
6) Generate the observations at the fusion center using equation (4.4) in the case of coherent reception or using equation (4.11) in the case of non coherent reception, for each and every sensor and at each time instant t.

Steps carried out for the simulation of auxiliary channel aware particle filtering algorithm with coherent reception for target tracking in WSN are:

7) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.
8) Compute the characterizing parameter using equation (3.14).
9) Estimate the pdf $p\left(z_{n,t}/s_{n,t}\right)$ using equation (4.10) and substitute this into equation (4.6) for estimating the pdf $p\left(z_{n,t}/\mathbf{x}_t^{(m)}\right)$.
10) Compute the weights using equation (3.7). Now based on these weights select the most promising particle streams.
11) Generate the new particles using equations (3.15) and (3.16).
12) Calculate the weights of newly generated particle streams and then estimate the target state using equation (3.17)
13) Repeat steps from 7 to 12 for each and every time instant t.
14) Compute the RMSE between the true state and the estimated state using equation (4.36).

Steps from 1 to 13 are repeated for each independent trial and RMSE is averaged over all independent trials.

Steps carried out for the simulation of auxiliary channel aware particle filtering algorithm with noncoherent reception for target tracking in WSN are:

7) Send the observations obtained at the fusion center through ED and the output of the ED can be computed using equation (4.12).
8) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.
9) Compute the characterizing parameter using equation (3.14).

10) Estimate the pdf $p\left(z_{n,t}/s_{n,t}\right)$ using equation (4.14) and (4.15). substitute $p\left(z_{n,t}/s_{n,t}\right)$ into equation (4.13) for estimating the pdf $p\left(z_{n,t}/\mathbf{x}_t^{(m)}\right)$.

11) Compute the weights using equation (3.7). Now based on these weights select the most promising particle streams.

12) Generate the new particles using equations (3.15) and (3.16).

13) Calculate the weights of newly generated particle streams and then estimate the target state using equation (3.17)

14) Repeat steps from 7 to 13 for each and every time instant t.

15) Compute the RMSE between the true state and the estimated state using equation (4.36).

Steps from 1 to 14 are repeated for each independent trial and RMSE is averaged over all independent trials.

Steps carried out for the simulation of cost reference channel aware particle filtering algorithm with coherent reception for target tracking in WSN are:

7) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.

8) Estimate the present state samples based on previous state samples using equation (4.20).

9) Substitute the present state samples in equation (4.19) for getting the measurement information at the sensors.

10) Take the decision at the sensors using equation (4.28) and estimate the observation information at the fusion center using equation (4.18).

11) Compute the risk function using equations (3.19) and compute the costs using equation (3.23). Now based on these costs select the most promising particle streams and directly remove the bad particles.

12) Generate the new particles using equations (3.15) and (3.16).

13) Calculate the costs of new generated particle streams using equation (3.18) and then estimate the target state using equation (3.17)

14) Repeat steps from 7 to 13 for each and every time instant t.

15) Compute the RMSE between the true state and the estimated state using equation (4.36).

Steps from 1 to 14 are repeated for each independent trial and RMSE is averaged over all independent trials.

Steps carried out for the simulation of cost reference channel aware particle filtering algorithm with noncoherent reception for target tracking in WSN are:

7) Send the observations obtained at the fusion center through ED and the output of the ED can be computed using equation (4.29).
8) Initially, generate the sequential Monte Carlo samples of the target state using prior distribution and set the weights of the particles to 1/M.
9) Estimate present state samples based on previous state samples using equation (4.32).
10) Substitute the present state samples in equation (4.31) for getting the measurement information at the sensors.
11) Take the decision at the sensors using equation (4.34) and estimate the observation information at the fusion center using equation (4.30) and the output of the ED can be computed using equation (4.35).
12) Compute the risk function using equations (3.19) and compute the costs using equation (3.23). Now based on these costs select the most promising particle streams and directly remove the bad particles.
13) Generate the new particles using equations (3.15) and (3.16).
14) Calculate the costs of new generated particle streams using equation (3.18) and then estimate the target state using equation (3.17)
15) Repeat steps from 7 to 14 for each and every time instant t.
16) Compute the RMSE between the true state and the estimated state using equation (4.36).

Steps from 1 to 15 are repeated for each independent trial and RMSE is averaged over all independent trials.

In Fig. 4.1, we display the root mean square errors (RMSEs) of the location estimate of the target obtained by the APF and CRPF algorithms with coherent and noncoherent reception that use binary sensor measurements, denoted by APF-coherent, APF-noncoherent, CRPF-coherent and CRPF-noncoherent, respectively. From Fig.4.1 we can observe that the APF has much better performance than CRPF regardless of the type of receiver. Similarly it can be seen that, APF with noncoherent reception does not have much degraded performance

compared to APF with coherent reception. The RMSEs were obtained by averaging over 100 different realizations.

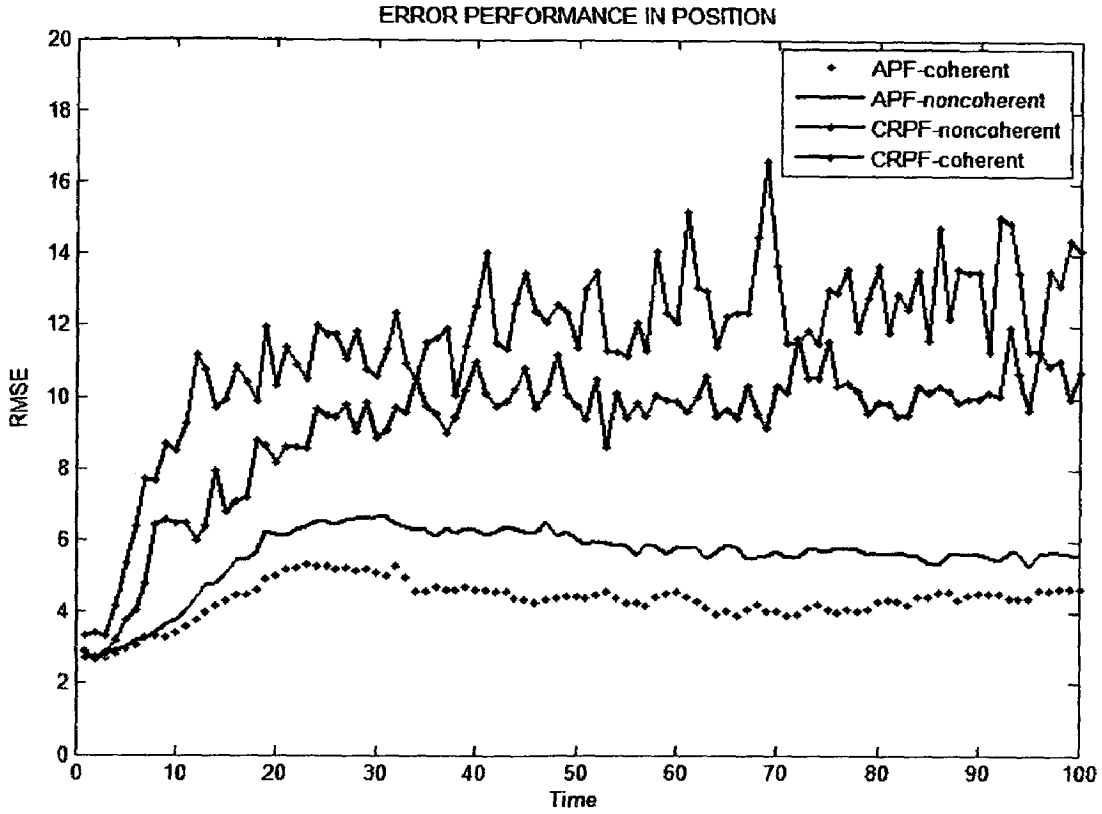


Fig4.1 RMSEs of the location estimates of the target obtained by the APF and CRPF algorithms with coherent and noncoherent reception.

In Fig. 4.2, we plot the cumulative distribution functions of the RMSE of APF and CRPF algorithms with coherent reception and noncoherent reception. From the graph we can say that 95% of the times the RMSE accumulated by the APF-coherent is less than 6m while 86% of the times the RMSE accumulated by the APF-noncoherent is less than 6m. Similarly, the RMSE accumulated by the CRPF-noncoherent is 90% of the times less than 12m while 55% of the times the RMSE accumulated by the CRPF-coherent is less than 12m.Clearly APF outperforms the CRPF considerably in the presence of fading channels. The RMSEs of 100 different trajectories were computed and summed over the entire time period.

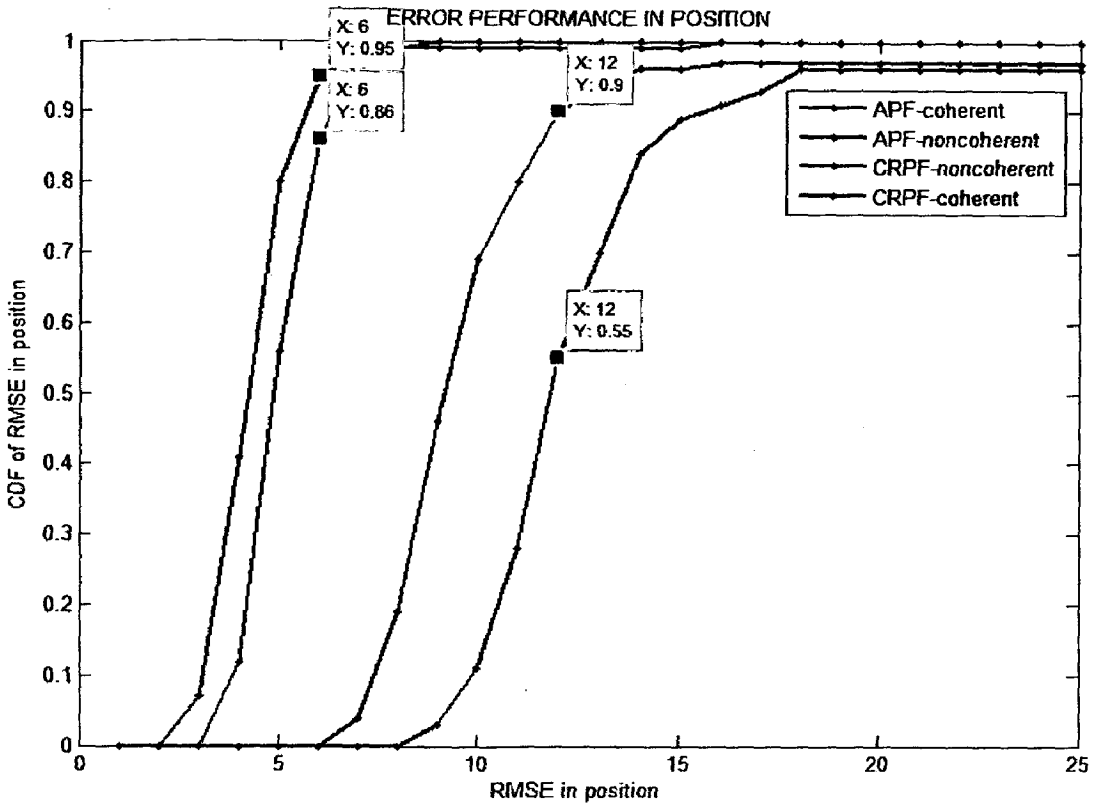


Fig 4.2 CDF of the RMSEs of APF-Bin and CRPF-Bin with coherent and noncoherent reception.

Chapter 5
CONCLUSIONS

Many practical applications which involve non-linear non-Gaussian state-space models require the estimation of the state which is highly intractable. Some of the these applications are localization of robots, estimating noisy digital communications signals, image processing, land vehicle navigation and aircraft tracking using radar measurements etc. The optimal solutions to these problems are often too computationally complex to implement by conventional signal processing methods. Particle filtering methods have become a very popular class of algorithms to solve these estimation problems numerically in an online manner, i.e. recursively as observations become available.

Particle filters are sequential Monte Carlo methods which can be applied to any state space model which generalize the Kalman filtering methods. Particle filter uses the concept of sequential importance sampling (SIS) for the recursive computation of a posteriori pdf by drawing of samples from the importance density with corresponding importance weights. This dissertation work is aimed at the application of particle filtering for target tracking in a wireless sensor networks and a comparative study of various versions of particle filtering. The conclusions drawn based on the simulation results are as follows:

Application of various versions of particle filtering for non linear estimation problem

We have used the state space approach for deriving the particle filtering algorithm for non linear estimation problem. Comparing the simulations results and by considering only computational complexity, the choice of ASIR gives reasonably good results with less number of particles as compared to other Particle Filters, when the process noise is equal to or greater than the measurement noise. Similarly when the process noise is very small compared to the measurement noise the sample impoverishment problem will be severe and we conclude that MCMC move particle filter will give reasonably good results compared to the other versions of particle filters.

Particle Filtering for target tracking in binary sensor networks

A detailed derivation and algorithm for evaluating Posterior Cramer-Rao Bound (PCRB) for target tracking in binary sensor networks (BSN) is given. Simulation results have shown that, the CRPF does not have much degraded performance as compared to APF even though it does not use probabilistic information and we can also observe that the RMSEs of APF and CRPF are very close to PCRB. From the simulations it can also be seen that there is no degradation in performance even though the emitted power by the target is assumed to be unknown. Considering the mixture noise model between the sensors and the fusion center, the simulation results have shown that, 85% of the times the RMSE accumulated by the CRPF-Bin is less than 12m while 53% of the times the RMSE accumulated by the APF-bin is less than 12m. Clearly CRPF-Bin outperforms the APF-Bin considerably in the presence of mixture noise.

Channel aware particle filtering for target tracking in binary sensor networks

The channel aware particle filtering approach is used for tracking the target in binary sensor networks by considering the fading channel between the sensors and the fusion center. In [16] SIR particle filter is applied for tracking the target in BSN in the presence of fading environment. To reduce the computational complexity a new CRPF algorithm is proposed for the above environment. We next consider the application of APF particle filter for improving the RMSE performance of the tracking system in the presence of fading environment. From the simulation results, it can be seen that APF outperforms the CRPF considerably in the presence of fading environment.

Future work

The optimal design of local sensor thresholds can be considered as design parameters for improving the tracking performance. Failure of sensors while tracking the target is a topic of significant interest. The channel aware particle filtering approach can be extended for multiple targets tracking in WSN [16]. The channel aware particle filtering approach can be applied to the problem of information-driven dynamic sensor collaboration in clutter environments with cost of increased power. In land vehicle navigation application the channel aware particle filtering approach along with Kalman

filter can be used for solving the fusion problem of the GPS, odometer, and digital road map measurements in the presence of GPS outages, and it may significantly improve the performance of the system compared to the Hybrid filter [18].

REFERENCES

[1] Zhe Chen, "Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond" Adaptive Syst. Lab., McMaster Univ., Hamilton, ON, Canada, 2003. Available:http://soma.crl.mcmaster.ca/~zhechen/download/ieee_bayesian.ps.

[2] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Transactions on Signal Processing*, vol.50, no.2, pp.174-188, February 2002.

[3] B. Ristic, S. Arulampalam, and N.J. Gordon," Beyond the Kalman Filter: Particle Filters for Tracking Applications," Artech House, Boston, MA, 2004.

[4] N.J. Gordon, D.J. Salmond, and A.F.M. Smith, "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation," *IEE Proceedings F, Radar, Sonar and Navigation*, vol. 140, no.2, pp. 107-113, April, 1993.

[5] Ying Liu, Benping Wang, Wang He, Jing Zhao, Zhi Ding, "Fundamental Principles and Applications of Particle Filters," *Proceedings of the 6th World Congress on Intelligent Control and Automation,* Dalian, China, vol. 2, pp.5327-5331, 21-23 June, 2006.

[6] Bolic, M.; Djuric, P.M.; Sangjin Hong, "New Resampling Algorithms for Particle Filters," Acoustics, Speech, and Signal Processing, 2003. *Proceedings. (ICASSP'03), 2003 IEEE International Conference,* vol.2, no., pp. II-589-92 vol.2, 6-10 April, 2003.

[7] Hol, Jeroen D.; Schon, Thomas B.; Gustafsson, Fredrik, "On Resampling Algorithms for Particle Filters," *Nonlinear Statistical Signal Processing Workshop, 2006 IEEE*, vol.1, no.1, pp.79-82, 13-15 September, 2006.

[8] J.Yick, Biswanath Mukherjee, Dipak Ghosal, "Wireless sensor network survey" *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol.52, no. 12, pp 2292-2330, August 2008.

[9] C.-Y. Chong and S. P. Kumar, "Sensor networks: Evolution opportunities and challenges," *Proceedings of IEEE*, vol. 91, no. 8, pp. 1247–1256, August 2003.

[10] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor Networks: A survey," *Computer Networks (Amsterdam, Netherlands: 1999)*, vol.38, no. 4, pp. 393–422, March 2002.

[11] M. Tubaishat and S. Madria, "Sensor networks: An overview," *IEEE Potentials*, pp. 20–23, April/May 2003.

[12] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *Proceedings of the1st International Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, pp. 150–161, November 5-7, 2003.

[13] F. Gustaffson, F. Gunnarsson, N. Bergman, U. Forssel, J. Jansson, R.Karlsson, and P.-J Nordlund, "Particle filtering for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing.*, vol. 50, no. 2, pp. 425–437, February 2002.

[14] X. Sheng and Y.-H. Hu, "Maximum likelihood multiple-source localization using acoustic energy measurements with wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 53, no. 1, pp. 44–53, January 2005.

[15] P. M. Djuric, M. Vemula, and M. F. Bugallo, "Target tracking by particle filtering in binary sensor networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2229–2238, June 2006.

[16] Onur Ozdemir, Ruixin Niu and Pramod K. Varshney, "Tracking in Wireless Sensor Networks Using Particle Filtering: Physical Layer Considerations," *IEEE Transactions on Signal Processing*, vol. 57, no. 5, pp. 1987-1999, May 2009.

[17] Hari Krishna Maganti, Daniel Gatica-Perez and Iain McCowan, "Speech Enhancement and Recognition in Meetings with an Audio-Visual Sensor Array," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2257-2269, November 2007.

[18] Christophe Boucher and Jean-Charles Noyer, "A Hybrid Particle Approach for GNSS Applications with Partial GPS Outages," *IEEE Transactions on Instrumentation and Measurement,* vol. 59, no. 3, pp. 498-505, March 2010.

[19] M. Pitt and N. Shepard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association,* vol. 94, no. 446, pp. 590–599, June 1999.

[20] M. F. Bugallo, J. Míguez, and P. M. Djuric´, "Positioning by cost reference particle filters: Study of various implementations," presented at the *2005 International Conference "Computer as a tool,"* EUROCON, Belgrade, Serbia and Montenegro, November 22-24, 2005.

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

[22] T. H. Kerr, "Status of CR-like lower bounds for nonlinear filtering," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 25, pp. 590–600, September 1989.

[23] R. Niu, B. Chen, and P. K. Varshney, "Fusion of decisions transmitted over Rayleigh fading channels in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 1018–1027, March 2006.

[24] R. Jiang and B. Chen, "Fusion of censored decisions in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 4, no. 6, pp. 2668–2673, November 2005.

[25] J.G. Proakis, "Digital Communications," 4th edition, McGraw-Hill, 2003.

[26] Theodore S.Rappaport, "Wireless Communications," Second edition, Eastern Economy Edition, 2006.