NUS
National University
of Singapore

Application of Improved Particle Filter in Multiple Maneuvering Target Tracking System

Liu Jing
(B.Eng, M.Eng)

PhD THESIS
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2006

Summary

Target tracking has been widely used in different fields such as surveillance, automated guidance systems, and robotics in general. The most commonly used framework for tracking is that of Bayesian sequential estimation. This framework is probabilistic in nature, and thus facilitates the modelling of uncertainties due to inaccurate models, sensor errors, environmental noise, etc. However, the application of the Bayesian sequential estimation framework to real world tracking problems is plagued by the difficulties associated with nonlinear and non-Gaussian situation. Realistic models for target dynamics and measurement processes are often nonlinear and non-Gaussian in type, so that no closed-form analytic expression can be obtained for tracking recursions. For general nonlinear and non-Gaussian models, particle filter has become a practical and popular numerical technique to approximate the Bayesian tracking recursions. This is due to its efficiency, simplicity, flexibility, ease of implementation, and modeling success over a wide range of challenging applications.

The purpose of this thesis is to develop effective particle filter based methods for target tracking application. The research work consists of four parts: i) particle filter based maneuvering target tracking algorithms, ii) particle filter based multiple target tracking algorithms, iii) particle filter based multiple maneuvering target tracking algorithms, and iv) the experiment of target tracking system based on multi-sensor fusion on a mobile robot platform.

The first part of the research work focuses on the single maneuvering target tracking algorithm. To estimate the maneuvering movement at different time steps, most of the traditional algorithms adopt the multiple possible model hypothesis. In this work, only one general model is utilized in the whole tracking process. Two different methods based on particle filter are proposed to track the wide variations in maneuvering movements.

The first method copes with the maneuvering target tracking problem using Markov chain Monte Carlo (MCMC) sampling based particle filter method, in which the particles are moved towards the posterior distribution of target state via MCMC sampling. However, the traditional MCMC sampling needs a lot of iterations to converge to the target posterior distribution, which is very slow and not suitable for real-time tracking. In order to speed up the convergence rate, a new method named adaptive MCMC based particle filter method, which is a combination of the adaptive Metropolis (AM) method and the importance sampling method, is proposed to track targets in real-time. Furthermore, a new method named interacting MCMC particle filter is proposed to avoid sample impoverishment induced by the maneuvering target movements, in which the importance sampling is replaced with interacting MCMC sampling. The sampling method is named interacting MCMC sampling since it incorporates the interaction of the particles in contrast with the traditional MCMC sampling method. The interacting MCMC sampling speeds up convergence rate effectively compared with the traditional MCMC sampling method.

The second method deals with the maneuvering target tracking problem based on the assumption that the maneuvering effect can be modeled by (part of) a white or colored noise process sufficiently well. The proposed method focuses on the identification of the equivalent process noise: the process noise is modeled as a dynamic system and a sampling based algorithm is proposed in the particle filter framework to identify the process noise.

In the second part of the research work, the multiple target tracking algorithms are discussed. State estimation and data association are two important aspects in multiple target tracking. Two algorithms based on particle filter are proposed to track multiple targets. The first algorithm uses the particle filter based multiple scan joint probabilistic data association filter (MS-JPDA filter), which examines the joint association hypothesis in a multi-scan sliding window and calculates the posterior marginal probability based on the multi-scan joint association hypothesis. The second algorithm, named multi-scan mixture particle filter, utilizes particle filter in the multiple target tracking and avoids the data association process. The posterior distribution of the target state is a multi-mode distribution and each mode corresponds to either the target or the clutter. In order to distinguish the targets from the clutters, multiple scan information is incorporated. Moreover, when new targets appear during tracking, new particles are sampled from the likelihood model (according to the most recent measurements) to detect the new modes appeared at each time step.

In the third part of the research work, a new algorithm is proposed to cope with the multiple maneuvering target tracking problem. The proposed algorithm is a combination of the process noise identification method for modeling highly maneuvering target, and the multi-scan JPDA algorithm for solving data association problem. The process noise identification process is effective in estimating both the maneuvering movement and the random acceleration of the target, avoiding the use of complicated multiple model approaches. The multi-scan JPDA is effective in maintaining the tracks of multiple targets using multiple scan information. The proposed algorithm is illustrated with an example involving tracking of two highly maneuvering, at times closely spaced and crossed, targets.

The fourth part of the research work is to build a target tracking system based on multi-sensor fusion, which is implemented on a mobile robot. A particle filter based tracker is developed in this work, which fuses color and sonar cues in a novel way. More specifically, color is introduced as the main visual cue and is fused with sonar localization cues. The generic objective is to track a randomly moving object via the pan-tilt camera and sonar sensors installed in the mobile robot. When moving randomly, the object's position and velocity vary quickly and are hard to track. This leads to serious sample impoverishment in particle filter and then the tracking algorithm fails. An improved particle filter with a new resampling algorithm is proposed to tackle this issue. Experiments are carried out to verify the proposed algorithm. The experimental results show that the robot is capable of continuously tracking a human's random movement at walking rate.

Successful results of target tracking should have a number of potential practical applications such as:

1. Improved human/computer interfaces: robot navigation system that can track the person while avoiding obstacles in certain environment.

2. Target detection and tracking is one of the important and fundamental technologies to develop real-world computer vision systems, e.g., visual surveillance systems and intelligent transport systems (ITSs).

3. Multiple maneuvering target tracking algorithm is important for the aircrafts tracking and monitoring system.

Acknowledgements

The past four years have presented a truly unique opportunity to study challenging problems in a world-class university. I am greatly indebted to the National University of Singapore for making this wonderful opportunity possible. I offer the sincerest of thanks to Professor Prahlad Vadakkepat, my thesis advisor, teacher and mentor. Your guidance, encouragement and support over the past four years have been nothing short of astounding. To my thesis committee members, Professor Xu Jianxin and Professor Tan Kok Kiong, many thanks for your kind help.

To my friends in the Mechatronics and Automation Lab, Guan Feng, Tang Kok Zuea, Wang Zhuping, Zhang Jin, Chan Kit Wai, Tan Shin Jiuh, Hong Fan, Liu Xin, Xiao Peng, Liu Yu, thanks for your valuable advices in my work.

To my family, thanks for your continual support over this time. To my husband, you have walked every step of this journey by my side. Thanks for your patience, encouragement, support and love. To my mother, wish you would recover from your illness soon.

Liu Jing

Dec. 1, 2006

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Target tracking has been widely used in different fields such as surveillance, automated guidance systems, and robotics in general. Typical examples include radar based tracking of aircrafts, sonar based tracking of sea animals or submarines, video based identification and tracking of people for surveillance or security purposes, laser based localization via mobile robot, and many more. The most commonly used framework for tracking is that of Bayesian sequential estimation. This framework is probabilistic in nature, and thus facilitates the modeling of uncertainties due to inaccurate models, sensor errors, environmental noise, etc. The general recursions update the posterior distribution of the target state, also known as the filtering distribution, through two stages: a prediction step that propagates the posterior distribution at the previous time step through the target dynamics to form the one step ahead prediction distribution, and a filtering step that incorporates the new data through Bayes rule to form the new filtering distribution. In theory the framework requires only the definition of a model for the target dynamics, a likelihood model for the sensor measurements, and an initial distribution for the target state.

The application of the Bayesian sequential estimation framework to real world tracking is plagued by the nonlinear and non-Gaussian nature of the problems. Realistic models for the target dynamics and measurement processes are often nonlinear and non-Gaussian, so that no closed-form analytic expression can be obtained for the tracking recursions. In fact, closed-form expressions are available only in a small number of cases. The most well-known of these arises when both the dynamic and likelihood models are linear and Gaussian, leading to the celebrated Kalman filter (KF) [1]. Since closed-form expressions are generally not available for nonlinear or non-Gaussian models, approximate methods are required.

For general nonlinear and non-Gaussian models, particle filtering [2, 3], also known as sequential Monte Carlo (SMC) [4, 5, 6], or CONDENSATION [7], has become a practical and popular numerical technique to approximate the Bayesian tracking recursions. This is due to its efficiency, simplicity, flexibility, ease of implementation, and modeling success over a wide range of challenging applications.

The target tracking process can be described as the task of estimating the state (states) of an target (targets) of interest both at the current time (filtering) and at any point in the future (prediction). The state estimation is conducted in two types of uncertainties: target model uncertainty and measurement uncertainty. Target model uncertainty exists because most of the targets do not follow predefined trajectories and their models are subject to random perturbations or maneuvers. The second type of uncertainty, measurement uncertainty, exists since the measured values from the targets are inaccurate (noisy), and the origins of the measurements are not perfectly certain. The measurements can be from the targets of interest, due to false alarms or clutters, or from other targets. In addition, the number of targets may not be necessarily known. In practice, the first type of uncertainty is mainly considered in maneuvering target tracking processes and the second type is considered in multiple target tracking processes.

This thesis investigates the particle filter based target tracking algorithms, including single maneuvering target tracking algorithm, multiple target tracking algorithm and multiple maneuvering target tracking algorithm. Finally, an experiment, where a mobile robot tracks a randomly moving object based on information from multiple sensors, is carried out to verify the proposed algorithm.

This chapter is organized as follows. Firstly, the Bayesian inference theory is introduced in Section 1.1. Then, considering that particle filter algorithm is used as the main method to solve the tracking problem in this thesis, the basic theory of particle filter and three variant algorithms of the standard particle filter are introduced in Section 1.2. A brief introduction on maneuvering target tracking algorithm and multiple target tracking algorithm is given to provide an outline of historical development and present status in these areas respectively in Sections 1.3 and 1.4. The objectives and organization of the thesis are presented in Sections 1.5 and 1.6 respectively.

1.1 Bayesian Inference Theory

Consider the dynamic system model representation:

$$x_{k+1} = f(x_k, v_k), \tag{1.1}$$

$$z_k = h(x_k, n_k). \tag{1.2}$$

Equation (1.1) is the state equation, where $x_k \in R^n$ is the state vector at time k, $f : R^n \times R^m \longrightarrow R^n$ is the system transition function and, v_k is a noise term whose known distribution is independent of time. Equation (1.2) is the observation equation, where $z_k \in R^p$ is the observation vector at time k, $h : R^n \times R^r \longrightarrow R^p$ is the measurement function and, n_k is a noise term whose known distribution is independent of both the system noise and time. Let $z_{1:k}$ denote $(z_1, ..., z_k)$, the available information at time k, and assume posterior distribution for x_1. The initial state of the system is known so that $p(x_1|z_0) = p(x_1)$.

The posterior distributions, $p(x_k|z_{1:k})(k \geq 1)$, and the associated expectation of some general function $g(x)$ are then estimated. The posterior distributions are estimated in two stages: prediction and update. In the prediction step, the posterior distribution, $p(x_{k-1}|z_{1:k-1})$, at time $k-1$ is propagated to the following time step, k, via the transition density $p(x_k|x_{k-1})$ as:

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}. \tag{1.3}$$

The update operation uses the latest measurement to modify the posterior distribution. This is achieved using the Bayesian theory, which is the mechanism to update the knowledge about the target state in light of extra information. The update equation is shown in (1.4):

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \tag{1.4}$$

where,

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k. \tag{1.5}$$

The associated expectation is computed as:

$$E(g(x_k)) = \int g(x_k)p(x_k|z_{1:k})dx_k. \tag{1.6}$$

The recurrence relations (1.3) and (1.4) form the basis for the optimal Bayesian solution. This recursive propagation of the posterior density is only a conceptual solution in that in general it cannot be determined analytically. Solutions do exist in a restrictive set of cases, including the Kalman filter and grid-based filters. Kalman filter assumes that the state function f and observation function h are linear and, v_k and n_k are additive Guassian noises of known variance. Grid-based filters provide optimal recursion of the filtered density if the state space is discrete and composed of a finite number of states.

However, considerations of realism imply that the linear and Gaussian assumptions are not always hold good in many applications.

There exist several approximate methods. The extended Kalman filter (EKF) [1] linearizes models with weak nonlinearities around the current state estimate, so that the Kalman filter recursions can still be applied. However, the performance of the EKF degrades rapidly as the nonlinearities become more severe. To alleviate this problem the unscented KF (UKF) [8, 9] maintains the second-order statistics of the target distribution by recursively propagating a set of carefully selected sigma points. This method requires no liberalization, and generally yields more robust estimates. One of the first attempts to deal with models with non-Gaussian state or observation noise is the Gaussian sum filter (GSF) [10] that works by approximating the non-Gaussian target distribution with a mixture of Gaussians. It suffers, however, from the same shortcoming as the EKF in that linear approximations are required. It also leads to a combinatorial growth in the number of mixture components over time, calling for ad-hoc strategies to prune the number of components to a manageable level. An alternative method, the approximate grid method, for non-Gaussian models that does not require any linear approximations has been proposed in [11]. It approximates the non-Gaussian state numerically with a fixed grid, and applies numerical integration for the prediction step and Bayes rule for the filtering step. However, the computational cost of the numerical integration grows exponentially with the dimension of the state-space, and the method becomes impractical for dimensions larger than four.

1.2 Particle Filter Algorithm

1.2.1 Basic Particle Filter Algorithm

Particle filtering is a sequential Monte Carlo methodology where the basic idea is the recursive computation of relevant probability distributions using the concepts of importance sampling and approximation of probability distributions with discrete random measures. The earliest applications of sequential Monte Carlo methods were in the area of growing polymers [12, 13], and later they expanded to other fields including physics and engineering. 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 then. The fast advances of computers in the last several years and the outstanding potential of particle filters have made them recently a very active area of research.

The sequential Monte Carlo approach is known variously as bootstrap filtering [2], the condensation algorithm [7], interacting particle approximations [14, 15], and survival of the fittest [16]. It is a technique for implementing a recursive Bayesian filter by Monte Carlo 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, the Monte Carlo characterization becomes an equivalent representation to the usual functional description of the posterior probability density function, and the particle filter approaches the optimal Bayesian estimate.

Particle filter uses sequential Monte Carlo methods for on-line learning within a Bayesian framework. Bayesian inference theory provides the framework to estimate the posterior distribution of the dynamic system and then Monte Carlo simulation methods are used to approximate the posterior distribution through sampled particles. In high-dimensional problems, the posterior probability distribution function is meaningfully nonzero only within a very small region [17]. The idea of biasing toward "importance" regions of the sample space then becomes essential for Monte Carlo simulation. In practice, a known easy to sample proposal distribution, known as importance sampling, is resorted to. Moreover, in order to process the new observation information as it arrives, sequential importance sampling is used to represent the importance weights in a recursive form. The Monte Carlo simulation method, importance sampling method, and sequential importance sampling method are respectively introduced in Sections 1.2.1.1, 1.2.1.2 and 1.2.1.3. A common problem with the Sequential Importance Sampling particle filter is the degeneracy phenomenon, where after a few iterations, all but one particle will have negligible weights. The degeneracy problem is introduced in Section 1.2.1.4, and two approaches used to reduce the effect of degeneracy are introduced in Sections 1.2.1.5 and 1.2.1.6 respectively.

1.2.1.1 Monte Carlo Simulation

The basic idea of Monte Carlo simulation is that the posterior distribution $p(x_{0:k}|z_{1:k})$ is approximated by a set of particles with associated weights $\{(x^i_{0:k}, w^i_k),\ i = 1, ..., NP\}$, where NP is the number of particles,

$$p(x_{0:k}|z_{1:k}) \approx \sum_{i=1}^{NP} w^i_k \delta(x_{0:k} - x^i_{0:k}). \tag{1.7}$$

where δ is the Dirac's delta function. The expectation of the general function $g(x)$ is approximated as:

$$\overline{E(g(x_{0:k}))} = \sum_{i=1}^{NP} g(x^i_{0:k}) w^i_k, \tag{1.8}$$

where, $x^i_{0:k}$ is the random sample drawn from the posterior distribution, $p(x_{0:k}|z_{1:k})$, and w^i_k is its associated weight.

1.2.1.2 Importance Sampling

Unfortunately it is often not possible to sample directly from the posterior distribution. This could be circumvented by drawing samples from a known proposal distribution $q(x_{0:k}|z_{1:k})$, which is easy to sample. The expectation of the general function $g(x)$ is represented in (1.9):

$$E(g(x_{0:k})) \quad = \int g(x_{0:k}) \tfrac{p(x_{0:k}|z_{1:k})}{q(x_{0:k}|z_{1:k})} q(x_{0:k}|z_{1:k}) dx_{0:k}. \tag{1.9}$$

The un-normalized importance weights $\tilde{w}_k(x_{0:k})$ are defined in (1.10).

$$\tilde{w}_k(x_{0:k}) = \frac{p(x_{0:k}|z_{1:k})}{q(x_{0:k}|z_{1:k})} \tag{1.10}$$

Substituting (1.9) in (1.10),

$$E(g(x_{0:k})) = \int [g(x_{0:k}) \tilde{w}_k(x_{0:k})] q(x_{0:k}|z_{1:k}) dx_{0:k}. \tag{1.11}$$

Using the Monte Carlo approximation:

$$\begin{aligned}\overline{E(g(x_{0:k}))} &= \frac{\frac{1}{NP}\sum_{i=1}^{NP} g(x_{0:k}^i)\tilde{w}_k(x_{0:k}^i)}{\frac{1}{NP}\sum_{i=1}^{NP} \tilde{w}_k(x_{0:k}^i)} \\ &= \sum_{i=1}^{NP} g(x_{0:k}^i) w_k(x_{0:k}^i)\end{aligned}, \tag{1.12}$$

where, $\{x_{0:k}^i, i = 1, \cdots, NP\}$ are independent random samples from $q(x_{0:k}|z_{1:k})$ and, $\{w_k(x_{0:k}^i), i = 1, \cdots, NP\}$ are the normalized importance weights.

$$w_k(x_{0:k}^i) = \frac{\tilde{w}_k(x_{0:k}^i)}{\sum_{i=1}^{NP} \tilde{w}_k(x_{0:k}^i)} \tag{1.13}$$

From this point onwards, $\tilde{w}_k(x_{0:k}^i)$ is simplified as $\tilde{w}_k^i$, and $w_k(x_{0:k}^i)$ as w_k^i.

1.2.1.3 Sequential Importance Sampling

For many problems, an estimate is required every time when new observation data arrives, for which a recursive filter is a convenient solution. The importance weight is represented in a recursive form. The received data is processed sequentially rather than in batch, so that it is neither necessary to store the complete data set nor to reprocess the existing data if new measurements become available.

To derive the weight update equation, the proposal distribution $q(x_{0:k}|z_{1:k})$ is factorized in (1.14),

$$q(x_{0:k}|z_{1:k}) = q(x_k|x_{0:k-1}, z_{1:k}) q(x_{0:k-1}|z_{1:k-1}) \ . \tag{1.14}$$

The posterior distribution is then expressed in a form as in (1.15).

$$\begin{aligned}p(x_{0:k}|z_{1:k}) &= \frac{p(z_k|x_k)p(x_k|x_{k-1})}{p(z_k|z_{1:k-1})} p(x_{0:k-1}|z_{1:k-1}) \\ &\propto p(z_k|x_k)p(x_k|x_{k-1})p(x_{0:k-1}|z_{1:k-1})\end{aligned} \tag{1.15}$$

By substituting (1.14) and (1.15) into (1.10), the normalized importance weight is derived in a sequential form,

$$\begin{aligned}w_k^i &\propto \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)p(x_{0:k-1}^i|z_{1:k-1})}{q(x_k^i|x_{0:k-1}^i, z_{1:k})q(x_{0:k-1}^i|z_{1:k-1})} \\ &= w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{0:k-1}^i, z_{1:k})}.\end{aligned} \tag{1.16}$$

If $q(x_k|x_{0:k-1}, z_{1:k}) = q(x_k|x_{k-1}, z_k)$, then the importance weight is only dependent on x_{k-1} and z_k. This is particularly useful when only a filtered estimate of $p(x_k|z_{1:k})$ is required at each time step. From this point onwards it is assumed so, except when explicitly stated otherwise. In such scenarios, only x_k needs to be stored; therefore, the path $x_{0:k-1}$ and the history of observations $z_{1:k-1}$ can be discarded. The modified normalized importance weight is then:

$$w_k^i \propto w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)}, \tag{1.17}$$

and the posterior density $p(x_k|z_{1:k})$ can be approximated as,

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{NP} w_k^i \delta(x_k - x_k^i). \tag{1.18}$$

It can be shown that as $NP \to \infty$, the approximation (1.18) approaches the true posterior density $p(x_k|z_{1:k})$.

The SIS algorithm thus consists of recursive propagation of the weights and support points as each measurement is received sequentially. A pseudo-code description of this algorithm is given by Algorithm 1.1.

Algorithm 1.1: SIS Particle Filter

```
$[\{x_k^i, w_k^i\}_{i=1}^{NP}] = SIS[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{NP}, z_k]$
• FOR $i = 1 : NP$
    – Draw $x_k^i$ from the distribution $q(x_k|x_{k-1}^i, z_k)$
    – Assign the particle a weight, $w_k^i$, according to (1.17)
• END FOR
```

1.2.1.4 Degeneracy Problem

A common problem with the SIS particle filter is the degeneracy phenomenon, where after a few iterations, all but one particle will have negligible weights. It has been shown in [4] that the variance of the importance weights can only increase over time, and thus, it is impossible to avoid the degeneracy phenomenon. This degeneracy implies that a large computational effort is devoted to updating particles whose contribution to the approximation is almost zero. A suitable measure of degeneracy of the algorithm is the effective sample size introduced in [18] and [6] and defined as,

$$N_{eff} = \frac{NP}{1 + Var(w_k^{*i})}, \tag{1.19}$$

where,

$$w_k^{*i} = \frac{p(x_k^i | z_{1:k})}{q(x_k^i | x_{k-1}^i, z_k)}, \tag{1.20}$$

is referred to as the true weight, which cannot be evaluated exactly. An estimate $\widehat{N_eff}$ of N_{eff} can be obtained by,

$$\widehat{N_eff} = \frac{1}{\sum_{i=1}^{NP} (w_k^i)^2}, \tag{1.21}$$

where w_k^i is the normalized weight. Notice that $N_{eff} \leq NP$, and small N_{eff} indicates severe degeneracy. Clearly, the degeneracy problem is an undesirable effect in particle filters. The brute force approach to reduce its effect is to use a very large NP. This is often impractical; therefore, we rely on two other methods:

a) good choice of importance density, and,

b) use of resampling.

These are described in Sections 1.2.1.5 and 1.2.1.6 respectively.

1.2.1.5 Good Choice of Importance Density

The first method involves choosing the importance density $q(x_k | x_{k-1}^i, z_k)$ to minimize $Var(w_k^{*i})$ so that N_{eff} is maximized. The optimal importance density function that minimizes the variance of the true weights conditioned on and has been shown [4] to be:

$$\begin{aligned} q(x_k | x_{k-1}^i, z_k)_{opt} &= p(x_k | x_{k-1}^i, z_k) \\ &= \frac{p(z_k | x_k, x_{k-1}^i) p(x_k | x_{k-1}^i)}{p(z_k | x_{k-1}^i)}. \end{aligned} \tag{1.22}$$

Substitution of (1.22) into (1.17) yields,

$$\begin{aligned} w_k^i &\propto w_{k-1}^i p(z_k|x_{k-1}^i) \\ &= w_{k-1}^i \int p(z_k|x_k')p(x_k'|x_{k-1}^i)dx_k' \end{aligned} \tag{1.23}$$

The choice of importance density (1.22) is optimal since for a given x_{k-1}^i, w_k^i takes the same value, whatever sample is drawn from $q(x_k|x_{k-1}^i, z_k)_{opt}$. Hence, conditional on x_{k-1}^i, $Var(w_k^{*i}) = 0$. This is the variance of different w_k^i resulting from different sampled x_k^i.

This optimal importance density (1.22) suffers from two major drawbacks. It requires the ability to sample from $p(x_k|x_{k-1}^i, z_k)$ and to evaluate the integral over the new state. In general, it may not be straightforward to carry out either. There are two cases where the use of the optimal importance density is possible.

The first case is when x_k is a member of a finite set. In such cases, the integral in (1.23) becomes a sum, and sampling from $p(x_k|x_{k-1}^i, z_k)$ is possible. An example of an application, when x_k is a member of a finite set, is a Jump-Markov linear system for tracking maneuvering targets [19]. The discrete model state (defining the maneuver index) is tracked using a particle filter, and (conditioned on the maneuver index) the continuous base state is tracked using a Kalman filter.

Analytic evaluation is possible for a second class of models for which $p(x_k|x_{k-1}^i, z_k)$ is Gaussian [4, 20]. This can occur if the dynamics are nonlinear and the measurements are linear.

For many other models, such analytic evaluations are not possible. However, it is possible to construct suboptimal approximations to the optimal importance density by using local linearization techniques [4]. Such linearizations use an importance density that is a Gaussian approximation to $p(x_k|x_{k-1}, z_k)$. Another approach is to estimate a Gaussian approximation to $p(x_k|x_{k-1}, z_k)$ using the unscented transform [21].

In practice, it is often convenient to choose the importance density to be the prior.

$$q(x_k|x_{k-1}^i, z_k) = p(x_k|x_{k-1}^i) \tag{1.24}$$

Substitution of (1.24) into (1.17) yields,

$$w_k^i \propto w_{k-1}^i p(z_k|x_k^i). \tag{1.25}$$

This would seem to be the most common choice of importance density since it is intuitive and simple to implement. However, there are a plethora of other densities that can be used, and the choice is the crucial design step in the design of a particle filter.

It is often the case that a good importance density is not available. For example, if the prior $p(x_k|x_{k-1})$ is used as the importance density and is a much broader distribution than the likelihood $p(z_k|x_k)$, then only a few particles will have high weights. Methods exist for moving the particles to be in the right place. The use of bridging densities [5] and progressive correction [22] introduce intermediate distributions between the prior and likelihood. The particles are then re-weighted according to these intermediate distributions and resampled, which "herds" the particles into the right part of the state space.

Another approach known as partitioned sampling [23] is useful if the likelihood is very peaked but can be factorized into a number of broader distributions. Typically, this occurs because each of the partitioned distributions are functions of some (not all) of the states. By treating each of these partitioned distributions in turn and resampling on the basis of each such partitioned distribution, the particles are again herded toward the peaked likelihood.

1.2.1.6 Resampling

The second method by which the effects of degeneracy can be reduced is to use resampling whenever a significant degeneracy is observed (i.e., when N_{eff} falls below some threshold N_T). The basic idea of resampling is to eliminate particles that have small weights and to concentrate on particles with large weights. The resampling step involves generating a new set $\{x_k^{*i}\}_{i=1}^{NP}$ by resampling (with replacement) NP times from an approximate discrete representation of $p(x_k|z_{1:k})$ given by,

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{NP} w_k^i \delta(x_k - x_k^i), \tag{1.26}$$

so that $Pr(x_k^{*i} = x_k^j) = w_k^j$. The resulting samples are in fact independent and identically distributed (i.i.d.) samples from the discrete density (1.26); therefore, the weights are now reset to $w_k^i = 1/NP$. It is possible to implement this resampling procedure in operations by sampling NP ordered uniforms using an algorithm based on order statistics [24, 25]. Note that other efficient (in terms of reduced MC variation) resampling schemes, such as stratified sampling and residual sampling [6], may be applied as alternatives to this algorithm. Systematic resampling [26] is the scheme which is simple to implement, taking NP times, and minimizing the MC variation. Its operation is described in Algorithm 1.2, where $U(a, b)$ is the uniform distribution on the interval (a, b) (inclusive of the limits). For each resampled particle x_k^{*j}, this resampling algorithm also stores the index of its parent, which is denoted by i^j. A generic particle filter is then described by Algorithm 1.3.

Algorithm 1.2: Resampling Algorithm

```
$[\{x_k^{*j}, w_k^j, i^j\}_{j=1}^{NP}] = RESAMPLE[\{x_k^i, w_k^i\}_{i=1}^{NP}]$

• Initialize the CDF: $c_1 = 0$
• FOR $i = 2 : NP$
    – Construct CDF: $c_i = c_{i-1} + w_k^i$
• END FOR
• Start at the bottom of the CDF: i=1
• Draw a starting point: $\mu_1 \sim U(0, NP^{-1})$
• FOR $j = 1 : NP$
    – Move along the CDF: $\mu_j = \mu_1 + NP^{-1}(j-1)$
    – WHILE $\mu_j > c_i$
        ∗ i=i+1
    – END WHILE
    – Assign sample: $x_k^{*j} = x_k^i$
    – Assign weight: $w_k^j = NP^{-1}$
    – Assign parent: $i^j = i$
• END FOR
```

Algorithm 1.3: Genetic Particle Filter

$[\{x_k^i, w_k^i\}_{i=1}^{NP}] = PF[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{NP}, z_k]$

```
• FOR $i = 1 : NP$
    – Draw $x_k^i \sim q(x_k | x_{k-1}^i, z_k)$
    – Assign the particle a weight, $w_k^i$, according to (1.17)
• END FOR
• Calculate the total weight: $t = SUM[\{w_k^i\}_{i=1}^{NP}]$
• FOR $i = 1 : NP$
    – Normalize: $w_k^i = t^{-1} w_k^i$
• END FOR
• Calculate $\widehat{N_{eff}}$ using (1.21)
• IF $\widehat{N_{eff}} < N_T$
    – Resample using Algorithm 1.2:
      $[\{x_k^i, w_k^i, -\}_{i=1}^{NP}] = RESAMPLE[\{x_k^i, w_k^i\}_{i=1}^{NP}]$
• END IF
```

Although the resampling step reduces the effects of the degeneracy, it introduces other practical problems. First, it limits the opportunity to parallelize since all the particles must be combined. Second, the particles that have high weights are statistically selected many times. This leads to a loss of diversity among the particles as the resultant sample will contain many repeated points. This problem, which is known as sample impoverishment, is severe in the case of small process noise. In fact, for the case of very small process noise, all particles will collapse to a single point within a few iterations. Third, since the diversity of the paths of the particles is reduced, any smoothed estimates based on the particles' paths degenerate. Schemes exist to counteract this effect. One approach considers the states for the particles to be predetermined by the forward filter and then obtains the smoothed estimates by recalculating the particles' weights via a recursion from the final to the first time step [27]. Another approach is to use MCMC [28].

There have been some systematic techniques proposed recently to solve the problem of sample impoverishment. One such technique is the resample-move algorithm [29], which draws conceptually on the same technologies of importance sampling-resampling and MCMC sampling, and avoids sample impoverishment. It does so in a rigorous manner that ensures the particles asymptotically approximate samples from the posterior and, therefore, is the method of choice of the authors. An alternative solution to the same problem is regularization [5]. This approach is frequently found to improve performance, despite a less rigorous derivation and is included here in preference to the resample-move algorithm since its use is so widespread.

1.2.2 Variant Algorithms of the Standard Particle Filter

The particle filtering algorithm presented in Section 1.2.1 forms the basis for most particle filters that have been developed so far. The various versions of particle filters proposed in the literature can be regarded as special cases of this general SIS algorithm. These special cases can be derived from the SIS algorithm by an appropriate choice of importance sampling density and/or modification of the resampling step. Three variant particle filters are listed below and the detailed descriptions on them can be found in [30].

i) sampling importance resampling (SIR) filter [2];

ii) auxiliary sampling importance resampling (ASIR) filter [31];

iii) regularized particle filter (RPF) [5].

1.3 Maneuvering Target Tracking Algorithms

In the history of development of maneuvering target tracking techniques, single model based adaptive Kalman filtering came into existence first [32, 33, 34]. Aidala [32] proposed the adaptive Kalman filtering method based on single motion model of the moving target in 1973. In the proposed method, the target maneuvering is estimated by adjusting the Kalman gain.

Decision-based techniques, which detect the manoeuvre and then cope with it effectively, appeared next. Examples of this approach include the input estimation (IE) techniques [35, 36], the variable dimension (VD) filter [37], the two-stage Kalman estimator [38] etc. In addition to basic filtering computation, these techniques require additional effort to detect the target maneuvers.

The decision based techniques are followed by multiple-model algorithms, which describe the motion of a target using multiple sub-filters. The generalized pseudo-Bayesian (GPB) method [39], the interacting multiple model (IMM) method [40, 41], and the adaptive interacting multiple model (AIMM) method [42] are included in this kind of approach. Using the multiple model based methods which use more than one model to describe the motion of the target, performance is enhanced. Among them, interact multiple model algorithm (IMM) is the most common one. The main feature of the IMM algorithm is its ability to estimate the state of a dynamic system with several behavior modes which can “switch” from one to another. In particular, the IMM estimator can be a self-adjusting variable-bandwidth filter, which makes it natural for tracking maneuvering targets.

The above methods solve the target tracking problem using linear tracking filters, mainly Kalman filter. In these methods target maneuvers are often described by linear models. However, the linear solution may not always be good especially in the condition when the state or measurement equation is nonlinear and the noises are non-Gaussian, for example, when the filter update is slow or the target maneuver is large. More recently, nonlinear filtering techniques have been gaining more attention and the particle filter algorithm is the most common one among them.

Particle filter, which uses sequential Monte Carlo methods for on-line learning within a Bayesian framework, can be applied to any state-space models. Particle filter is more suitable than Kalman filter and EKF when dealing with non-linear and non-Gaussian estimation problems.

The application of particle filter in maneuvering target tracking has been paid attention only in recent years [43, 44, 45, 46, 47, 5, 48, 49]. The simplest method is to implement the maneuvering target tracking problem in a particle filter framework. Karlsson [43] and Ikoma [44] applied optimal recursive Bayesian filters directly to the nonlinear target model.

Recently, several approaches, which use multiple models to describe the changing maneuvering model, have been proposed in the particle filter framework. One of the methods is based on the auxiliary particle filter. In [45], Karlsson used an auxiliary particle filter to track a highly maneuvering target. In this method, each particle is split deterministically into a number of possible maneuver hypotheses with each hypothesis corresponding to a specific model.

Other methods focus on how to switch between different motion models. In [46], Bayesian switching structure is chosen as the principle which determines switching between different models. A set of models are utilized to cope with the unknown maneuver. Moreover, to deal with non-Gaussian noise, Cauchy distribution is used as the system noise distribution. In [47] and [5], the maneuvering target tracking system is treated as a jump Markov linear system. MCMC process is used as the selection scheme to choose the motion model from a set of candidate models at some specific time step.

However, in the above approaches [45, 46, 47, 5], the possible motion models and transition probability matrices are assumed as known. In practice, the dynamics is hard to break up into several different motion models and the model transition probabilities are difficult to obtain. A general model is needed to cope with the wide variety of motions exhibited by the maneuvering target.

1.4 Multiple Target Tracking Algorithms

In the process of multiple target tracking, two distinct problems have to be solved jointly: data association and state estimation. Data association is a key problem in multiple targets tracking and determines which measurement corresponds to which target. A large number of strategies are available to solve the data association problem. These can be broadly categorized as either single frame assignment methods, or multi-frame assignment methods.

In the multi-frame data association methods, the measurements from one or more frames are associated with established tracks by solving an optimization problem with global constraints [50, 51].

In this thesis, the focus is put on the single frame methods. The multiple hypothesis tracking (MHT) [52] was proposed by Read in 1979. The MHT attempts to keep track of all the possible association hypotheses over time. This is an NP-hard problem, since the number of association hypotheses grows exponentially over time. Thus methods are required to reduce the computational complexity.

Compared with MHT, the nearest neighbor (NN) algorithm [53] is computationally simple and easily to implement. In NN algorithm, each target is associated with the closest measurement in the target space. However, such a simple procedure prunes away many feasible hypotheses.

In this respect the joint probabilistic data association (JPDA) filter [53, 54] is more appealing. At each time step infeasible hypotheses are pruned away using a gating procedure. A filtering estimate is then computed for each of the remaining hypotheses, and combined in proportion to the corresponding posterior hypothesis probabilities. The main shortcoming of the JPDA filter is that, to maintain tractability, the final estimate collapses to a single Gaussian, thus discarding pertinent information. Subsequent work addressed this shortcoming by proposing strategies to reduce the number of mixture components in the original mixture to a tractable level [55, 56]. Still, many feasible hypotheses may be discarded by the pruning mechanisms.

The probabilistic multiple hypotheses tracker (PMHT) [57, 58] assumes the association variables to be independent from the pruning work, which leads to an incomplete data problem that can be efficiently solved using the expectation maximization (EM) algorithm [59]. However, the PMHT is a batch strategy, and thus not suitable for online applications. The standard version of the PMHT is also generally outperformed by the JPDA filter. Some of the reasons for this, and a number of possible solutions, are discussed in [60].

Even though methods to solve the data association problem do not usually rely on linear and Gaussian models, this assumption is often made to simplify hypothesis evaluation for target originated measurements. For example, nonlinear models can be accommodated by suitable linearization using EKF. As for EKF, however, the performance of the algorithms degrades as the nonlinearities become more severe.

In recent years, particle filter has been introduced to estimate non-linear non-Gaussian dynamic processes for multiple target tracking. A stochastic simulation Bayesian method is reported in [61] for multiple target tracking. In this method, random samples are used to represent the posterior distribution of the target state. However, only one target is considered in the example outlined. More recently, particle filter has been applied with great success to different fields of multiple target tracking including computer vision [23, 62], mobile robot localization [63, 64] and air traffic control [65, 66]. The various methods adopted fall into the following five categories.

The first category introduces MCMC strategies to calculate the association probabilities. In [65] the distribution of the association hypotheses is calculated using a Gibbs sampler [67] at each time step. The method is similar in spirit to the one described in [68] which uses the MCMC techniques [69] to compute the correspondences between image points within the context of stereo reconstruction. The main problem with these MCMC strategies is that they are iterative in nature and take an unknown number of iterations to converge. They are thus not entirely suitable for online applications.

The second category treats the association variables as state variables. In [70], the association variables are sampled from an optimally designed importance distribution. The method is intuitively appealing since the association hypotheses are treated in a similar fashion to the target state, so that the resulting algorithm is non-iterative. It is, however, restricted to jump Markov linear systems (JMLS) [19]. An extension of this strategy based on the auxiliary particle filter (APF) [31] and the UKF, which is applicable to general jump Markov systems (JMS), is presented in [71]. Another similar approach is described in [72]. Samples for the association hypotheses are generated from an efficient proposal distribution based on the notion of a soft-gating of the measurements.

The third category combines the JPDAF with particle techniques to accommodate general nonlinear and non-Gaussian models [63, 73, 74, 43]. The data association problem is addressed directly in the context of particle filtering.

The fourth category relates to multiple target tracking problems based on raw measurements [75, 76]. These, so-called, track before detect (TBD) strategies construct a generative model for the raw measurements in terms of a multi-target state hypothesis, thus avoiding an explicit data association step. However, such measurements are not always readily available in practical systems, and may lead to a larger computational complexity.

The above four categories of methods use particles whose dimension is the sum of those of the individual state spaces corresponding to each target. They all suffer from the curse of dimensionality problem since with the increase in the number of targets, the size of the joint state-space increases exponentially. If care is not taken in the design of proposal distributions an exponentially increasing number of particles may be required to cover the support of the multi-target distribution and maintain a given level of accuracy.

The fifth category avoids the dimension problem through exploring the particle filter' ability to track multiple targets in a single-target state space. As pointed out in [77], particle filters may perform poorly when the posterior distribution of the target state is multiple-mode due to ambiguities and multiple targets in single-target state space. To circumvent this problem, a mixture particle filter method is introduced in [77], where each mode is modeled with an individual particle filter that forms part of the mixture. The filters in the mixture interact only through the computation of the importance weights. By distributing the resampling step to individual filters, the well known problem of sample impoverishment is avoided, which is largely responsible for losing track.

1.5 Objectives of the Thesis

In general, the objective of this thesis is to develop constructive and systematic target tracking algorithms in particle filter framework.

The first objective is to develop particle filter based methods for single maneuvering target tracking application. Two methods, MCMC based particle filter and process noise estimation based particle filter, are proposed to tackle the maneuvering target tracking problem.

The first method copes with the maneuvering target tracking problem by moving the particles towards the target posterior distribution via MCMC sampling. The target's state variables, such as the position and velocity, vary quickly and are not restricted to a fixed dynamic model when it performs maneuvering movements. New features of posterior distribution of the target state are encountered during the tracking process. In the MCMC based particle filter methods, the particles are moved towards the target posterior distribution to adapt to the new features formed during the tracking process. However, the traditional MCMC sampling needs a lot of iterations to converge to the target posterior distribution, which is very slow and not suitable for real-time tracking problem. In order to speed the convergence rate, a new method named adaptive MCMC based particle filter method, which is the combination of the adaptive Metropolis (AM) method and the importance sampling method, is proposed to tackle the real-time tracking problem. Furthermore, another novel method named interacting MCMC particle filter is proposed to avoid the sample impoverishment problem induced by the maneuvering movement, in which the importance sampling is replaced with interacting MCMC sampling. The sampling method is named interacting MCMC sampling since it incorporates the interaction of the particles in contrast with the traditional MCMC sampling method. The interacting MCMC sampling also speeds up the convergence rate effectively compared with the traditional MCMC sampling method.

The second method deals with the maneuvering target tracking problem based on the assumption that the maneuvering effect can be modeled by (part of) a white or colored noise process sufficiently well. This fundamental assumption converts the problem of maneuvering target tracking to that of state estimation in the presence of non-stationary process noise with unknown statistics. The proposed method focuses on the estimation of the equivalent process noise: the process noise is modeled as a dynamic system and a sampling based algorithm is proposed in the particle filter framework to deal with process noise estimation problem.

The second objective of this thesis is to cope with the multiple target tracking problem using improved particle filter algorithms. Two algorithms are proposed to solve the multiple target tracking problem. The first, which is referred as the particle filter based multi-scan JPDA filter, is an extension of the single scan JPDA methods proposed in [63, 73, 78]. In the proposed approach, the distributions of interest are the marginal filtering distributions for each of the targets, which is approximated with particles. The multi-scan JPDA filter examines the joint association hypothesis in a multi-scan sliding window and calculates the posterior marginal probability based on the multi-scan joint association hypothesis. Compared with the single scan JPDA methods, the multi-scan JPDA method uses richer information, which results in better estimated probabilities.

The second method, named as multi-scan mixture particle filter method, applies the particle filter method directly in the multiple target tracking process and avoids the data association problem. The proposed algorithm can track varying number of targets in a cluttered environment. The posterior distribution of the target state is a multiple-mode distribution and each mode either corresponds to a target or a clutter. In order to distinguish the targets from the clutters, multiple scan information is incorporated. Moreover, to tackle with the appearance of new targets, new particles are sampled from the likelihood model (according to the most recent measurements) to detect the new modes appeared at each time step. The proposed algorithm is capable of initiating tracks, maintaining the states of the targets, and detecting the appearance and disappearance of the targets.

The third objective of the thesis is to propose a new algorithm to tackle the multiple maneuvering target tracking problem. The proposed algorithm is a combination of the process noise identification method for modeling a highly maneuvering target, and the multi-scan JPDA algorithm for solving data association problem, in particle filter framework.

The fourth objective of the thesis is to build a target tracking system based on multi-sensor fusion implemented on a mobile platform, the Magellan robot. The issues associated with the integration of different subsystems (controllers and sensors), are also studied. The robot is capable of continuously tracking a human's random movement at walking rate.

The algorithms proposed have a number of possible potential applications such as:

1) Improved human/computer interfaces: robot navigation system that can track the person while avoiding obstacles in outside environment.

2) Target detection and tracking: real-world computer vision system that can assist in visual surveillance and intelligent vehicle monitoring.

3) Aircrafts tracking and monitoring: aircraft traffic control system that can track aircrafts.

1.6 Organization of the Thesis

The thesis is organized as described in the following:

In Chapter 2, two algorithms for single maneuvering target tracking are proposed in the classical particle filter framework. The first algorithm copes with the maneuvering target tracking problem by moving the particles towards the target posterior distribution area via MCMC sampling. Two improved MCMC sampling methods, the adaptive MCMC sampling method and interacting MCMC sampling method, are utilized to speed up the convergence rate.

The second algorithm deals with the maneuvering target tracking problem based on the assumption that the maneuvering effect can be modeled by (part of) a white or colored noise process sufficiently well. The proposed method focuses on the estimation of the equivalent process noise using particle filter algorithm.

In Chapter 3, two methods are proposed for multiple target tracking: the particle filter based multi-scan JPDA filter and multi-scan mixture particle filter. The particle filter based multi-scan JPDA filter is an extension of the single scan JPDA algorithms, which addresses the data association problem in multi-scan sliding window. The multi-scan mixture particle filter applies the particle filter method directly to the multiple target tracking process and avoids the data association problem.

In Chapter 4, a new algorithm, which is a combination of the process noise identification method for modeling highly maneuvering target, and the multi-scan JPDA algorithm for solving data association problem, is proposed to deal with the multiple maneuvering target tracking problem.

Chapter 5 is about a target tracking system based on multi-sensor fusion implemented on a Magellan mobile robot. The improved particle filter using a new adaptive resampling method is utilized effectively in tracking a randomly moving object.

Finally, conclusions and proposals for further research are made in Chapter 6.

Chapter 2

Particle Filter Based Maneuvering Target Tracking

Recently, nonlinear filtering techniques have been gaining momentum in maneuvering target tracking and particle filter is the most popular one among them. The popularity stems from its simplicity, flexibility and ease of implementation, especially the ability to deal with non-linear and/or non-Gaussian estimation problems.

The particle filter methods applied in the maneuvering target tracking can be divided into two categories: single model based methods and multiple model based methods. For the single model based methods, Karlsson [43] and Ikoma [44] applied optimal recursive Bayesian filters directly to the nonlinear target model.

More recently, several kinds of approaches, which use multiple models to describe the maneuvering models, have been proposed in the particle filter framework [45, 46, 47, 5]. A common assumption made in the multiple-model approaches is that the possible motion models and transition probability matrices are known. In practice, the dynamics is hard to break up into several different motion models and the model transition probabilities are difficult to obtain. A general model is needed to cope with the wide variety of motions exhibited by maneuvering targets.

In this thesis, a single dynamic model is adopted during the tracking process. Two methods, MCMC based particle filter and process noise estimation based particle filter, are proposed to tackle the maneuvering target tracking problem.

In the first method the wide variation of the maneuvering movement is tracked by moving the particles towards the target posterior distribution area via MCMC sampling. The target's state variables, such as position and velocity, vary and are not restricted to a fixed dynamic model when the target performs maneuvering movements. New features of the target posterior distribution emerge during the tracking process. In the proposed method, the particles are moved towards the target posterior distribution area to adapt to the new features emerged during tracking. The MCMC moves also ensure the particles asymptotically approximate samples from the posterior distribution.

However, the traditional MCMC sampling needs a lot of iterations to converge to the target posterior distribution, which is very slow and not suitable for real-time tracking. In order to speed up the convergence rate, a new method named adaptive MCMC based particle filter method, which is a combination of the adaptive Metropolis (AM) method and the importance sampling method, is proposed. Furthermore, another new method named interacting MCMC particle filter is proposed to avoid sample impoverishment induced by maneuvering movement, in which the importance sampling is replaced with interacting MCMC sampling. The sampling method is named interacting MCMC sampling since it incorporates the interaction of particles in contrast with the traditional MCMC sampling method. The interacting MCMC sampling also speeds up the convergence rate effectively compared with the traditional MCMC sampling method.

The second method deals with the maneuvering target tracking problem based on the assumption that the maneuver effect can be modeled by (part of) a white or colored noise process sufficiently well. This fundamental assumption converts the problem of maneuvering target tracking to that of state estimation in presence of non-stationary process noise with unknown statistics. This method focuses on the estimation of equivalent process noise: the process noise is modeled as a dynamic system and a sampling based algorithm is proposed in the particle filter framework to deal with the process noise estimation problem.

This chapter is organized as follows. The MCMC based particle filter method is introduced in Section 2.1, and the process noise estimation based method is presented in Section 2.2. The conclusions are drawn in Section 2.3.

2.1 MCMC Based Particle Filter Algorithm

The target's state variables, such as position and velocity, vary and are not restricted to a fixed dynamic model when the target performs maneuvering movements. New features of posterior distribution of the target state emerge during the tracking process. The standard particle filter can not cope with the new features of the posterior distribution since it provides no opportunity to generate new values for unknown quantities after their initial generation. Consequently, as the posterior distribution drifts away from these initial values, the particle base may degenerate to contain few distinct values of these variables. As a result, most of the particles are assigned with low weights and eliminated by the resampling process. This leads to serious sample impoverishment and then the tracking process fails.

There have been some systematic techniques proposed recently to solve the problem of sample impoverishment. One such technique is the regularized particle filter [79], which resamples from a continues approximation of the posterior density $p(x_k|z_{1:k})$, whereas the standard particle filter resamples from the discrete approximation of the posterior density. This approach is found to improve performance, which has a less rigorous derivation. An alternative solution to the same problem is the resample-move algorithm [29]. This technique uses periodic MCMC steps to diversify particles in an importance sampling-based particle filter. It does this in a rigorous manner that ensures the particles to asymptotically approximate samples from the posterior distribution. However, the traditional MCMC sampling needs a lot of iterations to converge to the target posterior distribution, which is very slow and not suitable for real-time tracking.

In this work, the adaptive MCMC sampling method is utilized to speed up the convergence rate. The adaptive Metropolis (AM) method [80] is one of the adaptive MCMC methods. In the AM method, the proposal distribution is a Gaussian distribution centered at the current state and the covariance is calculated using all of the previous states. In the proposed method which is named adaptive MCMC based particle filter, the AM method is combined with the importance sampling method in the particle filter framework. The proposed algorithm reduces the number of iterations at each time step making it suitable for real-time target tracking.

Similar to the resample-move method, the adaptive MCMC based particle filter method diversifies the particles after resampling, which reduces sample impoverishment, though it can not avoid it effectively. The introduction of MCMC to improve importance sampling suggests that MCMC alone could be used to obtain a particle filter that can effectively handle sample impoverishment [81].

A new method, named interacting MCMC particle filter, is proposed to handle sample impoverishment in this work. The particles are sampled from the target posterior distribution via direct interacting MCMC sampling method, which avoids sample impoverishment effectively.

The interacting MCMC particle filter also accelerates the MCMC convergence rate. The objective of MCMC move is to herd the particles to the area with high posterior distribution density. In the standard MCMC based particle filter method, each particle is propagated independently, however, neglecting the information from other particles. It is easier and faster to reach the high posterior density area if more information is incorporated. Inspired by the particle swarm algorithm, which locates optimal regions of complex search spaces through the interaction of individuals in a population of particles, the proposed algorithm propagates each particle based on both its historical information and the information from other particles. The proposed algorithm is named interacting MCMC particle filter since it incorporates the interaction of the particles in contrast with the traditional MCMC based particle filter. It is well known that all numerical integration approaches perform better when correlation among the components is low. In particular, MCMC algorithm converges rapidly. In the interacting MCMC particle filter method, at each time step the introduction of the interaction of particles reduces the correlation among a particle's history states, which speeds up the convergence rate.

The rest of the sections are organized as follows: The basic theory of Markov chain Monte Carlo is briefly introduced in Section 2.1.1. The adaptive MCMC based particle filter and the interacting MCMC particle filter are presented in Section 2.1.2 and Section 2.1.3 respectively.

2.1.1 Basic Theory of Markov Chain Monte Carlo Process

MCMC methods define a Markov Chain over the space of configurations X, such that the stationary distribution of the chain is equal to a target distribution $\pi(X)$. The Metropolis-Hastings (MH) algorithm [82] is one way to simulate the target distribution $\pi(X)$ from such a chain. The pseudocode for the MH algorithm in this context is as follows [69].

Algorithm 2.1: Metropolis-Hastings Algorithm

Start with an arbitrary initial configuration X_0, then iterate for $\tau = 0, \cdots, B + M$, where $B + M$ is the number of iterations at each time step.

1. Propose a new assignment X' by sampling from the proposal density function $Q(X'; X_\tau)$.

2. Sample $\rho \sim U(0,1)$, where $U(0,1)$ is a uniform distribution in the interval $(0,1)$.

3. Calculate the acceptance ratio,

$$a = \frac{\pi(X')}{\pi(X_\tau)} \frac{Q(X_\tau; X')}{Q(X'; X_\tau)}. \tag{2.1}$$

4. If $\rho \leq min\{1, a\}$, then accept move:

$$X_{\tau+1} = X', \tag{2.2}$$

   else reject move:

$$X_{\tau+1} = X_\tau. \tag{2.3}$$

It is a standard practice to discard a number of initial "burn-in" samples, say B of them, to allow the MH algorithm to converge to a stationary distribution. In the proposed algorithm, the target distribution $\pi(X)$ is chosen as the approximate posterior distribution $\hat{p}(x_k|z_{1:k})$ and at each time step k, the MH algorithm is used to generate a set of samples from $\hat{p}(x_k|z_{1:k})$.

2.1.2 Adaptive MCMC Based Particle Filter Algorithm

The slow convergence rate is a major problem associated with the traditional MCMC algorithms. Many adaptive MCMC methods have been proposed to speed up the convergence rate, and the adaptive Metropolis (AM) method [80] is one among them. In the proposed adaptive MCMC based particle filter algorithm, the AM method is combined with the importance sampling in a particle filter framework, which reduces the number of iterations at each time step making it suitable for real-time target tracking.

2.1.2.1 Adaptive Metropolis Method

The adaptive Metropolis (AM) method allows the transition kernel to adapt whenever new features of posterior distribution are encountered during the tracking process.

The definition of the AM algorithm is based on the classical random walk Metropolis algorithm. The proposal distribution $Q(\cdot|x_0, \cdots, x_{\tau-1})$ employed in the AM algorithm is a Gaussian distribution with the mean x_τ at the current point and covariance $C_\tau = C_\tau(x_0, \cdots, x_{\tau-1})$.

The crucial thing regarding the adaptation is how the covariance of the proposal distribution depends on the history of the chain. In the AM algorithm this is achieved by setting $C_\tau = s_d Cov(x_0, \cdots, x_{\tau-1}) + s_d \varepsilon I_d$ after the initial "burn-in" time, where s_d is the scaling parameter that depends only on the dimension d of the state, and $\varepsilon > 0$ is a constant which may be assigned with very small value compared to the size of S. As a basic choice for the scaling parameter the value of $s_d = (2.4^2/d)$ is adopted. Such a choice optimizes the mixing properties of the Metropolis search in the case of Gaussian targets and Gaussian proposals [83].

To begin with, an arbitrary strictly positive definite initial covariance C_0 is selected, which of course is chosen according to the best priori knowledge. An index $\tau_0 > 0$ is selected for the length of an initial period and C_τ is define as,

$$C_\tau = \begin{cases} C_0, & \tau \le \tau_0 \\ s_d Cov(x_0, \cdots, x_{\tau-1}) + s_d \varepsilon I_d, & \tau > \tau_0 \end{cases} . \tag{2.4}$$

Recalling the definition of the empirical covariance matrix determined by points $x_0, \cdots, x_t \in R^d$:

$$Cov(x_0, \cdots, x_t) = \frac{1}{t}\left(\sum_{i=0}^{t} x_i x_i^T - (t+1)\bar{x}_t \bar{x}_t^T\right), \tag{2.5}$$

where $\bar{x}_t = \frac{1}{t+1}\sum_{i=0}^{t} x^i$ and the elements $x_i \in R^d$ are considered as column vectors. For $\tau \geq \tau_0 + 1$, as per the equation (2.4) the covariance matrix C_τ satisfies the recursion formula,

$$C_{\tau+1} = \frac{\tau-1}{\tau} C_\tau + \frac{s_d}{\tau}(\tau \bar{x}_{\tau-1}\bar{x}_{\tau-1}^T - (\tau+1)\bar{x}_\tau \bar{x}_\tau^T + x_\tau x_\tau^T + \varepsilon I_d). \tag{2.6}$$

This allows to calculate C_τ without too much computational cost since the mean $\bar{x}_\tau$ also satisfies an obvious recursion formula. The pseudocode for the AM algorithm in this context is as follows:

Algorithm 2.2: Adaptive Metropolis Algorithm

Start with an arbitrary initial configuration X_0, then iterate for $\tau = 0, \cdots, B + M$.

(1) Sample $\epsilon \sim N_m(0, C_\tau)$, the normal distribution. Move the particles:

$$X' = X_\tau + \epsilon. \tag{2.7}$$

(2) Sample $\rho \sim U(0, 1)$, where $U(0, 1)$ is a uniform distribution in the interval $(0, 1)$.

(3) Calculate the acceptance ratio,

$$a = \frac{\pi(X')}{\pi(X_\tau)} \frac{Q(X_\tau; X')}{Q(X'; X_\tau)}. \tag{2.8}$$

(4) If $\rho \leq min\{1, a\}$, then accept move:

$$X_{\tau+1} = X', \tag{2.9}$$

else reject move:

$$X_{\tau+1} = X_\tau. \tag{2.10}$$

(5) Adaptation step: update the covariance of the proposal distribution depending on the history of the chain.

$$C_{\tau+1} = \frac{\tau - 1}{\tau} C_\tau + \frac{s_d}{\tau} (\tau \bar{X}_{\tau-1} (\bar{X}_{\tau-1})^T - (\tau + 1) \bar{X}_\tau (\bar{X}_\tau)^T + X_\tau (X_\tau)^T + \varepsilon I_d), \tag{2.11}$$

where,

$$\bar{X}_\tau = \frac{1}{\tau + 1} \sum_{j=0}^{\tau} X_j. \tag{2.12}$$

2.1.2.2 Adaptive MCMC Based Particle Filter Algorithm

In the proposed adaptive MCMC based particle filter algorithm, at each time step, NP particles are propagated based on the dynamic model to obtain the predicted particles. Each predicted particle is evaluated according to the likelihood function and is assigned with a weight. The predicted particles are resampled according to their corresponding weights to obtain the resampled particles. The resampled particles are then diversified through iterations of adaptive MCMC sampling procedure. The steps of the proposed algorithm are listed in the following:

Algorithm 2.3: Adaptive MCMC Based Particle Filter Algorithm

(i) Initialization: Sample x_0^i from the initial posterior distribution $p(x_0)$ and set the weights w_0^i to $\frac{1}{NP}$, $i = 1, ..., NP$

(ii) Prediction: Each particle is passed through the system model to obtain the predicted particles:

$$\hat{x}_k^i = f(x_{k-1}^i, v_{k-1}^i), \tag{2.13}$$

where v_{k-1}^i is a sample drawn from the probability density function of the system noise $p_v(v)$.

(iii) Update: Once the observation data, z_k, is measured, evaluate the importance weight of each predicted particle as per (2.14) and obtain the normalized weight for each particle as per (2.15).

$$\widetilde{w}_k^i = w_{k-1}^i p(z_k|\hat{x}_k^i) \tag{2.14}$$

$$w_k^i = \frac{\widetilde{w}_k^i}{\sum_{i=1}^{N} \widetilde{w}_k^i} \tag{2.15}$$

Thus define a discrete distribution $\{w_k^i : i = 1, \cdots, NP\}$ over $\{\hat{x}_k^i : i = 1, \cdots, NP\}$, with importance weight w_k^i associated with element $\hat{x}_k^i$ at time k. The estimate of the posterior distribution, $\hat{p}(x_k|z_{1:k})$, can be represented as,

$$\hat{p}(x_k|z_{1:k}) = \sum_{i=1}^{NP} w_k^i \delta(x - \hat{x}_k^i). \tag{2.16}$$

(v) Resampling the particles: Resample the discrete distribution $\{w_k^i : i = 1, \cdots, NP\}$ NP times to generate particles $\{x_k^j : j = 1, \cdots, NP\}$, so that for any j, $Pr\{x_k^j = \hat{x}_k^i\} = w_k^i$. Set the weights w_k^i to $\frac{1}{NP}$, $i = 1, ..., NP$.

(vi) Adaptive MCMC diversification: For each resampled particle $\{x_k^i,\ i = 1, \cdots, NP\}$, repeat the following steps ((1) $\sim$ (7)) $B + M$ times, where B is the length of burn-in period and M is the number of MCMC iterations.

(1): Initializing, set

$$\tau = 0, \tag{2.17}$$

$$\chi_{k,0}^i = x_k^i, \tag{2.18}$$

$$C_{k,0}^i = I_d. \tag{2.19}$$

(2): MH algorithm, sample $\epsilon \sim N_m(0, C_{k,\tau}^i)$. Move the particles:

$$\chi' = \chi_{k,\tau}^i + \epsilon. \tag{2.20}$$

(3): Sample $\rho \sim U(0,1)$, where $U(0,1)$ is a uniform distribution in the interval $(0,1)$.

(4): Calculate the acceptance ratio,

$$a = \frac{\hat{p}(\chi'|Z_{1:k})}{\hat{p}(\chi_{k,\tau}^i|Z_{1:k})} \frac{Q(\chi_{k,\tau}^i; \chi')}{Q(\chi'; \chi_{k,\tau}^i)}. \tag{2.21}$$

(5): If $\rho \leq min\{1, a\}$, then accept move:

$$\chi_{k,\tau+1}^i = \chi', \tag{2.22}$$

else reject move:

$$\chi_{k,\tau+1}^i = \chi_{k,\tau}^i. \tag{2.23}$$

(6): Adaptation step: update the covariance of the proposal distribution depending on the history of the chain.

$$C_{k,\tau+1}^i = \frac{\tau - 1}{\tau} C_{k,\tau}^i + \frac{s_d}{\tau}[\tau \bar{\chi}_{k,\tau-1}^i (\bar{\chi}_{k,\tau-1}^i)^T - (\tau+1)\bar{\chi}_{k,\tau}^i (\bar{\chi}_{k,\tau}^i)^T + \chi_{k,\tau}^i (\chi_{k,\tau}^i)^T + \varepsilon I_d]. \tag{2.24}$$

Where,

$$\bar{\chi}_{k,\tau}^{i} = \frac{1}{\tau+1}\sum_{j=0}^{\tau}\chi_{k,j}^{i}. \tag{2.25}$$

(7): At the end of $B+M$ iterations, obtain the diversified particles as per (2.26).

$$x_k^i = \chi_{k,B+M}^i \tag{2.26}$$

Finally, move to the prediction stage (ii).

2.1.2.3 Simulation Results and Analysis

The conventional MCMC based particle filter (resample-move algorithm) and the adaptive MCMC based particle filter are compared in the following two examples: a) a robot equipped with sonar tracks a maneuvering target and, b) a radar tracks an aircraft performing coordinated turn. In the first example, the dynamic model of the maneuvering target is represented as:

$$X_k = \Phi X_{k-1} + \Gamma[a_{k-1} + m_{k-1}(s,t)], \tag{2.27}$$

where $X_k = [px, vx, py, vy]_k^T$ is the state vector; px and vx are respectively the position and velocity of the moving object along the Cartesian frame x axis; and, py, vy along the y axis. $m_{k-1}(s,t)$ is the maneuver-induced acceleration. s and t are the start and end times of the maneuver. a_{k-1} accounts for the random acceleration of the target, which is generated from a zero mean Gaussian distribution. Φ is the transition matrix.

$$\Phi = \begin{bmatrix} 1 & \triangle T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \triangle T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.28}$$

where $\triangle T$ is time interval. Γ is a unity matrix.

The robot installed with sonar sensors is positioned at the origin of the plane. The measurement equation is as follows:

$$Z_k = h(X_k) + n_k, \tag{2.29}$$

where $Z_k = [z_1, z_2]_k$ is the observation vector. z_1 is the distance between the robot and moving object and z_2 is the bearing angle. The measurement noise $n_k = [n_{z_1}, n_{z_2}]_k$ is a zero mean Guassian white noise process with variance $R : E[n_k \; n_j] = R\delta_{kj}$, where,

$$R = \begin{bmatrix} \sigma_{z_1}^2 & 0 \\ 0 & \sigma_{z_2}^2 \end{bmatrix}. \tag{2.30}$$

(2.29) is expanded as:

$$z_{1,k} = \sqrt{(px_k - x_R)^2 + (py_k - y_R)^2} + n_{z_1,k}, \tag{2.31}$$

$$z_{2,k} = tan^{-1}(\frac{py_k - y_R}{px_k - x_R}) + n_{z_2,k}, \tag{2.32}$$

Equation (2.31) describes the changing distance between the robot and moving object. (x_R, y_R) is the position of the robot in Cartesian coordinates. Equation (2.32) describes the object's changing bearing angle.

In this example, the target considered executes a 3 leg maneuvering sequence. The target starts at location [0.5 1] in Cartesian coordinates in meters with initial velocity [3 1] (in m/s). Its trajectory is: a straight line with constant velocity between 0 and 20 s, a sharp left turn ($[-40 \; 40]$ in m/s^2) occurs at 20 s, a straight line with constant velocity between between 20 and 30 s, a sharp right turn ($[24 \; -24]$ in m/s^2) occurs at 30 s, and finally a straight line with constant velocity between 30 and 100 s. It is assumed that the dynamic model of the maneuvering target (2.27) is unknown and a simple motion model (2.33) is adopted in the two algorithms that are being compared. The simulation parameters are listed in Table 2.1.

$$X_k = \Phi X_{k-1} + v_{k-1} \tag{2.33}$$

The tracking trajectory of one successful realization performed by the two algorithms are shown respectively in Fig.2.1 and Fig.2.2. One hundred (100) Monte Carlo simulations are carried out. The simulation comparison between the two algorithms is presented in the form of the position Root Mean Square Error (RMSE) in Table 2.2. The adaptive MCMC based particle filter gained superior tracking performance than

Table 2.1: Simulation parameters

Simulation Parameter	Value
Number of particles	200
Sampling internal ($\triangle T$)	0.1 s
The variance matrix Q of process noise	$diag\{1\ 1\ 1\ 1\}$
The variance matrix R of observation noise	$diag\{0.1\ 0.01\}$
The bound of the uniform distribution (d)	$\{1\ 1\ 1\ 1\}^T$
Signal-to-noise ratio (SNR)	9.5 dB

Table 2.2: Performance comparison

	RMSE	MCMC Iteration Number	Tracking Loss Rate (%)
MCMC based PF	0.4691	5	50
Adaptive MCMC based PF	0.2759	5	10

the conventional MCMC based particle filter with the same MCMC iteration number. Also the RMSEs at each time step of the two algorithms are presented respectively in Fig.2.3 and Fig.2.4. The definition of RMSEs can be found in [45].

The tracking loss rate is defined as the ratio of the failed simulation number to the total simulation number carried out, which is used to evaluate the robustness of the algorithm. The tracking loss rate is listed in Table 2.2. The adaptive MCMC based particle filter (with tracking loss rate 10%) is more robust than the conventional MCMC based particle filter (with tracking loss rate 50%). Fig.2.5 $\sim$ Fig.2.8 show a failure tracking process performed by the conventional MCMC based particle filter algorithm. When sample impoverishment began at time step 38 (Fig. 2.8, all the particles were assigned with zero weights), the x and y tracking trajectories started to diverge at time step 38 (Fig. 2.6 and Fig. 2.7). The tracking failure is due to sample impoverishment and reducing the sample impoverishment can improve the robustness of the algorithm effectively.

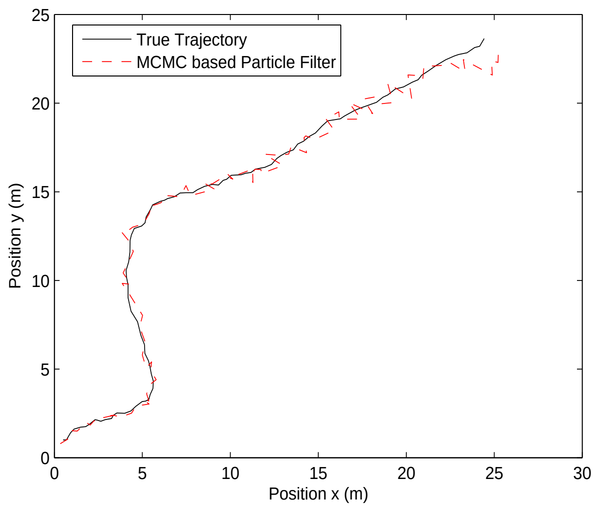

Figure 2.1: Tracking trajectory (MCMC based particle filter)

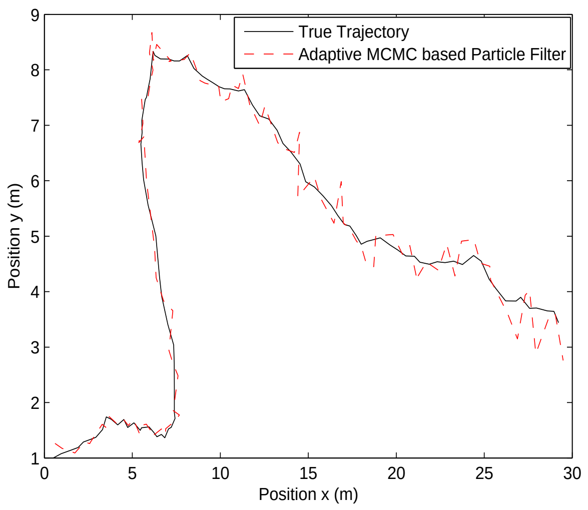

Figure 2.2: Tracking trajectory (adaptive MCMC based particle filter)

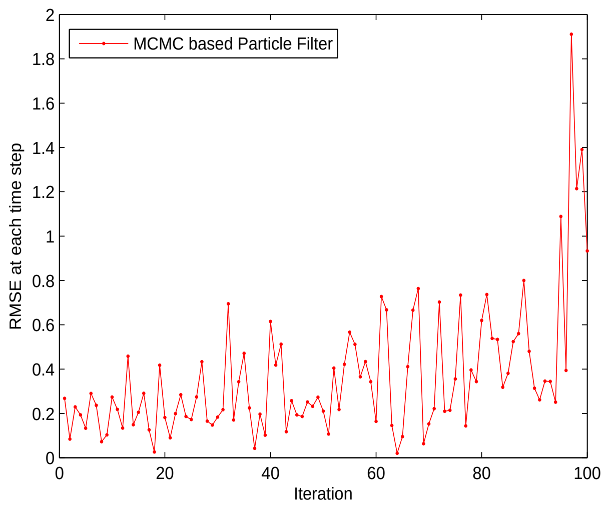

Figure 2.3: RMSE at each time step (MCMC based particle filter)

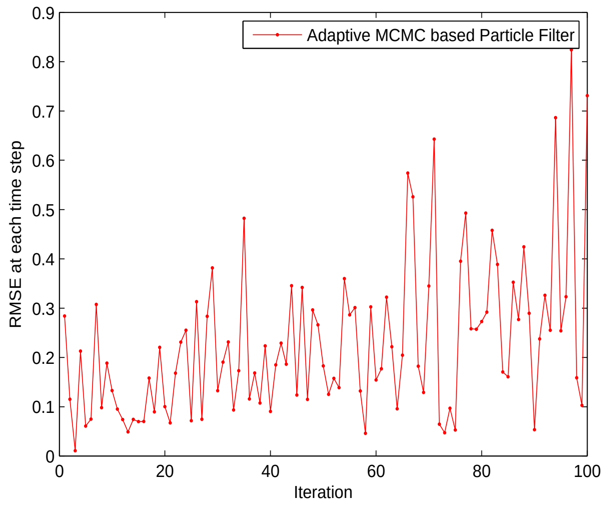

Figure 2.4: RMSE at each time step (adaptive MCMC based particle filter)

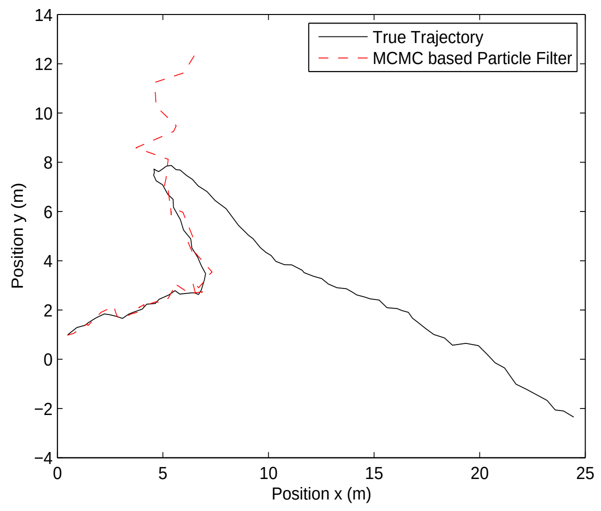

Figure 2.5: Failure tracking trajectory (MCMC based particle filter)

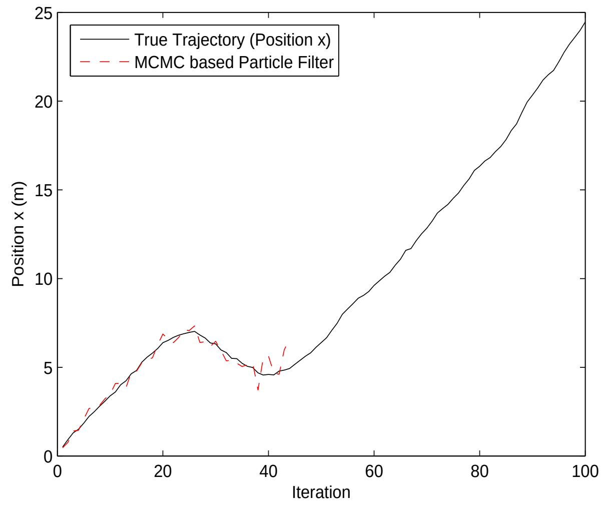

Figure 2.6: Failure tracking trajectory: position x (MCMC based particle filter)

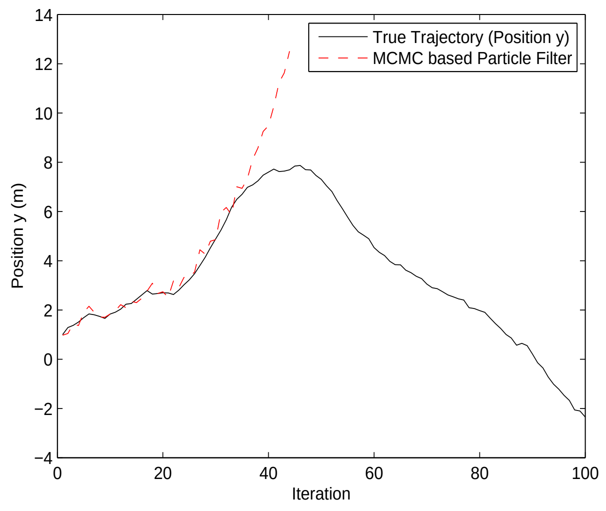

Figure 2.7: Failure tracking trajectory: position y (MCMC based particle filter)

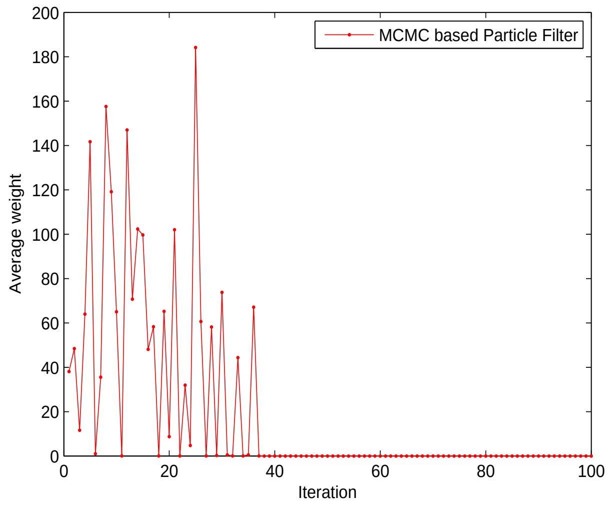

Figure 2.8: Failure tracking process: average weight (MCMC based particle filter)

The example above is basically designed for tracking systems in which the filters are uncoupled such that, for instance, tracking in the x and y directions are independent. In reality, typical target maneuvers, such as an aircraft performing a coordinated turn, produces motion that is highly correlated across the tracking directions.

In the second example, a track-while-scan (TWS) radar tracks an aircraft performing coordinated turn. The coordinated turn model considered is:

$$\begin{aligned} px_{k+1} &= px_k + \frac{sin\omega T}{\omega} vx_k - \frac{1 - cos\omega T}{\omega} vy_k, && (2.34)\\ py_{k+1} &= py_k + \frac{1 - cos\omega T}{\omega} vx_k + \frac{sin\omega T}{\omega} vy_k, && (2.35)\\ vx_{k+1} &= vx_k cos\omega T - vy_k sin\omega T, && (2.36)\\ vy_{k+1} &= vx_k sin\omega T + vy_k cos\omega T, && (2.37) \end{aligned}$$

where ω denotes the turn rate in radians per second. The measurement equations of the TWS radar are similar with those of the first example.

In this example, an aircraft executing a 3 leg maneuvering coordinate turn is considered: constant velocity, $3 \times g$ turn, $-3 \times g$ turn, and finally moving with a constant velocity. The initial velocity is about $75m/s$. No multiple dynamic models and transition probability matrix are assumed.

The tracking trajectory of the maneuvering aircraft performed by the two algorithms are shown respectively in Fig.2.9 and Fig.2.10. One hundred (100) Monte Carlo simulations are carried out. The simulation comparison between the two algorithms is presented in Table 2.3. Also the RMSE at each time step of the two algorithms are presented respectively in Fig.2.11 and Fig.2.12. The adaptive MCMC based particle filter gained superior tracking performance than the MCMC based particle filter both in accuracy and robustness (Table 2.3).

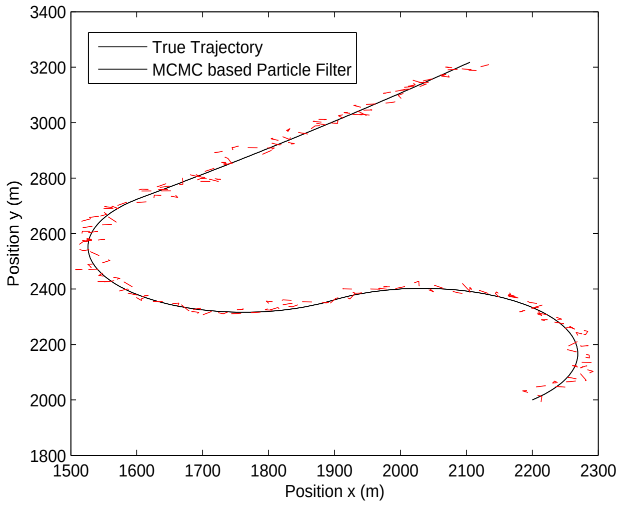

Figure 2.9: Tracking trajectory via MCMC based particle filter using 5 MCMC iterations

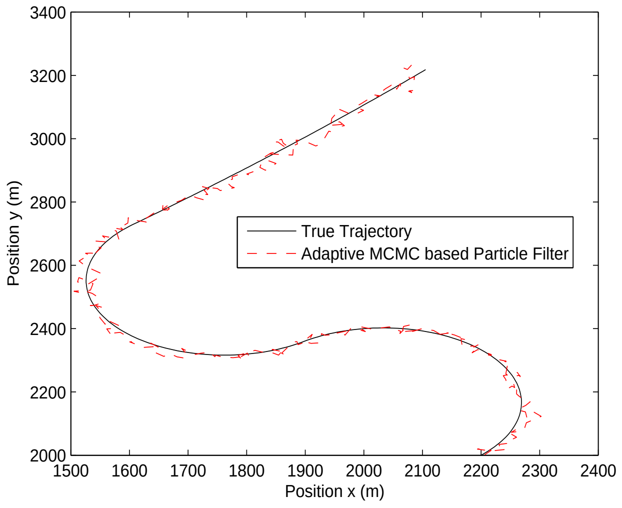

Figure 2.10: Tracking trajectory via adaptive MCMC based particle filter using 5 MCMC iterations

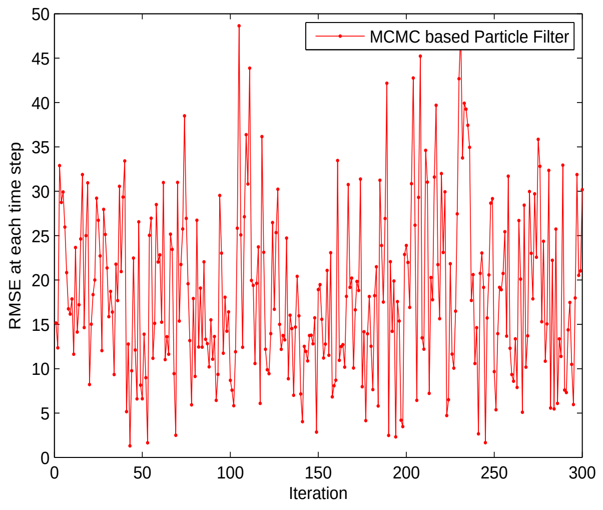

Figure 2.11: RMSE at each time step via MCMC based particle filter using 5 MCMC iterations

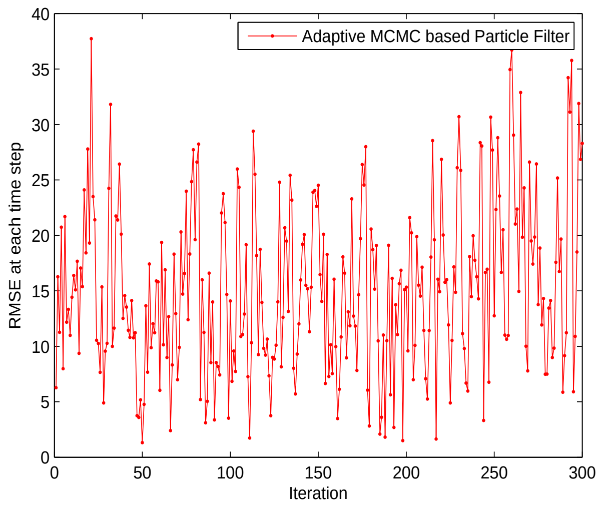

Figure 2.12: RMSE at each time step via adaptive MCMC based particle filter using 5 MCMC iterations

Table 2.3: Performance comparison

	RMSE	MCMC Iteration Number	Tracking Loss Rate (%)
MCMC based PF	20.8755	5	37
Adaptive MCMC based PF	16.9720	5	4

2.1.3 Interacting MCMC Particle Filter

Similar to resample-move method, the adaptive MCMC based particle filter diversifies particles after resampling, which reduces sample impoverishment but could not avoid it absolutely. The introduction of MCMC to improve importance sampling suggests that MCMC alone could be used to obtain a particle filter that can effectively handle the sample impoverishment problem [81]. However, for the traditional MCMC move, it is very slow to converge to the target posterior distribution. Though the AM algorithm can speed up the convergence rate, it is still slow for real-time tracking, which warrants the need for a faster and more effective method.

A new method, named interacting MCMC particle filter, is proposed to move the particles towards the target posterior distribution quickly. The objective of general MCMC move is to search the target posterior distribution via particles' moves. In the MCMC based particle filter method [29], each particle is propagated based on its previous states. During the MCMC move iterations, the trajectory of each particle is developed independently, which means that the searching is carried out by each individual particle, neglecting the information from other particles. It is reasonable that incorporating the neighborhood particles' information accelerates the search. For example, when several people search for a piece of gold in a wide area concurrently, each person looks for the gold depending on not only his previous experience but also the information from his partners (where the gold may appear most probably), and adjust the search strategy accordingly. The particle swarm algorithm finds optimal regions of complex search spaces through the interaction of individuals in a population of particles. In the proposed algorithm, during the MCMC iterations, each particle is propagated based on not only the past information but the information from other particles as well. The proposed algorithm is named as interacting MCMC particle filter since it incorporates the interaction of the particles in contrast with the traditional MCMC based particle filter.

It is well known that all numerical integration approaches perform better when correlation among the components is low. In particular, MCMC algorithm converges rapidly. In the interacting MCMC particle filter method, at each time step the introduction of the interaction of particles reduces the correlation among one particle's history states, which speeds up the MCMC convergence rate.

The basic theory of particle swarm algorithm is introduced in Section 2.1.3.1, and the proposed interacting MCMC particle filter algorithm is presented in Section 2.1.3.2. Finally, the simulation results and analysis are presented in Section 2.1.3.3.

2.1.3.1 Particle Swarm Algorithm

Particle swarm adaptation has been shown to successfully optimize a wide range of functions [84], [85], [86]. The algorithm, which is based on a metaphor of social interaction, searches a space by adjusting the trajectories of individual vectors, called "particles" as they are conceptualized as moving points in multidimensional space. Each particle is drawn stochastically toward the position of its own previous best performance, pB_i, and the position of the best previous performance of its neighbors, gB, where i denotes the *ith* particle. The algorithm in pseudocode follows [87].

Algorithm 2.4: Particle Swarm Algorithm

```
(1) Initialize population.
(2) Do,
    for $i = 1$ to Population Size,
```

$$V_i = V_i + \varphi_1(pB_i - S_i) + \varphi_2(gB - S_i), \tag{2.38}$$

$$S_i = S_i + V_i, \tag{2.39}$$

Next i,

Until termination criterion is met.

S_i represents the state vector of the ith particle, and V_i denotes the velocity of S_i. The variables φ_1 and φ_2 are random positive numbers, drawn from a uniform distribution and defined by an upper limit φ_{max}, whose value is chosen as 2 in this work.

2.1.3.2 Interacting MCMC Particle Filter Algorithm

Here a different Monte Carlo approximation of the target posterior distribution, in terms of unweighted samples, is proposed based on the interacting MCMC sampling method. In particular, the posterior distribution $p(x_{k-1}|z_{1:k-1})$ at time $k-1$ is represented as a set of NP unweighted samples $p(x_{k-1}|z_{1:k-1}) \approx \{x_{k-1}^i\}_{i=1}^{NP}$. According to the Bayesian theory, the posterior filtering distribution at time step k, $p(x_k|z_{1:k})$, can be represented as:

$$p(x_k|z_{1:k}) \approx cp(z_k|x_k) \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}. \tag{2.40}$$

Instead of importance sampling, interacting MCMC is used to sample from (2.40) at each time step. The sampling procedure results in an unweighted particle approximation for the posterior distribution $p(x_k|z_{1:k}) \approx \{x_k^i\}_{i=1}^{NP}$.

In the proposed interacting MCMC particle filter algorithm, at each time step, the particles are propagated based on the dynamic model to obtain the predicted particles. The predicted particles are then chosen as the starting points in the following interacting MCMC sampling procedure. During the interacting MCMC iterations, the proposal function based on the particle swarm algorithm is used to generate a set of samples from the target posterior distribution. The algorithm steps are listed in the following:

Algorithm 2.5: Interacting MCMC Particle Filter Algorithm

(i) Initialization: Sample x_0^i from the prior distribution $p(x_0)$ and set the weights w_0^i to $\frac{1}{NP}$, $i = 1, ..., NP$.

(ii) Prediction: Each particle is passed through the system model to obtain the predicted particles:

$$\hat{x}_k^i = f(x_{k-1}^i, v_{k-1}^i), \tag{2.41}$$

where v_{k-1}^i is a sample drawn from the probability density function of the system noise $p_v(v)$.

(iii) Update: Once the observation data, z_k, is measured, evaluate the importance weight of each predicted particle in (2.42) and obtain the normalized weight for each particle as in (2.43).

$$\widetilde{w}_k^i = w_{k-1}^i p(z_k|\hat{x}_k^i) \tag{2.42}$$

$$w_k^i = \frac{\widetilde{w}_k^i}{\sum_{i=1}^N \widetilde{w}_k^i} \tag{2.43}$$

Thus define a discrete distribution $\{w_k^i : i = 1, \cdots, NP\}$ over $\{\hat{x}_k^i : i = 1, \cdots, NP\}$, with probability mass w_k^i associated with element $\hat{x}_k^i$ at time step k. The approximate posterior distribution, $\hat{p}(x_k|z_{1:k})$, can be estimated as,

$$\hat{p}(x_k|z_{1:k}) = \sum_{i=1}^{NP} w_k^i \delta(x - \hat{x}_k^i). \tag{2.44}$$

(iv) Interacting MCMC move: For each predicted particle $\hat{x}_k^i$, $i = 1, \cdots, NP$, repeat the following steps $B + M$ times (B is the length of burn-in period and M is the number of MCMC iterations) :

(1) Initialization, set,

$$\tau = 0, \tag{2.45}$$

$$\chi_{k,0}^i = \hat{x}_k^i, \tag{2.46}$$

$$V_{k,0}^i = 0, \tag{2.47}$$

$$\xi_{k,0}^i = w_k^i, \tag{2.48}$$

where τ denotes the τth MCMC iteration.

(2) Search among the weights $\{\xi^i_{k,\tau}, i = 1, \cdots, NP\}$ to obtain the largest weight, and identify the particle corresponding to the largest wight, $gB_{k,\tau}$. For each specific particle $\chi^i_{k,\tau}$, search among its history weights, $\{\xi^i_{k,\lambda}, \lambda = 0, \cdots, \tau\}$ and obtain the particle with the largest weight, $pB^i_{k,\tau}$.

(3) For each particle $\chi^i_{k,\tau}$, calculate its velocity $V^i_{k,\tau+1}$ (2.49), and then propagate it to the next position (2.50),

$$V^i_{k,\tau+1} = V^i_{k,\tau} + \varphi_1 \times (gB_{k,\tau} - \chi^i_{k,\tau}) + \varphi_2 \times (pB^i_{k,\tau} - \chi^i_{k,\tau}), \tag{2.49}$$

$$\hat{\chi}^i_{k,\tau+1} = \chi^i_{k,\tau} + V^i_{k,\tau+1}. \tag{2.50}$$

(4) Sample $\rho \sim U(0,1)$, where $U(0,1)$ is a uniform distribution in the interval $(0,1)$.

(5) Calculate the acceptance ratio,

$$a = \frac{\hat{p}(\hat{\chi}^i_{k,\tau+1}|z_{1:k})}{\hat{p}(\chi^i_{k,\tau}|z_{1:k})} \frac{Q(\chi^i_{k,\tau}; \hat{\chi}^i_{k,\tau+1})}{Q(\hat{\chi}^i_{k,\tau+1}; \chi^i_{k,\tau})}. \tag{2.51}$$

(6) If $\rho \leq min\{1, a\}$, then accept move:

$$\chi^i_{k,\tau+1} = \hat{\chi}^i_{k,\tau+1}, \tag{2.52}$$

else reject move:

$$\chi^i_{k,\tau+1} = \chi^i_{k,\tau}. \tag{2.53}$$

(7) Calculate the weight of each particle:

$$\xi^i_{k,\tau+1} = p(z_k|\chi^i_{k,\tau+1}). \tag{2.54}$$

(8) $\tau = \tau + 1$, then move to step (2).

Finally, obtain the resampled particles,

$$x^i_k = \chi^i_{k,B+M}, i = 1, \cdots, NP, \tag{2.55}$$

and set the weights w^i_k to $\frac{1}{NP}$, $i = 1, ..., NP$.

Move to the prediction stage **(ii)**.

2.1.3.3 Simulation Results and Analysis

In this section, the interacting MCMC particle filter algorithm is used to track the maneuvering target as in Section 2.1.2.3. The direct MCMC based particle filter algorithm and the direct adaptive MCMC based particle filter algorithm, which replace the importance sampling with MCMC sampling and adaptive MCMC sampling respectively, are also carried out for comparison.

In the first example of tracking ground maneuvering target, the trajectory tracking in one simulation round performed by the three algorithms are shown respectively in Fig.2.13 $\sim$ Fig.2.15. Large position diversions exist in the trajectories tracked by the direct MCMC based particle filter algorithm and direct adaptive MCMC based particle filter algorithm (Fig. 2.13 and Fig. 2.14). The interacting MCMC particle filter gained smooth tracking trajectory (Fig.2.15). One hundred (100) Monte Carlo simulations are carried out. The simulation comparison between the three algorithms is presented in Table 2.4. The interacting MCMC particle filter has rather smaller RMSE value (0.38) than the other two algorithms with the same MCMC iteration number. This shows that the interacting MCMC sampling converged much faster than the other two algorithms. Also the RMSE at each time step of the three algorithms are presented respectively in Fig.2.16 $\sim$ Fig.2.18. The tracking loss rate is listed in Table 2.4, and all the three algorithms have zero percent tracking loss rate. The reason is that the direct MCMC sampling methods draw the particles directly from the posterior distribution without using the resampling step, which results in sample impoverishment. As a result, sample impoverishment is avoided and the robustness of the proposed algorithm is improved.

In the second example of tracking an aircraft performing coordinated turn, both the direct MCMC based particle filter and the direct adaptive MCMC based particle filter failed in tracking due to large position diversions. The tracking trajectory of one realization performed by the interacting MCMC particle filter algorithm is shown in Fig.2.19. One hundred (100) Monte Carlo simulations are carried out using interacting

Table 2.4: Performance comparison

	RMSE	MCMC Iteration Number	Tracking Loss Rate (%)
Direct MCMC based PF	3.95	5	0
Direct Adaptive MCMC based PF	1.81	5	0
Interacting MCMC PF	0.38	5	0

MCMC particle filter. The associated RMSE is 27.2044 and the RMSE at each time step is shown in Fig. 2.20.

Finally, a performance comparison between all the MCMC based particle filters (proposed in Section 2.1.2 and Section 2.1.3) for the first simulation example is given in Table 2.5. It can be seen that the adaptive MCMC based particle filter and the interacting particle filter are the two best ones among the algorithms. The former is more accurate and the latter is more robust with similar accuracy level.

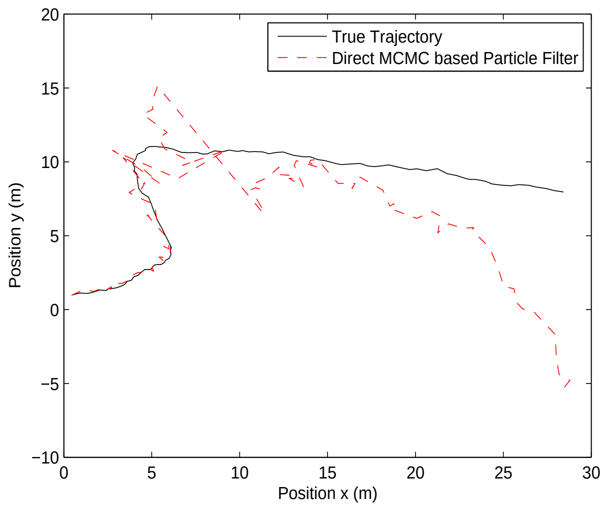

Figure 2.13: Tracking trajectory via direct MCMC based particle filter

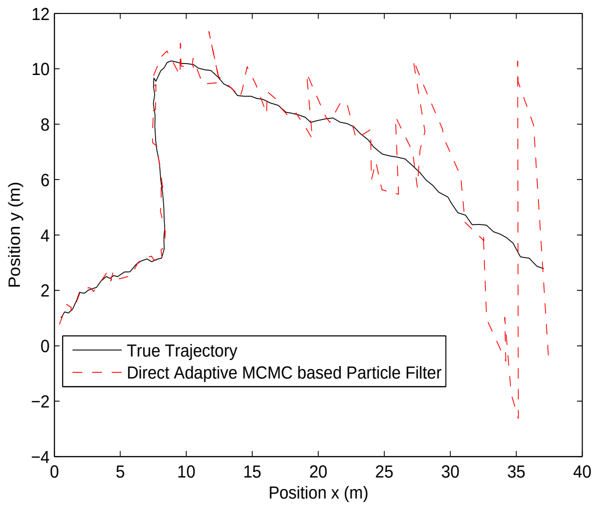

Figure 2.14: Tracking trajectory via direct adaptive MCMC based particle filter

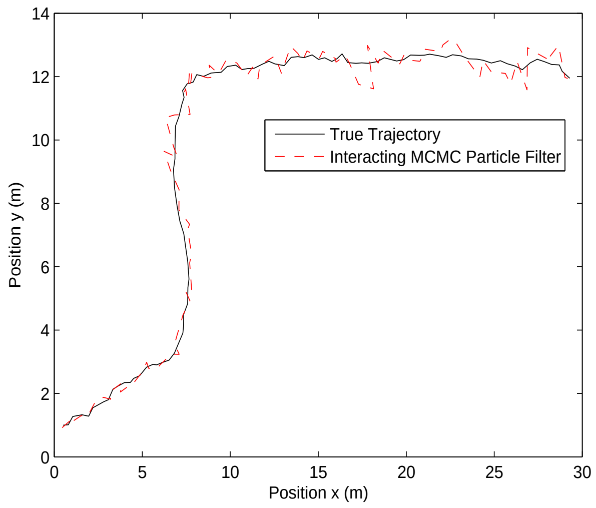

Figure 2.15: Tracking trajectory via interacting MCMC particle filter

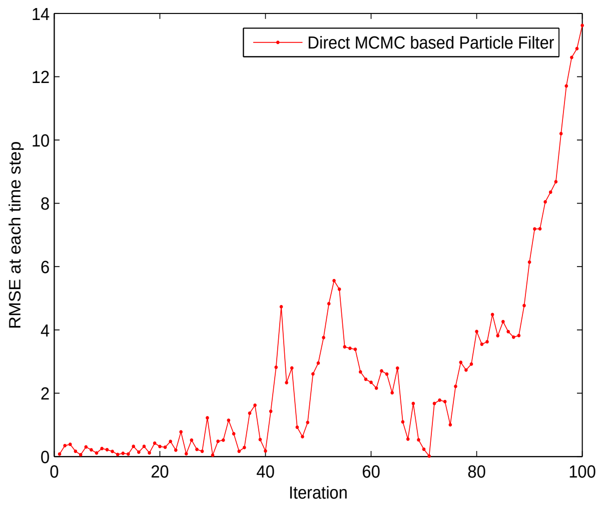

Figure 2.16: RMSE at each time step via direct MCMC based particle filter

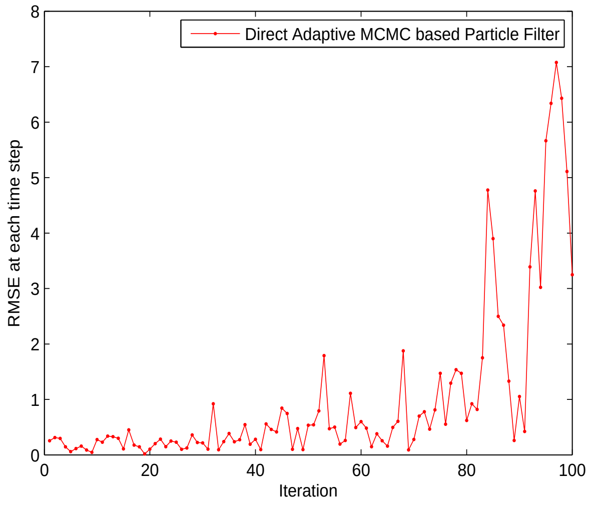

Figure 2.17: RMSE at each time step via direct adaptive MCMC based particle filter

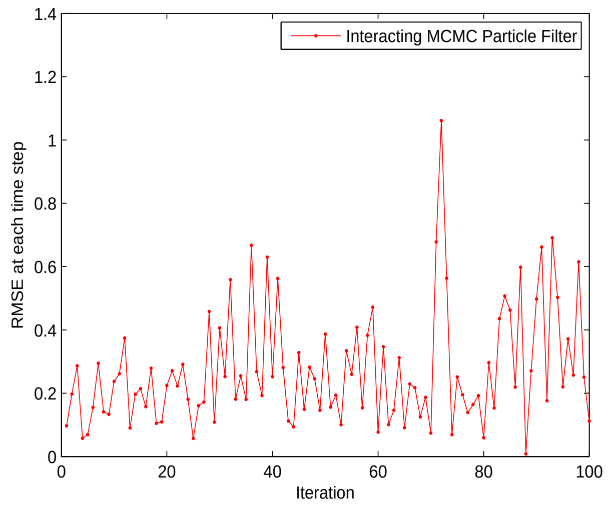

Figure 2.18: RMSE at each time step via interacting MCMC particle filter

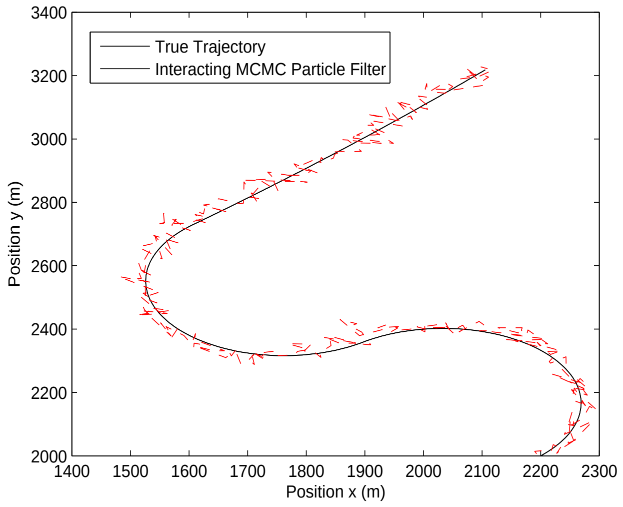

Figure 2.19: Tracking trajectory via interacting MCMC particle filter

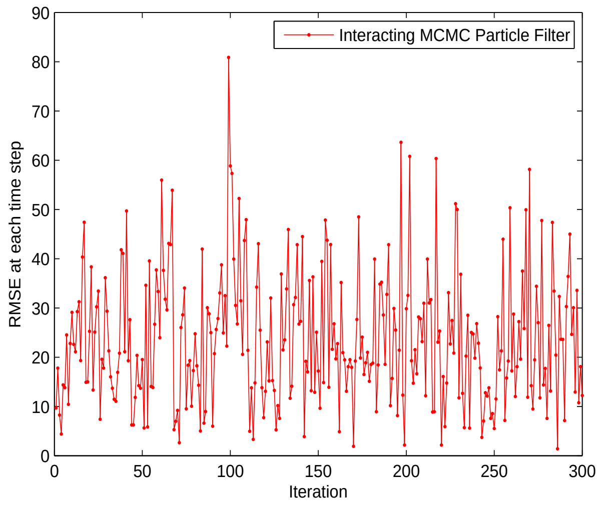

Figure 2.20: RMSE at each time step via interacting MCMC particle filter

Table 2.5: Performance comparison

	RMSE	MCMC Iteration Number	Tracking Loss Rate (%)
MCMC based PF	0.4691	5	50
Adaptive MCMC based PF	0.2759	5	10
Direct MCMC based PF	3.95	5	0
Direct Adaptive MCMC based PF	1.81	5	0
Interacting MCMC PF	0.38	5	0

2.2 Process Noise Estimation based Particle Filter

2.2.1 Introduction

A process noise estimation based particle filter algorithm is proposed to cope with the maneuvering target tracking problem in this section.

An accurate dynamic model is essential for robust tracking and for achieving real-time performance. When the tracking object deviates significantly from the learned dynamics, for example performing maneuvering movement, the estimation methods based on single fixed motion model fails. Several multiple model methods are used to deal with such tracking problems, such as interacting multiple model (IMM) methods [88, 89, 90, 91], variable dimension filter methods [92, 93] and neural fuzzy network methods [94]. In these approaches, the target maneuvers or uncertain dynamics are often described by multiple linearized models. The possible motion models and transition probability matrices are assumed known in these methods. In practice, the dynamics is hard to break up into several different motion models and the model transition probabilities are difficult to obtain. A general model is needed to cope with the wide variety of motions exhibited by a maneuvering target.

Equivalent-noise approach [95, 96, 97] is an approach, which uses one general model in maneuvering target tracking. It is assumed that the maneuver effect can be modeled by (part of) a white or colored noise process sufficiently well. The statistics of the equivalent noise are non-stationary in general. This fundamental assumption converts the problem of maneuvering target tracking to that of state estimation in the presence of non-stationary process noise. Numerous techniques have been developed for such state estimation problems in stochastic systems research over the past several decades, in particular, from late $1960's$ to early $1980's$, and adaptive Kalman filter [32, 33, 98, 34] is the most popular one among them. However, almost all of the approaches are limited to linear systems.

In this work, the equivalent-noise approach is extended to the nonlinear and non-Gaussian system. Kalman filter is not appropriate since it is based on the linear system assumption and the linear solution is of limited power. Particle filter methods are chosen as the estimation methods for their simplicity, flexibility and ease of implementation, and especially for their ability to deal with nonlinear and non-Gaussian estimation problem, which is a challenging one in maneuvering target tracking applications.

In standard particle filter algorithm, it is assumed that the process noise distribution is stationary and known. However, the noise statistics are often unknown or at least not known perfectly. In this case, the standard particle filter may yield poor results or even diverge if it uses erroneous noise statistics. It is important to identify the noise statistics. In this thesis, the terminology "noise statistics" is used to refer to the process noise distribution.

In the literature, only a few works utilize the process noise identification problem in particle filter. In [99], several different known models for the process noise are considered for suitability in tracking a target which may perform a rate-limited turn. In [100], the authors assume that the process noise is distributed as additive Gaussian distribution with fixed unknown covariance matrices. These fixed unknown covariance matrices are marginalized out and then the sequential processing is carried out on the state variables. In [101], the process noise is modeled as a first-order autoregressive (AR) system excited by a zero mean Gaussian process. The closed-form representation of the system is rendered tractable and solved iteratively by dynamically sampling the state space. More recently, Gaussian approximation methods are used to deal with non-Gaussian noises. The posterior distribution of the target state is approximated by single Gaussians [102] and weighted Gaussian mixtures [103]. In [102], the underlining assumption is that the predictive and filtering distributions are approximated as Gaussians. When dealing with a nonlinear system with process noise whose distribution may vary widely, the posterior distribution may not be approximated properly by only Gaussians. In [103], the posterior distribution of the target state is not restricted to single Gaussian assumption and the method can deal with multi-model posterior distribution more effectively than [102]. With non-Gaussian noise approximated by Gaussian mixtures, the non-Gaussian noise models are approximated by banks of Gaussian noise models. The noise is assumed in additive form and a number of Gaussian models are required to implement the algorithm, which seems very complex.

The novelty of the proposed method is that the posterior distribution of process noise is not parametrically given and/or a priori fixed, but dynamically approximated using the particle filter algorithm. The process noise is modeled as a dynamic system and the state vector of the noise system is chosen as the noise vector. At the beginning of each time step, a set of process noise samples are drawn from a uniform distribution, which is noninformative and assumed as the prior distribution of the process noise system. The process noise samples are evaluated by the likelihood function including current measurements and are assigned with corresponding weights. The posterior distribution of the process noise system is approximated by the process noise samples and their associated weights. A new set of process noise samples are then generated from the approximate posterior distribution of process noise at the resampling stage. A standard particle filter for state estimation is then run using the new distributed process noise samples.

The proposed algorithm resembles the auxiliary particle filter [31] since both extend the trajectories of the particles based on the information from current observations. When the process noise is large, the performance of the auxiliary particle filter degrades [30]. In comparison, the proposed algorithm is robust to process noise variation.

The equivalent-noise approach is briefly introduced in Section 2.2.2 and, in Section 2.2.3 the basic theory and procedure of particle filter for state estimation are provided.

The proposed algorithm is introduced in Section 2.2.4. In Section 2.2.5 the proposed algorithm is compared with the IMM algorithm, which is the most popular algorithm in maneuvering target tracking.

2.2.2 Equivalent-noise Approach

Equivalent-noise approach is a popular approach in maneuvering target tracking. Almost all types of target motion can be described by the following state-space model,

$$x_k = f(x_{k-1}, u_{k-1}, v^*_{k-1}), \tag{2.56}$$

where x is the state, u is the maneuver acceleration, and v^* is the process noise. In the equivalent-noise approach, the basic assumption is that the maneuver effect can be modeled by (part of) a white or colored noise process sufficiently well. In other words, it is assumed that the above equation that describes target motions can be simplified to,

$$x_k = f(x_{k-1}; v_{k-1}), \tag{2.57}$$

with an adequate accuracy, where v is equivalent noise that quantifies the error of this model in describing the target motions, in particular, maneuvers. Of course, the statistics of this noise v, non-stationary in general, are not known. Valid or not, this fundamental assumption converts the problem of maneuvering target tracking to that of state estimation in the presence of non-stationary process noise with unknown statistics.

Numerous techniques have been developed for such state estimation problems in stochastic systems research over the past several decades, in particular, from late $1960's$ to early $1980's$. Almost all of them are limited to linear systems, assuming that the system dynamics can be described by,

$$x_k = Fx_{k-1} + \Gamma v_{k-1}, \tag{2.58}$$

with noise v of unknown statistics. Traditionally, the state estimation using a linear system of observations in white noise is considered as an essential part of what is known as adaptive Kalman filtering.

In this work, the equivalent-noise approach is extended to the nonlinear and non-Gaussian system. Kalman filter is not appropriate since it is based on the linear system assumption and the linear solution is of limited power. Particle filter methods are chosen as the estimation methods for their simplicity, flexibility and ease of implementation, and especially for their ability to deal with nonlinear and non-Gaussian estimation problem, which is a challenging one in maneuvering target tracking applications.

2.2.3 Basic Theory of Particle Filter

The objective of tracking is to recursively estimate x_k from a sequence of measurements up to time step k, $z_{1:k} = \{z_1, z_2, \cdots, z_k\}$. The observation model is described as,

$$z_k = h(x_k, n_k), \tag{2.59}$$

where h is a possibly nonlinear function. n_k is the observation noise with zero mean Gaussian distribution. From the Bayesian perspective, the tracking problem is to recursively calculate the posterior distribution $p(x_k|z_{1:k})$.

In this work, particle filter is considered to solve the state estimation problem due to its ability to tackle the non-linear and non-Gaussian systems. The posterior distribution $p(x_k|z_{1:k})$ is approximated by a set of particles with associated weights. The detailed introduction to particle filter algorithm can be found in [30]. The procedures of standard particle filter are listed in the following:

Algorithm 2.6: Standard Particle Filter

(i) Initialization: Sample initial particles $\{x_0^i, i = 1, ..., NP\}$ from the initial posterior distribution $p(x_0)$ and set the weights w_0^i to $\frac{1}{NP}$, $i = 1, ..., NP$. NP is the number of particles.

(ii) Prediction: Particles at time step $k-1$, $\{x_{k-1}^i, i = 1, ..., NP\}$, are passed through the system model (2.57) to obtain the predicted particles at time step k, $\{\hat{x}_k^i, i = 1, ..., NP\}$:

$$\hat{x}_k^i = f(x_{k-1}^i, v_{k-1}^i), \tag{2.60}$$

where v_{k-1}^i is a sample drawn from the probability density function of the system noise $p_v(v)$.

(iii) Update: Once the observation data, z_k, is measured, evaluate the importance weight of each predicted particle (2.61) and obtain the normalized weight for each particle (2.62).

$$\tilde{w}_k^i = p(z_k|\hat{x}_k^i) \tag{2.61}$$

$$w_k^i = \frac{\tilde{w}_k^i}{\sum_{i=1}^{NP} \tilde{w}_k^i} \tag{2.62}$$

Obtain $\tilde{x}_k$, the estimate of the state at time step k as,

$$\tilde{x}_k = \sum_{i=1}^{NP} w_k^i \hat{x}_k^i. \tag{2.63}$$

(iv) Resample : Resample the discrete distribution $\{w_k^i : i = 1, \cdots, NP\}$ NP times to generate particles $\{x_k^j : j = 1, \cdots, NP\}$, so that for any j, $Pr\{x_k^j = \hat{x}_k^i\} = w_k^i$. Set the weights w_k^i to $\frac{1}{NP}$, $i = 1, ..., NP$, and move to Stage (ii).

In the above algorithm it is assumed that the process noise samples are drawn from a known posterior distribution (Stage (ii)). When the process noise distribution is unknown and varying due to the uncertain dynamics, an estimation procedure for the process noise should be performed before propagating the particles (Stage (ii)).

2.2.4 Process Noise Identification

In this section, a novel method is proposed for process noise identification. The process noise is modeled as a dynamic system. The noise vector v_{k-1} is chosen as the state of the noise system. The observation vector is z_k, which is same as the dynamic system (2.59). The observation equation is defined in (2.64),

$$\begin{aligned} z_k &= h(x_k, n_k) \\ &= h[f(x_{k-1}, v_{k-1}), n_k]. \end{aligned} \tag{2.64}$$

Since there is no information about the prior distribution of the process noise system, the prior distribution is assumed as a uniform distribution $U(-d, d)$. d is the known process noise bound accounting for the maximin uncertain dynamics. At each time step, a set of process noise samples $\{\hat{v}^j_{k-1}, j = 1, \cdots, NP\}$ are drawn from the uniform distribution, where NP is the number of process noise samples, which is equal to the number of particles.

According to particle filter algorithm, the process noise samples $\{\hat{v}^j_{k-1}, j = 1, \cdots, NP\}$ are assigned with corresponding intermediate weights $\{iw^j_k, j = 1, \cdots, NP\}$, which are obtained from the likelihood function (2.65).

$$iw^j_k = p(z_k|\hat{v}^j_{k-1}) \tag{2.65}$$

The posterior distribution of the process noise is approximated by the process noise samples and their associated weights:

$$p(v_{k-1}|z_{1:k}) \approx \Sigma^{NP}_{j=1} iw^j_k \cdot \delta(v_{k-1} - \hat{v}^j_{k-1}). \tag{2.66}$$

To calculate the noise sample weight iw^j_k, the likelihood function $p(z_k|\hat{v}^j_{k-1})$ is expanded based on the resampled state particles at time step $k-1$, $\{x^i_{k-1}, i = 1, \cdots, NP\}$.

$$p(z_k|\hat{v}^j_{k-1}) = \Sigma^{NP}_{i=1} p(z_k|x^i_{k-1}, \hat{v}^j_{k-1}) p(x^i_{k-1}|\hat{v}^j_{k-1}). \tag{2.67}$$

Since x^i_{k-1} and $\hat{v}^j_{k-1}$ are independent, $p(x^i_{k-1}|\hat{v}^j_{k-1}) = p(x^i_{k-1})$.

The ith resampled particle at time step $k-1$, x^i_{k-1}, satisfies $Pr\{x^i_{k-1} = \hat{x}^n_{k-1}\} = w^{n(i)}_{k-1}$, where $\hat{x}^n_{k-1}$ is the particle before resampling at time step $k-1$ and $w^{n(i)}_{k-1}$ is its associated normalized weight. The probability of $p(x^i_{k-1})$ is,

$$p(x^i_{k-1}) = w^{n(i)}_{k-1}. \tag{2.68}$$

To calculate $p(z_k|x^i_{k-1}, \hat{v}^j_{k-1})$, define $\mu^{i,j}_k$ as the intermediate particle,

$$\mu^{i,j}_k = f(x^i_{k-1}, \hat{v}^j_{k-1}), \tag{2.69}$$

and expand $p(z_k|x^i_{k-1}, \hat{v}^j_{k-1})$ based on $\mu^{i,j}_k$,

$$p(z_k|x^i_{k-1}, \hat{v}^j_{k-1}) = \Sigma^{NP}_{p=1}\Sigma^{NP}_{q=1} p(z_k|\mu^{p,q}_k, x^i_{k-1}, \hat{v}^j_{k-1})p(\mu^{p,q}_k|x^i_{k-1}, \hat{v}^j_{k-1}). \tag{2.70}$$

Since x^i_{k-1} and $\hat{v}^j_{k-1}$ are known, and $\mu^{p,q}_k$ is obtained from a purely deterministic relationship in (2.69), we obtain,

$$p(\mu^{p,q}_k|x^i_{k-1}, \hat{v}^j_{k-1}) = \begin{cases} 1 & p = i \;\; and \;\; q = j \\ 0 & p \neq i \;\; or \;\; q \neq j \end{cases}, \tag{2.71}$$

and,

$$p(z_k|x^i_{k-1}, \hat{v}^j_{k-1}) = p(z_k|\mu^{i,j}_k). \tag{2.72}$$

Combining (2.68) and (2.72) with (2.67) results in,

$$p(z_k|\hat{v}^j_{k-1}) = \sum_{i=1}^{NP} p(z_k|\mu^{i,j}_k) w^{n(i)}_{k-1}. \tag{2.73}$$

At each time step, the predicted process noise samples are drawn from a uniform distribution. Each predicted process noise sample $\hat{v}^j_{k-1}$ is evaluated and assigned its corresponding weight iw^j_k in (2.65). The weights $\{iw^j_k : j = 1, \dots, NP\}$ are then normalized. The resampling procedure is then used to re-distribute the predicted process noise samples: the discrete distribution $\{iw^j_k : j = 1, \cdots, NP\}$ is resampled NP times to generate samples $\{v^i_{k-1} : i = 1, \cdots, NP\}$, so that for any i, $Pr\{v^i_{k-1} = \hat{v}^j_{k-1}\} = iw^j_k$. From the resampling process, the predicted process noise samples with large weights are duplicated while the samples with small weights are eliminated.

The standard particle filter procedure for state estimation follows next. The predicted particles $\{\hat{x}^i_k, : i = 1, \dots, NP\}$ are then obtained based on the new process noise samples $\{v^i_{k-1} : i = 1, \dots, NP\}$ through the dynamic model. The predicted particles are updated and resampled as in the conventional particle filter algorithm.

The complete algorithm including the process noise estimation and state estimation parts is summarized below:

Algorithm 2.7: Process Noise Estimation based Particle Filter

(i) At time step $k-1$, draw process noise samples $\{\hat{v}_{k-1}^{j} : j = 1, \ldots, NP\}$ from a uniform distribution $U(-d, d)$.

(ii) Calculate the intermediate particles $\{\mu_k^{i,j} : i = 1, \cdots, NP; j = 1, \cdots, NP\}$ according to (2.69).

(iii) Calculate the process noise sample weights $\{iw_k^j : j = 1, \cdots, NP\}$ as per (2.73) and normalize each weight.

(iv) Resample the discrete distribution $\{iw_k^j : j = 1, \cdots, NP\}$, NP times to generate the new process noise samples $\{v_{k-1}^i : i = 1, \cdots, NP\}$, so that for any i, $Pr\{v_{k-1}^i = \hat{v}_{k-1}^j\} = iw_k^j$. Set the weights iw_k^j to $\frac{1}{NP}$, $i = 1, ..., NP$.

(v) Obtain the predicted particles at time step k from the new process noise samples as per (2.60) (same as Stage (ii), Section 2.2.3).

(vi) Calculate the un-normalized and normalized particle weights based on (2.61) and (2.62), and obtain the estimate of the state as per (2.63) (same as Stage (iii), Section 2.2.3).

(vii) Generate new particles through resampling the discrete distribution of particle weights (same as Stage (iv), Section 2.2.3). Then move to Stage (i).

Simplification of the Proposed Algorithm

In the proposed algorithm, at each iteration, $NP * NP$ intermediate particles are calculated through the permutation of particles and process noise samples in (2.69) and evaluated as per (2.72). This increases the computation burden and the algorithm runs slowly compared to the conventional particle filter and auxiliary particle filter, which are based on NP particles. It is observed that at each time step, after resampling the particles focus in some smaller area and a large portion of particles has the same value. To simplify the algorithm, it is assumed that the particles are less variable compared with the process noise samples. In (2.69), particles $\{x^i_{k-1}, i = 1, \cdots, NP\}$ are replaced by the estimate of the state at time step $k-1$, $\tilde{x}_{k-1}$,

$$\tilde{x}_{k-1} = \sum_{i=1}^{NP} w^i_{k-1} x^i_{k-1}. \tag{2.74}$$

(2.69) is then reduced as:

$$\mu^j_k = f(\tilde{x}_{k-1}, \hat{v}^j_{k-1}), \tag{2.75}$$

and $p(z_k|\hat{v}^j_{k-1})$ is expanded directly on μ^j_k,

$$p(z_k|\hat{v}^j_{k-1}) = \Sigma^{NP}_{\lambda=1} p(z_k|\mu^\lambda_k) p(\mu^\lambda_k|\hat{v}^j_{k-1}). \tag{2.76}$$

Similar to (2.71), we can obtain,

$$p(\mu^\lambda_k|\hat{v}^j_{k-1}) = \begin{cases} 1 & \lambda = j \\ 0 & \lambda \neq j \end{cases}, \tag{2.77}$$

which leads to,

$$p(z_k|\hat{v}^j_{k-1}) = p(z_k|\mu^j_k). \tag{2.78}$$

In the simplified version of the proposed algorithm, the number of intermediate particles is reduced to NP, which reduces the computation burden and increases the computing speed. More importantly, the performance of the algorithm with the simplification procedure is on par with that of the complex version without simplification, which is verified through simulation.

2.2.5 Simulation Results for Maneuvering Target Tracking

The simulation study using nearly coordinate turn model (2.34~2.37) is performed. The maneuvering target tracking is done by setting up a 2D flight path in $x-y$ plane, which is similar to the path considered in [104]. The target starts at location $[-310\ 310]$ in Cartesian coordinates in meters with initial velocity (in m/s) $[10\ -400]$. The trajectory is considered: a straight line with constant velocity between 0 and 17 s, a coordinated turn (0.09 rad/s) between 17 and 34 s, a straight line with constant velocity between 34 and 51 s, a coordinated turn (0.09 rad/s) between 51 and 68 s, and a straight line with constant velocity between 68 and 85 s.

In the particle filter based process noise identification method, one general model (2.79) is adopted during the whole tracking process.

$$X_k = \Phi X_{k-1} + \Gamma v_{k-1}, \tag{2.79}$$

$$\Phi = \begin{bmatrix} 1 & \Delta T & \Delta T^2/2 & 0 & 0 & 0 \\ 0 & 1 & \Delta T & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta T & \Delta T^2/2 \\ 0 & 0 & 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.80}$$

$$\Gamma = I_{6\times 6}, \tag{2.81}$$

where Φ is the transition matrix and ΔT is the sample interval. $X_k = [px, vx, ax, py, vy, ay]_k^T$ is the state vector; px, vx and ax denote respectively the position, velocity and acceleration of the moving object along the x axis of Cartesian frame; and, py, vy and ay along the y axis. The equivalent process noise, $v_{k-1} = [v_{px}, v_{vx}, v_{ax}, v_{py}, v_{vy}, v_{ay}]_{k-1}^T$, with unknown statistics is required to be identified. The bound of the process noise (d), which accounts for the uncertain dynamics, is chosen as $\{20\ m, 20\ m/s, 10\ m/s^2, 20\ m, 20\ m/s, 10\ m/s^2\}$. The number of the process noise samples is equal to the number of particles, which is set as 500. The algorithm is initialized with Gaussians around the initial state of the true target.

A track-while-scan (TWS) radar is positioned at the origin of the plane. The measurement equation is as follows:

$$Z_k = h(X_k) + n_k, \tag{2.82}$$

where $Z_k = [z_1, z_2]_k$ is the observation vector. z_1 is the distance between the radar and the target, and z_2 is the bearing angle. The measurement noise $n_k = [n_{z_1}, n_{z_2}]_k$

is a zero mean Guassian white noise process with standard deviations of 20 m (σ_{z_1}) and 0.01 rad (σ_{z_2}). The sampling interval is $\Delta T = 1s$.

A comparative study is made using the traditional tracking method based on a IMM-filter consisting of three extended Kalman filter (EKF) with different motion models. The three motion models considered are nearly constant velocity model, Wiener process acceleration model (nearly constant acceleration motion) and Wiener process acceleration model (model with large acceleration increments). The details regarding these models may be found in [104]. The initial model probabilities and the mode switching probability matrix are set the same values as in [104].

The simulation results are obtained from 100 Monte Carlo runs. Fig. 2.21 and Fig. 2.22 show one realization respectively performed by the IMM filter and the proposed method. The root mean-square error (RMSE) in position at each time step for the two methods are respectively shown in Fig. 2.23 and Fig. 2.24. The performance of the two methods are also compared via global RMSE (in position), tracking loss rate and executing time, which are listed in Table. 2.6. To assess the computational requirements of the two methods, the CPU time needed to execute one time step in MATLAB 7.1 on a 3 GHz (Mobile) Pentium IV operating under Windows 2000 is computed.

From the simulation results, it can be seen that the proposed method gains a 53% increase in accuracy (RMSE) and 6% increase in robustness (tracking loss rate) compared to the IMM filter, with computing time per time step within the limits of practically realisable systems. Moreover, the proposed method needs neither the possible multiple motion models nor the transition probability matrices, which are assumed as known in the IMM filter. As a result, the proposed method is a more general method for maneuvering target tracking.

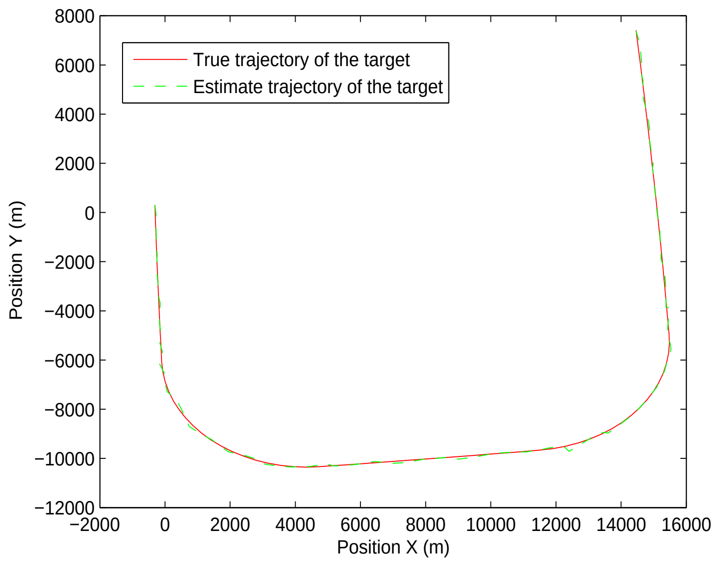

Figure 2.21: True and estimate trajectories of the single maneuvering target using IMM method

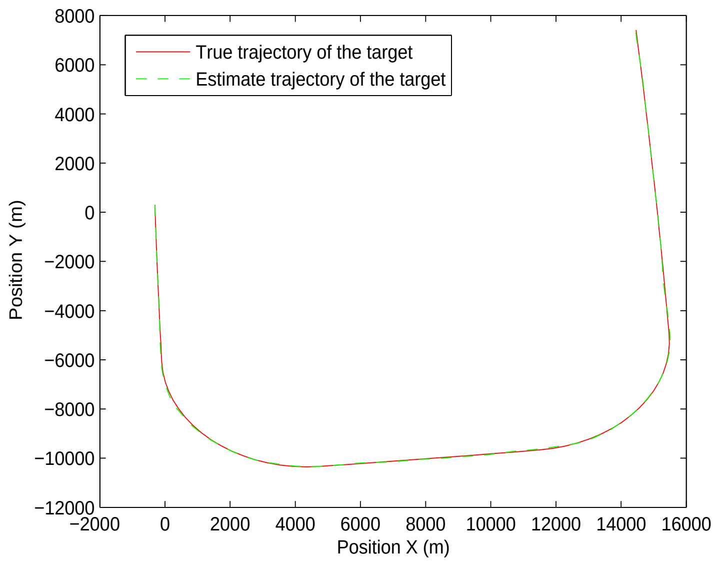

Figure 2.22: True and estimate trajectories of the single maneuvering target using particle filter based process noise identification method

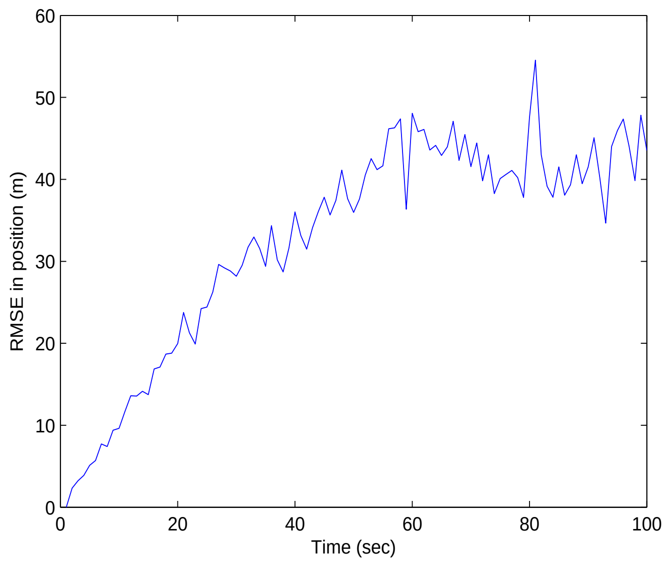

Figure 2.23: RMSE in position using IMM method

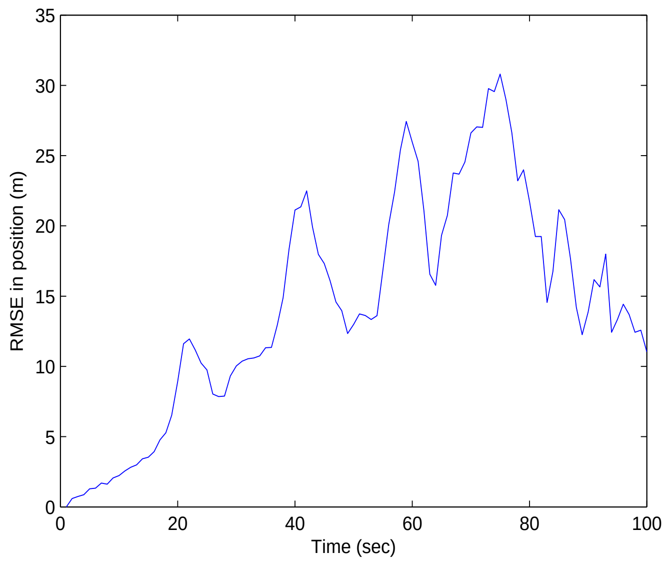

Figure 2.24: RMSE in position using particle filter based process noise identification method

Table 2.6: Performance Comparison

	RMSE (m)	Executing Time (s)	Tracking Loss Rate
IMM Method	35.2695	0.0239	6%
Particle Filter Based Process Noise Identification Method	16.4747	0.4503	0

2.3 Conclusions

In this chapter, two methods, MCMC based particle filter and process noise estimation based particle filter, are proposed to tackle the maneuvering target tracking problem.

In the MCMC based particle filter methods, the wide variation of the maneuvering movement is tracked by moving the particles towards the target posterior distribution area via MCMC sampling. In order to speed up the convergence rate, a new method named adaptive MCMC based particle filter method, which is a combination of the adaptive Metropolis (AM) method and the importance sampling method, is proposed. Furthermore, another new method named interacting MCMC particle filter is proposed to avoid sample impoverishment induced by maneuvering movement, in which the importance sampling is replaced with interacting MCMC sampling. The sampling method is named interacting MCMC sampling since it incorporates the interaction of particles. The interacting MCMC sampling also speeds up the convergence rate effectively compared with the traditional MCMC sampling method.

The process noise estimation based particle filter method is utilized to estimate the process noise which is unknown and has a varying distribution. The proposed algorithm is compared with the IMM algorithm on a synthetic problem of maneuvering target tracking.

Chapter 3

Particle Filter Based Multiple Target Tracking

In recent years, particle filter has been introduced to estimate non-linear and non-Gaussian dynamic process for multiple target tracking. Particle filter has been applied with great success to different fields of multiple target tracking including computer vision [23, 62], mobile robot localization [63, 64] and air traffic control [65, 66]. The particle filter based multiple target tracking methods can be divided into five categories (introduced in Section 1.4).

In this section, two different algorithms based on particle filter are presented for multiple target tracking. The first, which we refer to as the particle filter based multi-scan joint probabilistic data association (MS-JPDA) filter, is an extension of the single scan JPDA methods proposed in [63, 73, 78]. Similar to [78], each of the tracking targets is assigned with a corresponding particle filter. The distribution of interest is the marginal filtering distribution for each of the targets, which is approximated with particles. In contrast to the single scan JPDA methods, the MS-JPDA filter examines the joint association hypothesis in a multi-scan sliding window and calculates the posterior marginal probability based on the multi-scan joint association hypothesis. Compared with the single scan JPDA methods, the multi-scan JPDA method uses richer information, which results in better estimated probabilities.

The second method, named as multi-scan mixture particle filter, applies the particle filter method directly in the multiple target tracking process and avoids the data association problem. The posterior distribution of the target state is a multi-mode distribution and each mode corresponds to either the target or the clutter. In order to distinguish the targets from the clutters, multiple scan information is incorporated. Moreover, to deal with the new target appearance problem, new particles are sampled from the likelihood model (according to the most recent measurements) to detect the new modes appearing at each time step.

This chapter is organized as follows. The multi-scan JPDA filter is introduced in Section 3.1 and the multi-scan mixture particle filter algorithm is introduced in Section 3.2. The conclusions are drawn in Section 3.3.

3.1 Particle Filter Based Multi-scan JPDA Algorithm

3.1.1 Multiple Target Tracking Model

The number of targets (M) to be tracked is assumed as fixed and known. Each target is parameterized by a state $x_{m,k}$, where m denotes the mth target and k denotes time step k. The combined state, $x_k = (x_{1,k}, \cdots, x_{M,k})$, is the concatenation of the individual target states. The individual targets are assumed to evolve independently according to Markovian dynamic models $p_m(x_{m,k}|x_{m,k-1})$. The observation vector collected at time step k is denoted by $\{z_{j,k}, j = 1, \cdots, N_k\}$, where N_k is the number of measurements at time step k. The N_k measurements include both the target measurements and clutter measurements. NC_k is defined as the number of clutter measurements, and NT_k as the number of target measurements ($N_k = NC_k + NT_k$).

The key question of data association is how to assign the individual measurements to the respective targets. In the JPDA framework, a joint association event λ is defined based on the measurement to target association hypothesis. The elements of the association vector $\lambda = (\lambda_1 \cdots \lambda_{N_k})$ are given by,

$$\lambda_j = \begin{cases} 0 & IF\ measurement\ j \\ & is\ due\ to\ clutter \\ m \in \{1 \cdots M\} & IF\ measurement\ j \\ & stems\ from\ target\ m. \end{cases} \tag{3.1}$$

The JPDA computes β_{jm}, the posterior probability that the jth measurement is associated with the mth target, by summing over the probabilities of the corresponding joint association events, i.e.,

$$\beta_{jm} = p(\lambda_{j,k} = m | z_{1:k}) = \sum_{\{\lambda_k \in \Lambda_k \ : \ \lambda_{j,k} = m\}} p(\lambda_k | z_{1:k}), \tag{3.2}$$

where Λ_k is the set of all valid joint measurement to target association events. λ_k denotes a joint association event at time step k and $\lambda_{j,k}$ denotes the jth association variable of λ_k.

The posterior probability for the joint association event can be expressed by (3.3),

$$\begin{aligned} p(\lambda_k | z_{1:k}) &\propto p(z_k | \lambda_k, z_{1:k-1}) p(\lambda_k | z_{1:k-1}) \\ &\propto p(z_k | \lambda_k, z_{1:k-1}) p(\lambda_k), \end{aligned} \tag{3.3}$$

where the conditioning of λ_k on the measurements $z_{1:k-1}$ has been eliminated and $p(\lambda_k)$, the joint association prior is assumed as uniform distribution. $p(z_k|\lambda_k, z_{1:k-1})$ is expressed based on the likelihood function,

$$\begin{aligned} & p(z_k | \lambda_k, z_{1:k-1}) \\ & = \textstyle\prod_{j \in I_{0,k}} p_c(z_{j,k}) \prod_{j=1}^{NT_k} p(z_{j,k} | x_{\lambda_{j,k},k}) \\ & = (V)^{-NC_k} \textstyle\prod_{j=1}^{NT_k} p(z_{j,k} | x_{\lambda_{j,k},k}), \end{aligned} \tag{3.4}$$

where, the likelihood in the second product can be written as,

$$p(z_{j,k} | x_{\lambda_{j,k},k}) = \begin{cases} 1 & IF \quad \lambda_{j,k} = 0 \\ p(z_{j,k} | x_{\lambda_{j,k},k}) & otherwise. \end{cases} \tag{3.5}$$

Finally, we obtain,

$$p(\lambda_k|z_{1:k}) \propto p(\lambda_k)(V)^{-NC_k} \cdot \prod_{j=1}^{NT_k} p(z_{j,k}|x_{\lambda_{j,k},k}). \tag{3.6}$$

3.1.2 Particle Filter Based JPDA filter

Instead of maintaining the filtering distribution for the joint state $p(x_k|z_{1:k})$ the JPDA filter updates the marginal filtering distributions for each of the targets $p_m(x_{m,k}|z_{1:k})$, $m = 1 \cdots M$, through the Bayesian sequential estimation recursions [78]. According to the Bayesian inference theory,

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \tag{3.7}$$

which leads to (3.8).

$$p_m(x_{m,k}|z_{1:k}) \propto p_m(z_k|x_{m,k})p_m(x_{m,k}|z_{1:k-1}) \tag{3.8}$$

The likelihood for the mth target, $p_m(z_k|x_{m,k})$, can be derived as per [78],

$$p_m(z_k|x_{m,k}) \quad = \quad \beta_{0m} + \sum_{j=1}^{N_k} \beta_{jm} p(z_{j,k}|x_{m,k}), \tag{3.9}$$

where β_{jm} is the posterior probability that the jth measurement is associated with the mth target, with β_{0m} the posterior probability that the mth target is undetected.

The prediction probability $p_m(x_{m,k}|z_{1:k-1})$ can be expanded based on $x_{m,k-1}$,

$$\begin{aligned} p_m(x_{m,k}|z_{1:k-1}) \quad = \quad & \int [p_m(x_{m,k}|x_{m,k-1}) \times \\ & p_m(x_{m,k-1}|z_{1:k-1})] dx_{m,k-1}. \end{aligned} \tag{3.10}$$

Thus with the definitions for the one step ahead prediction distribution in (3.10) and the filtering distribution in (3.8), the JPDA filter fits within the Bayesian sequential estimation framework.

The original formulation of the JPDA filter in [53] and [54] assumes linear and Gaussian forms for the dynamic and likelihood models, and a Gaussian approximation for the filtering distribution. In order to make the general JPDA framework applicable to general nonlinear and non-Gaussian models, it can be implemented based on particle filter techniques.

The marginal filtering distributions for each of the targets are represented with particles, as is the case for the standard JPDA. More specifically, for the *mth* target, assume that a set of particles $\{w^i_{m,k-1}, x^i_{m,k-1}\}_{i=1}^{NP}$ is available, approximately distributed according to the marginal filtering distribution at the previous time step $p_m(x_{m,k-1}|z_{1:k-1})$. NP is the number of particles. At the current time step the predicted particles for the target state are generated from a suitably constructed proposal distribution, which may depend on the old state and the new measurements, i.e.,

$$\hat{x}^i_{m,k} \sim q_m(x_{m,k}|x^i_{m,k-1}, z_k), i = 1 \cdots NP. \tag{3.11}$$

Define $\tilde{x}_{m,k}$ as the pre-approximation of $x_{m,k}$, the state of the *mth* target at time step k. $\tilde{x}_{m,k}$ is estimated based on the predicted particles and their associated previous weights at time step $k-1$.

$$\tilde{x}_{m,k} = \sum_{i=1}^{NP} w^i_{m,k-1}\hat{x}^i_{m,k} \tag{3.12}$$

(3.12) can now be substituted into (3.6) to obtain $p(\lambda_k|z_{1:k})$, from which approximations for the marginal measurement to target association posterior probability, β_{jm}, can be computed according to (3.2). These approximations can, in turn, be used in (3.9) to calculate the target likelihood. Finally, setting the new importance weights to,

$$\begin{aligned} &w^i_{m,k} \propto w^i_{m,k-1}\frac{p_m(z_k|\hat{x}^i_{m,k})p_m(\hat{x}^i_{m,k}|x^i_{m,k-1})}{q_m(\hat{x}^i_{m,k}|x^i_{m,k-1}, z_k)}, \\ &\textstyle\sum_{i=1}^{NP} w^i_{m,k} = 1, \end{aligned} \tag{3.13}$$

leads to the sample set $\{w^i_{m,k}, \hat{x}^i_{m,k}\}_{i=1}^{NP}$ being approximately distributed according to the marginal filtering distribution at the current time step $p_m(x_{m,k}|z_{1:k})$.

A summary of the particle filter based JPDA filter algorithm is presented in what follows. Assuming that the sample sets $\{w^i_{m,k-1}, x^i_{m,k-1}\}_{i=1}^{NP}, m = 1 \cdots M$, are approximately distributed according to the corresponding marginal filtering distributions at the previous time step $p_m(x_{m,k-1}|z_{1:k-1}), m = 1 \cdots M$, the algorithm proceeds as follows at the current time step.

Algorithm 3.1: Particle Filter Based JPDA Filter

1. For $m = 1 \cdots M, i = 1 \cdots NP$, generate predicted particles for the target states $\hat{x}^i_{m,k} \sim q_m(x_{m,k}|x^i_{m,k-1}, z_k)$.

2. For $m = 1 \cdots M$, calculate $\tilde{x}_{m,k}$, the pre-approximation of $x_{m,k}$,
$$\tilde{x}_{m,k} = \sum_{i=1}^{NP} w^i_{m,k-1} \hat{x}^i_{m,k}. \tag{3.14}$$

3. Enumerate all the valid joint measurement to target association events to form the set Λ_k.

4. For each $\lambda_k \in \Lambda_k$, compute the posterior probability of the joint association event,
$$p(\lambda_k|z_{1:k}) \propto p(\lambda_k)(V)^{-NC_k} \cdot \prod_{j=1}^{NT_k} p(z_{j,k}|\tilde{x}_{\lambda_{j,k},k}). \tag{3.15}$$

5. For $m = 1 \cdots M, j = 1 \cdots N_k$, compute the marginal association posterior probability β_{jm},
$$\beta_{jm} = p(\lambda_{m,k} = j|z_{1:k}) = \sum_{\{\lambda_k \in \Lambda_k \; : \; \lambda_{m,k}=j\}} p(\lambda_k|z_{1:k}). \tag{3.16}$$

6. For $m = 1 \cdots M, i = 1 \cdots NP$, compute the target likelihood for each predicted particle,
$$p_m(z_k|\hat{x}^i_{m,k}) = \beta_{0m} + \sum_{j=1}^{N_k} \beta_{jm} p(z_{j,k}|\hat{x}^i_{m,k}). \tag{3.17}$$

7. For $m = 1 \cdots M, i = 1 \cdots NP$, compute and normalize the particle weights,
$$\begin{aligned} & w^i_{m,k} \propto w^i_{m,k-1} \frac{p_m(z_k|\hat{x}^i_{m,k}) p_m(\hat{x}^i_{m,k}|x^i_{m,k-1})}{q_m(\hat{x}^i_{m,k}|x^i_{m,k-1}, z_k)}, \\ & \textstyle\sum_{i=1}^{NP} w^i_{m,k} = 1. \end{aligned} \tag{3.18}$$

8. Resample the discrete distribution $\{w^i_{m,k} : i = 1, \cdots, NP\}$ NP times to generate particles $\{x^j_{m,k} : j = 1, \cdots, NP\}$, so that for any j, $Pr\{x^j_{m,k} = \hat{x}^i_{m,k}\} = w^i_{m,k}$. Set the weights $w^i_{m,k}$ to $\frac{1}{NP}$, $i = 1, ..., NP$, and move to Stage 1.

The resulting sample sets $\{w^i_{m,k}, x^i_{m,k}\}^{NP}_{i=1}, m = 1 \cdots M$, are then approximately distributed according to the corresponding marginal filtering distributions at the current time step $p_m(x_{m,k}|z_{1:k}), m = 1 \cdots M$.

3.1.3 Particle Filter Based Multi-scan JPDA Algorithm

The standard single scan JPDA algorithm updates a track with a weighted sum of the measurements which could have reasonably originated from the target in track. The only information the standard JPDA algorithm uses is the measurements on the present scan and the state vectors.

If more scans of measurements are used, additional information is available resulting in better computed probabilities. The best possible filter for a single target in clutter (in the Bayesian point of view) is a weighted average of all combinations of measurements form the initial to the present time [53]. The same idea holds true for the multiple target case. If a tracking system could use all combinations of measurements in a weighted average from the initiation of each track to the present scan, all available information would be used to compute those weights and the best Bayesian track could be produced. Use of all available information from the past to the present is what the multiple hypothesis tracking (MHT) method tries to exploit. Most, if not all, tracking system is unable to store all of the measurements from all the scans. Therefore, a Bayesian tracking system can at best rely on a sliding window of scans. This section extends single scan JPDA to multiple scan JPDA filter for a sliding window of scans, both based on particle filter algorithm.

The multiple scan JPDA calculation examines multiple scan joint association events. The measurement to target association event of multiple scan is defined as $\lambda_{k-L+1:k}$, where L denotes the length of the multiple scan sliding window. $\lambda_{k-L+1:k}$

is composed by the association vectors at each scan in the sliding window, $\lambda_{k-L+1:k} = (\lambda_{k-L+1}, \lambda_{k-L+2}, \cdots, \lambda_k)$. The elements of the association vector at time step k, $\lambda_k = (\lambda_{1,k}, \cdots, \lambda_{j,k}, \cdots, \lambda_{N_k,k})$ are given by,

$$\lambda_{j,k} = \begin{cases} 0 & IF\ measurement\ j \\ & is\ due\ to\ clutter \\ m \in \{1 \cdots M\} & IF\ measurement\ j \\ & stems\ from\ target\ m. \end{cases} \tag{3.19}$$

The heart of the new algorithm is to find the posterior probability for the joint association event of multiple scans. That is to calculate $p(\lambda_{k-L+1:k}|z_{1:k})$ and it can be written as,

$$\begin{aligned} & p(\lambda_{k-L+1:k}|z_{1:k}) \\ & \propto p(z_k \cdots z_{k-L+1}|\lambda_{k-L+1:k}, z_{1:k-L})p(\lambda_{k-L+1:k}|z_{1:k-L}) \\ & \propto p(z_k \cdots z_{k-L+1}|\lambda_{k-L+1:k}, z_{1:k-L})p(\lambda_{k-L+1:k}), \end{aligned} \tag{3.20}$$

where the conditioning of $\lambda_{k-L+1:k}$ on the history of measurements before the sliding window has been eliminated. The distribution of the measurements in the sliding window based on a specific association event is given by,

$$\begin{aligned} & p(z_k \cdots z_{k-L+1}|\lambda_{k-L+1:k}, z_{1:k-L}) \\ & = \prod_{s=1}^{L}[\prod_{j=1}^{N_{k-L+s}} p(z_{j,k-L+s}|\lambda_{k-L+1:k}, z_{1:k-L})]. \end{aligned} \tag{3.21}$$

To reduce the notation, the index of the scan s in the sliding window is denoted by $k_s = k - L + s$. Then we obtain,

$$\begin{aligned} & p(z_k \cdots z_{k-L+1}|\lambda_{k-L+1:k}, z_{1:k-L}) \\ & = \prod_{s=1}^{L}[\prod_{j=1}^{N_{k_s}} p(z_{j,k_s}|\lambda_{k-L+1:k}, z_{1:k-L})] \\ & = \prod_{s=1}^{L}[\prod_{j \in I_{0,k_s}} p_C(z_{j,k_s}) \cdot \prod_{j=1}^{NT_{k_s}} p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s})] \\ & = \prod_{s=1}^{L}[(V)^{-NC_{k_s}} \cdot \prod_{j=1}^{NT_{k_s}} p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s})], \end{aligned} \tag{3.22}$$

where,

$$p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s}) = \begin{cases} 1 & IF \quad \lambda_{j,k_s} = 0 \\ p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s}) & otherwise. \end{cases} \tag{3.23}$$

The joint hypothesis association of multiple scan is obtained as,

$$\begin{aligned} p(\lambda_{k-L+1:k}|z_{1:k}) \quad &\propto \quad p(\lambda_{k-L+1:k}) \textstyle\prod_{s=1}^{L}[(V)^{-NC_{k_s}} \\ &\quad \cdot \textstyle\prod_{j=1}^{NT_{k_s}} p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s})]. \end{aligned} \tag{3.24}$$

Then the marginal measurement to target association posterior probabilities considering multiple scans can be computed according to (3.25).

$$\begin{aligned} \beta_{jm} \quad &= \quad p(\lambda_{j,k} = m|z_{1:k}) \\ &= \quad \textstyle\sum_{\{\lambda_{k-L+1:k} \in \Lambda_{k-L+1:k} \ : \ \lambda_{j,k}=m\}} p(\lambda_{k-L+1:k}|z_{1:k}) \end{aligned} \tag{3.25}$$

These approximations can, in turn, be used in (3.9) to approximate the target likelihood. Finally, setting the new importance weights via (3.13), which leads to the sample set $\{w^i_{m,k}, \hat{x}^i_{m,k}\}_{i=1}^{NP}$ being approximately distributed according to the marginal filtering distribution at the current time step $p_m(x_{m,k}|z_{1:k})$.

A summary of the particle filter based multiple scan JPDA filter algorithm is presented in what follows. Assuming that the sample sets $\{w^i_{m,k-1}, x^i_{m,k-1}\}_{i=1}^{NP}, m = 1 \cdots M$, are approximately distributed according to the corresponding marginal filtering distributions at the previous time step $p_m(x_{m,k-1}|z_{1:k-1}), m = 1 \cdots M$, the algorithm proceeds as follows at the current time step.

Algorithm 3.2: Particle Filter Based Multi-scan JPDA Filter

1. For $m = 1 \cdots M, i = 1 \cdots NP$, generate predicted particles for the target states $\hat{x}^i_{m,k} \sim q_m(x_{m,k}|x_{m,k-1}, z_k)$.

2. For $m = 1 \cdots M$, calculate $\tilde{x}_{m,k}$, the pre-approximation of $x_{m,k}$,

$$\tilde{x}_{m,k} = \sum_{i=1}^{NP} w^i_{m,k-1} \hat{x}^i_{m,k}. \tag{3.26}$$

3. Enumerate all the valid joint measurement to target association events in the sliding window $k - L + 1 : k$ to form the set $\Lambda_{k-L+1:k}$.

4. For each $\lambda_{k-L+1:k} \in \Lambda_{k-L+1:k}$, compute the posterior probability of the joint association event in multiple scan,

$$\begin{aligned} p(\lambda_{k-L+1:k}|z_{1:k}) \quad \propto \quad & p(\lambda_{k-L+1:k}) \textstyle\prod_{s=1}^{L}[(V)^{-NC_{k_s}} \\ & \cdot \textstyle\prod_{j=1}^{NT_{k_s}} p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s})]. \end{aligned} \tag{3.27}$$

5. For $m = 1 \cdots M, j = 1 \cdots N_k$, compute the marginal association posterior probability,

$$\begin{aligned} & \beta_{jm} \\ & = p(\lambda_{j,k} = m|z_{1:k}) \\ & = \textstyle\sum_{\{\lambda_{k-L+1:k} \in \Lambda_{k-L+1:k} \;:\; \lambda_{j,k}=m\}} p(\lambda_{k-L+1:k}|z_{1:k}). \end{aligned} \tag{3.28}$$

6. For $m = 1 \cdots M, i = 1 \cdots NP$, compute the target likelihood,

$$p_m(z_k|\hat{x}^i_{m,k}) = \beta_{0m} + \sum_{j=1}^{N_k} \beta_{jm} p(z_{j,k}|\hat{x}^i_{m,k}). \tag{3.29}$$

7. For $m = 1 \cdots M, i = 1 \cdots NP$, compute and normalize the particle weights,

$$\begin{aligned} & w^i_{m,k} \propto w^i_{m,k-1} \tfrac{p_m(z_k|\hat{x}^i_{m,k}) p_m(\hat{x}^i_{m,k}|x^i_{m,k-1})}{q_m(\hat{x}^i_{m,k}|x^i_{m,k-1}, z_k)}, \\ & \textstyle\sum_{i=1}^{NP} w^i_{m,k} = 1. \end{aligned} \tag{3.30}$$

8. Resample the discrete distribution $\{w^i_{m,k} : i = 1, \cdots, NP\}$ NP times to generate particles $\{x^j_{m,k} : j = 1, \cdots, NP\}$, so that for any j, $Pr\{x^j_{m,k} = \hat{x}^i_{m,k}\} = w^i_{m,k}$. Set the weights $w^i_{m,k}$ to $\frac{1}{NP}$, $i = 1, ..., NP$, and move to Stage 1.

3.1.4 Simulation Results and Analysis

The simulation is carried out for tracking two slow-maneuvering targets in clutter. The Wiener process acceleration model (2.79) is chosen as the motion model for the two targets. The process noise $v_{k-1} = [v_{px}, v_{vx}, v_{ax}, v_{py}, v_{vy}, v_{ay}]^T_{k-1}$, is a zero mean Guassian white noise process with standard deviations of 1 m ($\sigma_{v_{px}}$), 1 m/s ($\sigma_{v_{vx}}$), 20 m/s^2 ($\sigma_{v_{ax}}$), 1 m ($\sigma_{v_{py}}$), 1 m/s ($\sigma_{v_{vy}}$) and 20 m/s^2 ($\sigma_{v_{ay}}$).

Target one starts at location $[-310\ 310]$ in $x-y$ Cartesian coordinates in meters with the initial velocity (in m/s) $[10\ \ -400]$. Target two starts at location (in m) $[-310\ \ -20310]$ with the initial velocity (in m/s) $[10\ 400]$. A track-while-scan (TWS) radar is positioned at the origin of the plane, whose details are provided in Section 2.2.5.

The sampling interval is $\Delta T = 1s$ and it is assumed that the probability of detection $P_D = 0.9$ for the radar. For generating measurements in simulations, the clutter is assumed uniformly distributed with density $1 \times 10^{-6}/m^2$.

In the particle filter based multi-scan methods, each target model is assigned with 500 particles. The length of the multiple scan sliding window (L) is chosen as 3. The algorithm is initialized with Gaussians around the initial states of the true targets.

The proposed method is compared with the standard JPDA filter and the particle filter based single scan JPDA method on tracking multiple slow-maneuvering targets. The simulation results are obtained from 100 Monte Carlo runs. Fig. 3.1 $\sim$ Fig. 3.3 show one realization respectively performed by the three methods. The RMSE in position at each time step for the three methods are respectively shown in Fig. 3.4 $\sim$ Fig. 3.6. The performance of the three methods are also compared via global RMSE (in position), executing time, tracking loss rate and swap rate, which are listed in Table. 3.1.

Compared with the standard JPDA method (based on extended Kalman filter), the particle filter based JPDA methods (single scan and multiple scan) are much more accurate and robust, at the cost of longer computing time. This verifies that when dealing with nonlinear problem (nonlinear observation equation) and large random acceleration (large process noise), the performance of particle filter is better than extended Kalman filter using local linearization. Compared to the particle filter based single scan JPDA method, the particle filter based multi-scan JPDA method provides better performance since the additional information of more scans improve the association probabilities resulting in lower estimation errors (RMSE) and larger robustness (tracking loss rate).

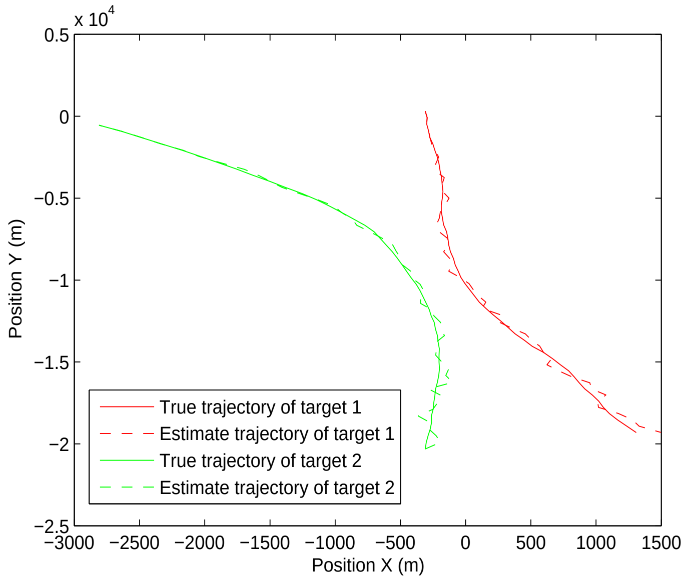

Figure 3.1: True and estimate trajectories of two targets using JPDA method

3.2 Multi-scan Mixture Particle Filter

As discussed in Section 1.4, most of the particle filter based multiple target tracking methods (including the first four categories) use particles whose dimension is the sum of those of the individual state space corresponding to each target. They suffer from the curse of dimensionality problem since as the number of targets increases, the size of the joint state-space increases exponentially. If care is not taken in the design of proposal distributions an exponentially increasing number of particles may be required to cover the support of the multi-target distribution and maintain a given level of accuracy.

However, the fifth category based on mixture particle filter avoids the dimension problem by exploring the particle filter's ability to track multiple targets in a single-target state space. The posterior distribution of the target state is a multi-mode

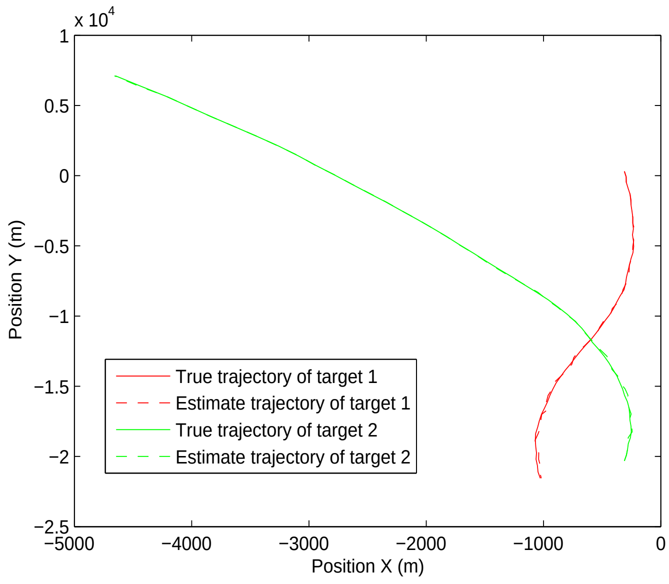

Figure 3.2: True and estimate trajectories of two targets using particle filter based single scan JPDA method

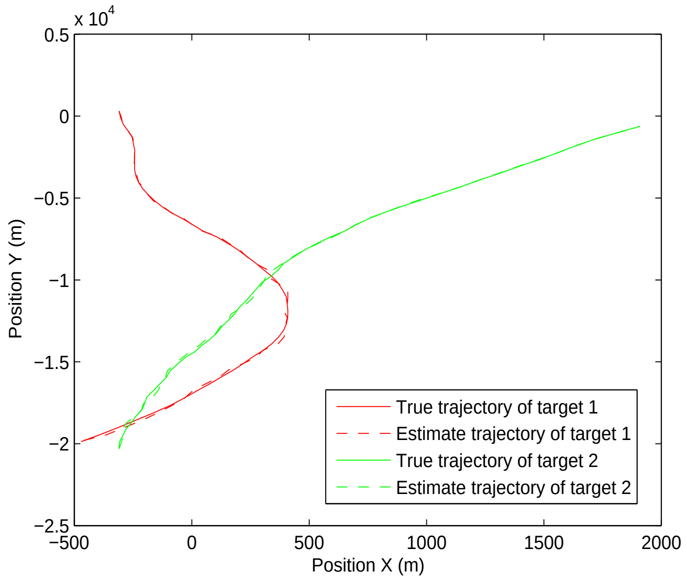

Figure 3.3: True and estimate trajectories of two targets using particle filter based multi-scan JPDA method

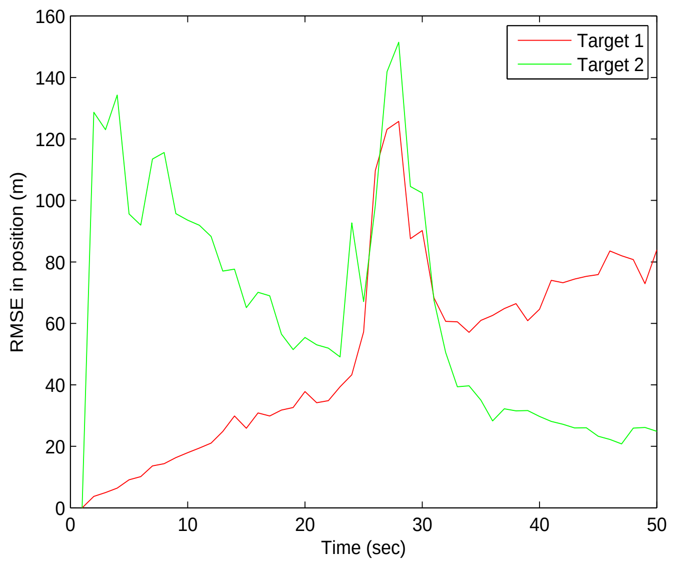

Figure 3.4: RMSE in position using JPDA method

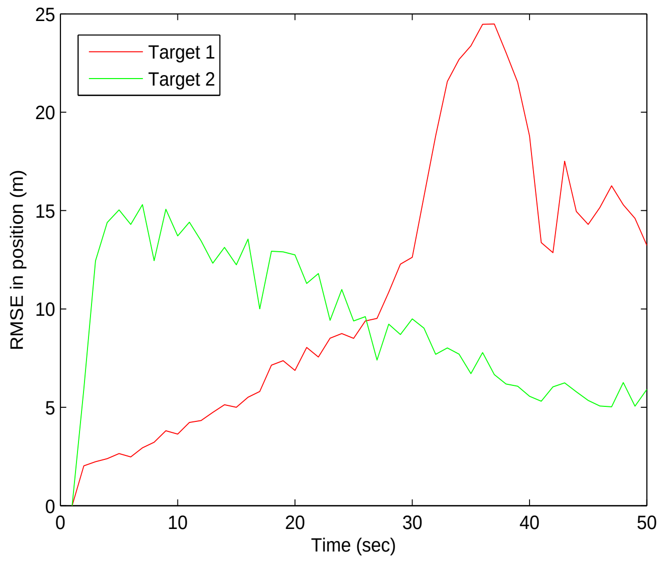

Figure 3.5: RMSE in position using particle filter based single scan JPDA method

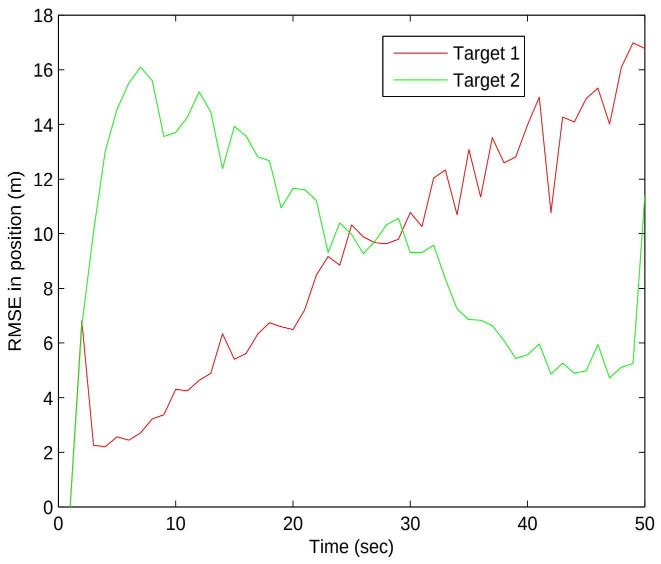

Figure 3.6: RMSE in position using particle filter based multi-scan JPDA method

Table 3.1: Performance Comparison

	RMSE (m)	Executing Time (s)	Tracking Loss Rate	Swap Rate
JPDA Method	Target1: 59.5403, Target2: 74.8563	0.0758	24%	0
Particle Filter Based Single Scan JPDA	Target1: 20.8201, Target2: 17.0738	1.2893	6%	0
Particle Filter Based Multiple Scan JPDA	Target1: 10.0852, Target2: 10.3445	1.831	0	0

distribution and each mode corresponds to either the target or the clutter. As pointed out in [77], particle filters may perform poorly when the posterior distribution of the target state is multi-mode in nature as a result of ambiguities and multiple targets in single-target state space. To circumvent this problem, a mixture particle filter method is introduced in [77], where each mode is modeled with an individual particle filter that forms part of the mixture. The filters in the mixture interact only through the computation of the importance weights. By distributing the resampling step to individual filters, the well known problem of sample impoverishment is avoided, which is largely responsible for loss of track.

However, for the algorithm reported in [77], it is difficult to handle the new target appearance problem. In the initialization process of the algorithm, each mode is assigned with a particle filter and no new particle filters are incorporated to represent the new modes occurred due to the appearance of new targets during the subsequent tracking process. Moreover, the algorithm is utilized in a clutter-free environment. Tracking in cluttered environment is tackled in [105], where a data association method based on the assumption of the motion continuity is used to find the mode corresponding to the true target. However, it is assumed that the number of targets is fixed and known in [105], which cannot work when the number of targets varies.

In this thesis, a new algorithm named multi-scan mixture particle filter is proposed to track varying number of targets in cluttered environment. In order to distinguish the targets from the clutters, multiple scan information is incorporated. Moreover, to track the newly appeared targets, a set of new particles are sampled from the likelihood model (according to the most recent measurements) detecting the new modes at each time step. When targets disappear, the modes corresponding to the disappeared targets are assigned with small existence probabilities and are eliminated via the decision process.

The rest of this section is organized as follows. The mixture particle filter algorithm is introduced in Section 3.2.1. The proposed multi-scan mixture particle filter algorithm is presented in Section 3.2.2. Simulation results and analysis are provided in Section 3.2.3.

3.2.1 Mixture Particle Filter

The mixture particle filter is utilized to deal with multiple target tracking problem in [77]. The posterior distribution, $p(x_k|z_{1:k})$, is modeled as an M-component non-parametric mixture model:

$$p(x_k|z_{1:k}) = \sum_{m=1}^{M} \pi_{m,k} p_m(x_k|z_{1:k}), \tag{3.31}$$

where $p_m(x_k|z_{1:k})$ is the filtering distribution for the mth component. The component weight $\pi_{m,k}$ is computed from its previous weight and, the sum of the weights of the particles belonging to the mth component,

$$\pi_{m,k} = \frac{\pi_{m,k-1}\tilde{W}_{m,k}}{\sum_{n=1}^{M} \pi_{n,k-1}\tilde{W}_{n,k}}, \tag{3.32}$$

where,

$$\tilde{W}_{m,k} = \sum_{i \in I_m} \tilde{w}_m^i, \tag{3.33}$$

and, I_m is the set of indices of the particles belonging to the mth component. $\tilde{w}_m^i$ is the un-normalized importance weight of the ith particle in the mth component.

This means that the filtering recursion can be performed for each component individually. Hence in mixture particle filter, each component is modeled with an individual particle filter. The filters in the mixture interact only through the computation of the component weights. By distributing the resampling step to individual filters, the multi-modal distribution is maintained during the propagation in time.

In [77], the mixture particle filter method is applied in a clutter-free environment. At the end of the tracking process, all the final modes correspond to the targets. However, it is difficult for the mixture particle filter to detect new targets that appear during the tracking process.

3.2.2 Multi-scan Mixture Particle Filter

3.2.2.1 Overview of the Proposed Algorithm

A flow diagram of the proposed tracking algorithm is shown in Fig 3.7. The tracking process can be mainly divided into three parts: the initialization process, the normal tracking process and the new mode detection process (Fig 3.7).

The initialization process is composed of N_{ini} scans. Initially, NP particles are sampled uniformly from the surveillance area. The importance weights of the particles are calculated at the end of the first scan and the particles are then resampled according to their corresponding weights. The resampled particles are clustered to different modes. Each mode is assigned with an individual particle filter. The existence probability of each mode, θ_k^m $(m = 1, \cdots, M)$, is calculated. In the subsequent scans in the initialization process, each mode is propagated based on the dynamic model and θ_k^m $(m = 1, \cdots, M)$ is calculated. At the end of the initialization process, a decision is made based on θ_k^m $(m = 1, \cdots, M,\ k = N_{ini})$: the mode with θ_k^m larger than or equal to T_{target} is treated as target mode; the mode with θ_k^m less than or equal to $T_{clutter}$ is treated as clutter mode; while the mode with θ_k^m in the range $(T_{clutter}, T_{target})$ is treated as tentative target mode. T_{target} and $T_{clutter}$ are the thresholds for the target and clutter detection respectively. The modes corresponding to the clutters are eliminated, while the modes corresponding to the targets and tentative targets are retained and propagated to the following scan.

In the subsequent normal tracking process, when the new measurements arrive at time step $k + 1$, the modes from the previous scan (including target modes and tentative target modes) are propagated according to the dynamic model to obtain the predicted modes, and the existence probability of each predicted mode is calculated. The predicted modes and their associated existence probabilities are input to a decision module (Fig. 3.8). After the decision process, the predicted modes are classified into three kinds of modes: the target mode, the tentative target mode and clutter mode based on their associated existence probabilities.

In the new mode detection process, NP new particles are sampled from the likelihood function including the current measurements. The new particles are resampled according to their associated weights and the resampled particles are clustered to obtain new modes, which are treated as tentative target modes and are returned for the following time step iteration.

3.2.2.2 Calculation of the Existence Probability

The calculation of the existence probability for each mode is similar to the calculation of the component weight $\pi_{m,k}$ [77]. For each mode, its existence probability is computed based on multi-scan information.

$$\theta_k^m = \frac{\theta_{k-1}^m \tilde{W}_{m,k}}{\sum_{n=1}^{M} \theta_{k-1}^n \tilde{W}_{n,k}} \tag{3.34}$$

$$\tilde{W}_{m,k} = \sum_{i \in I_m} \tilde{w}_m^i \tag{3.35}$$

where I_m is the set of indices of the particles belonging to the mth mode and $\tilde{w}_m^i$ is the un-normalized importance weight of the ith particle in the mth mode.

3.2.2.3 Sampling from the Likelihood Function

The posterior distribution of the target state varies when tracking varying number of targets. The number of modes of the posterior distribution may increase or decrease due to the appearance or disappearance of targets. New features of the posterior distribution are found during the tracking process. The standard particle filter based on the traditional prior model sampling method can not cope with the new features, since it provides no opportunity to generate new values for unknown quantities after their initial generation. An additional procedure is needed to sample new particles to adapt to the new features of the posterior distribution.

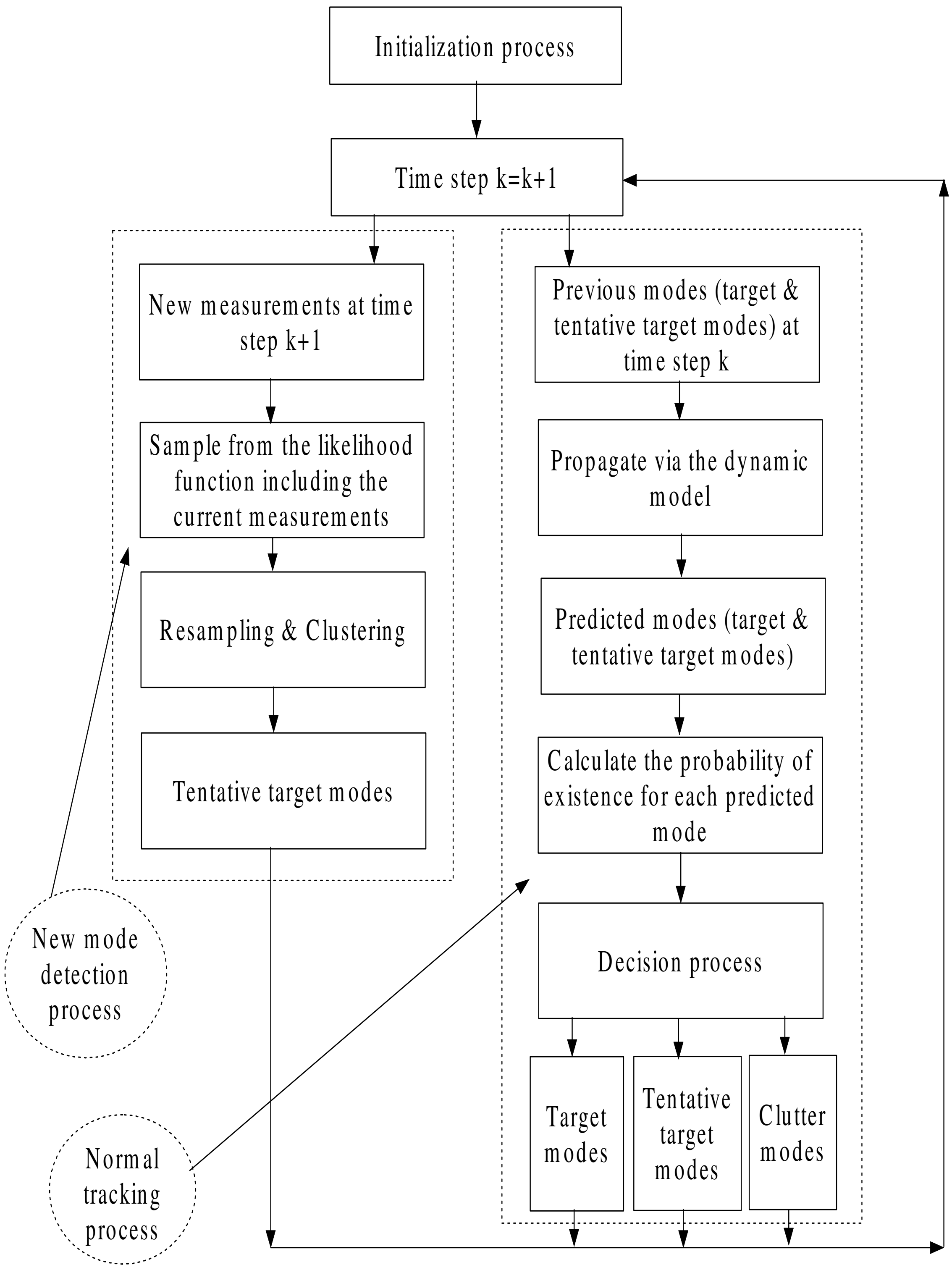
Initialization process
Time step k=k+1
New measurements at time step k+1
Sample from the likelihood function including the current measurements
Resampling & Clustering
Tentative target modes
Previous modes (target & tentative target modes) at time step k
Propagate via the dynamic model
Predicted modes (target & tentative target modes)
Calculate the probability of existence for each predicted mode
Decision process
Target modes
Tentative target modes
Clutter modes
New mode detection process
Normal tracking process

Figure 3.7: Flow diagram of the proposed tracking algorithm

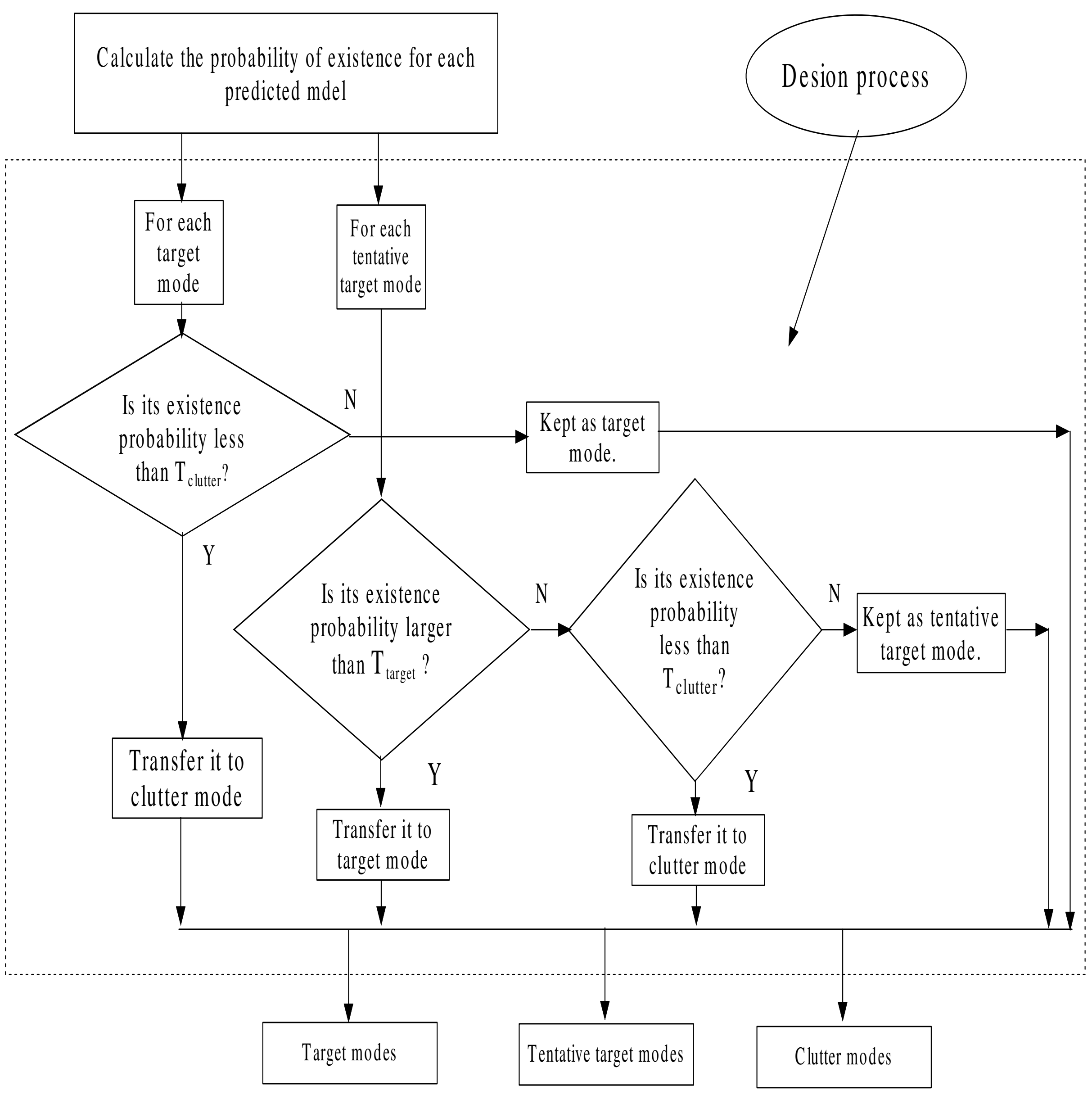
Calculate the probability of existence for each predicted mdel
Desion process
For each target mode
For each tentative target mode
Is its existence probability less than $T_{clutter}$?
N
Kept as target mode.
Y
Is its existence probability larger than T_{target} ?
N
Is its existence probability less than $T_{clutter}$?
N
Kept as tentative target mode.
Transfer it to clutter mode
Y
Transfer it to target mode
Y
Transfer it to clutter mode
Target modes
Tentative target modes
Clutter modes

Figure 3.8: Flow diagram of the decision module

To adapt to the new features, samples are drawn according to the most recent measurements. The key idea is to sample x_k^i directly from the likelihood model,

$$\bar{q} = p(z_k|x_k). \tag{3.36}$$

The method, which samples from the likelihood model, can be viewed as the logical "inverse" of the prior model sampling method. Rather than forward-guessing and then using the importance factors to adjust the likelihoods of hypothesis based on measurements, the likelihood sampling method guesses "backwards" from the measurements and adjusts the importance factors based on the belief $p(x_{k-1}|z_{1:k-1})$.

The importance weight of the sample x_k^i that is sampled from $\bar{q}$ can be written as:

$$\begin{aligned} w_k^i &\propto [p(z_k|x_k)]^{-1} p(z_k|x_k^i) p(x_k^i|x_{k-1}^i) p(x_{k-1}^i|z_{1:k-1}) \\ &= p(x_k^i|x_{k-1}^i) p(x_{k-1}^i|z_{1:k-1}). \end{aligned} \tag{3.37}$$

Computing these importance weights is not trivial, since $p(x_{k-1}|z_{1:k-1})$ is represented by a set of samples. The strategy here is to employ a two-staged approach that first approximates $p(x_k^i|x_{k-1}^i)p(x_{k-1}^i|z_{1:k-1})$ and then use this approximate density to calculate the desired importance weights. The following procedure implements this importance sampler as shown in [5]:

(i) Generate a set of samples x_k^i, first by sampling from $p(x_{k-1}^i|z_{1:k-1})$ and then sampling from $p(x_k^i|x_{k-1}^i)$. Obviously, these samples approximate $p(x_k^i|x_{k-1}^i)p(x_{k-1}^i|z_{1:k-1})$.

(ii) Transform the resulting samples set into a kd-tree [106, 107]. The tree generalizes samples to arbitrary states, x_k^i, in state space, which is necessary to calculate the desired importance weights.

(iii) Lastly, sample x_k^i from the proposal distribution $p(z_k|x_k^i)$. Assign each sample with an importance weight that is proportional to its probability under the previously generated kd-tree.

Consequently, the likelihood sampling method possesses complimentary strengths and weaknesses: while it is ideal for detecting the new features of the posterior distribution, its performance is affected by high measurement noise.

3.2.3 Simulation Results and Analysis

The proposed multi-scan mixture particle filter algorithm is simulated to initiate tracks, and track variable number of targets, handling the target appearance and disappearance problems. In the process of initiating tracks, nothing is known about the number of targets and their initial positions.

The target motion is modeled in Cartesian coordinates as,

$$X_k = \Phi X_{k-1} + \Gamma v_{k-1}, \tag{3.38}$$

$$\Phi = \begin{bmatrix} 1 & \Delta T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.39}$$

$$\Gamma = I_{4\times4}, \tag{3.40}$$

where Φ is the transition matrix and ΔT is the sample interval. $X_k = [px, vx, py, vy]_k^T$ is the state vector; px and vx denote respectively the position and velocity of the moving object along the x axis of Cartesian frame; and, py and vy along the y axis. $v_k = [v_{px}, v_{vx}, v_{py}, v_{vy}]_k^T$ is the zero mean Guassian white noise process with covariance $Q : E[v_k\ v_j^T] = Q\delta_{jk}$, where,

$$Q = \begin{bmatrix} \sigma_{px}^2 & 0 & 0 & 0 \\ 0 & \sigma_{vx}^2 & 0 & 0 \\ 0 & 0 & \sigma_{py}^2 & 0 \\ 0 & 0 & 0 & \sigma_{vy}^2 \end{bmatrix}. \tag{3.41}$$

A linear sensor is assumed with measurement equation,

$$Z_k = HX_k + n_k, \tag{3.42}$$

where,

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{3.43}$$

The measurement noise $n_k = [n_{z_1}, v_{z_2}]_k^T$ is a zero mean Guassian white noise process with variance $R : E[n_k \; n_j^T] = R\delta_{kj}$, where,

$$R = \begin{bmatrix} \sigma_{z_1}^2 & 0 \\ 0 & \sigma_{z_2}^2 \end{bmatrix}. \tag{3.44}$$

A two-dimensional surveillance situation is considered. The area under surveillance is 40 m long and 40 m wide. The false measurements satisfied a Poisson distribution with density 3 $/m^2$. At the first time step of the simulation, one target appeared at position $(10\ m, 5\ m)$ with velocity $(-2\ m/s, -1\ m/s)$, and another target appeared at position $(-10\ m, 5\ m)$ with velocity $(-2\ m/s, -1\ m/s)$. Both targets maintained their velocities thereafter. At time step 25, the third target appeared at position $(-10\ m, -5\ m)$ with a constant velocity $(2\ m/s, 1\ m/s)$. The third target disappeared at time step 30. Each simulation consisted of 50 time steps.

3.2.3.1 Initiating Tracks

Initially, NP particles are sampled uniformly from the surveillance area (Fig. 3.9). At the second scan, the particles are resampled according to their associated weights and are clustered to different modes (Fig. 3.10). During the subsequent initialization process, in each scan every mode is propagated based on the dynamic model and the existence probability of every mode, $\theta_k^m, m = 1, \cdots, M$, is calculated. At the end of the initialization process, the decision is made based on the existence probabilities of the modes: the mode with θ_k^m larger than or equal to T_{target} is treated as target mode; the mode with θ_k^m less than or equal to $T_{clutter}$ is treated as clutter mode; while the mode with θ_k^m in the range $(T_{clutter}, T_{target})$ is treated as tentative target mode. The modes corresponding to the clutters are eliminated, while the modes corresponding to the targets and tentative targets are retained and propagated to the subsequent scan (Fig. 3.11).

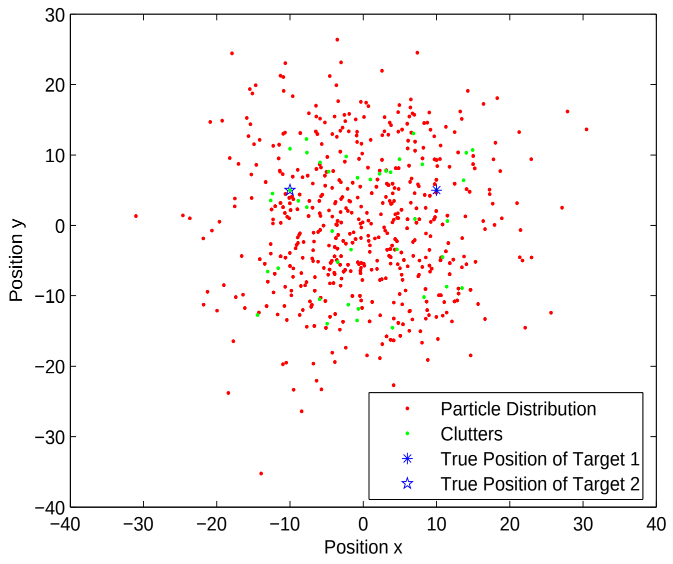

Figure 3.9: Initiating tracks (frame 1)

3.2.3.2 Detecting the Target Appearance

At time step 25, the third target appeared at position $(-10\ m, -5\ m)$ with a constant velocity $(2\ m/s, 1\ m/s)$. NP new particles are sampled from the likelihood function including the current measurements. The new particles are resampled according to their associated weights and the resampled particles are clustered to obtain new modes, which are treated as tentative target modes (Fig. 3.12). In the subsequent scans, the existence probability of each mode is calculated based on the information from the previous scans. At time step 28, the tentative target mode corresponding to the third target is transferred to the true target, and the tentative target modes corresponding to clutters are eliminated (Fig. 3.13).

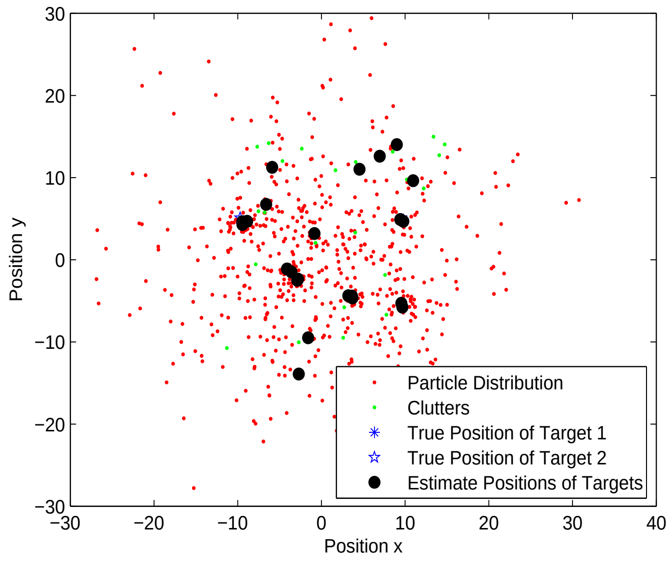

Figure 3.10: Initiating tracks (frame 2)

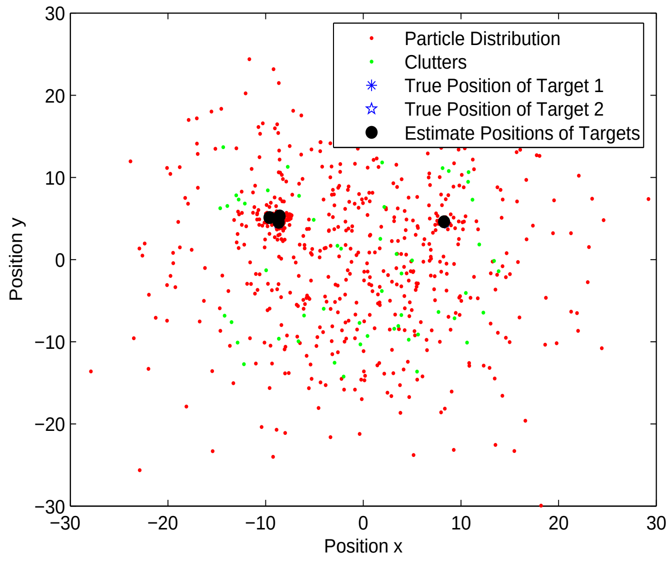

Figure 3.11: Initiating tracks (frame 10)

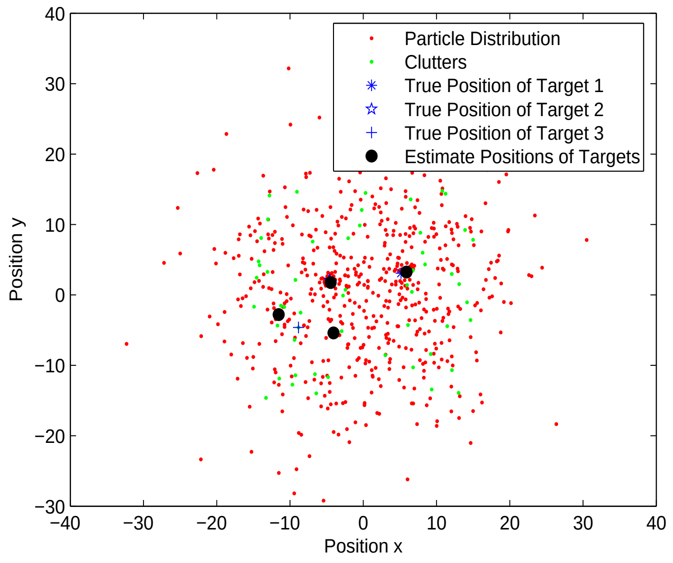

Figure 3.12: Detecting the target appearance (frame 25)

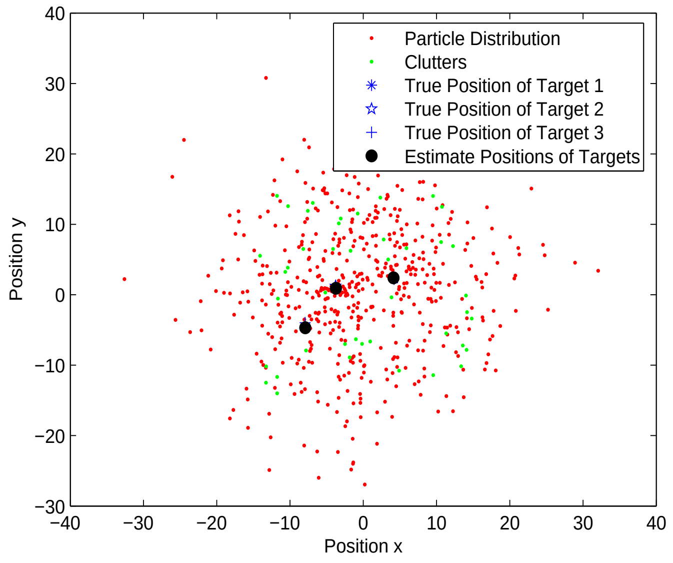

Figure 3.13: Detecting the target appearance (frame 28)

3.2.3.3 Detecting the Target Disappearance

The third target disappeared at the end of time step 44. In the subsequent scans, the existence probability of the mode corresponding to the disappeared target is calculated based on the information from the previous scans. At time step 48, the mode corresponding to the disappeared target is eliminated once its existence probability falls below the threshold $T_{clutter}$ (Fig. 3.14).

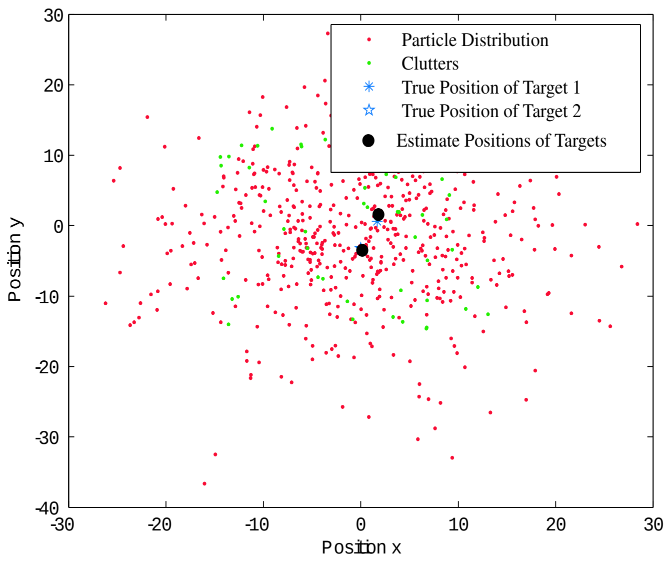

Figure 3.14: Detecting the target disappearance (frame 48)

3.3 Conclusions

In this chapter, two different algorithms based on particle filter are presented for multiple target tracking: the particle filter based multi-scan joint probabilistic data association (MS-JPDA) filter and, the multi-scan mixture particle filter. In the MS-JPDA filter, each of the tracking targets is assigned with a corresponding particle filter. The distribution of interest is the marginal filtering distribution for each of the targets, which is approximated with particles. The MS-JPDA filter examines the joint association hypothesis in a multi-scan sliding window and calculates the posterior marginal probability based on the multi-scan joint association hypothesis. Compared with the single scan JPDA methods, the multi-scan JPDA method uses richer information, which results in better estimated probabilities.

The multi-scan mixture particle filter applies the particle filter method directly in the multiple target tracking process and avoids the data association problem. The posterior distribution of the target state is a multi-mode distribution and each mode corresponds to either the target or the clutter. In order to distinguish the targets from the clutters, multiple scan information is incorporated. Moreover, to deal with the new target appearance problem, new particles are sampled from the likelihood model (according to the most recent measurements) to detect the new modes appearing at each time step.

Chapter 4

Multiple Maneuvering Target Tracking By Improved Particle Filter Based on Multi-scan JPDA

4.1 Introduction

Target tracking is utilized widely in the area of weapon delivery systems, air defence, ocean/battle field surveillance, air traffic control, and robotics. By far, the most complicated case in target tracking is to track multiple maneuvering targets in heavy clutter. This class of problem has received considerable attention in recent years.

Sequential Bayesian framework is the most commonly used framework for multiple maneuvering target tracking applications in which the posterior distribution of the target state is recursively estimated upon receipt of new observations in time. The implementation of the sequential Bayesian framework to real world tracking applications faces a few challenges. The state-space models are often nonlinear and non-Gaussian so that no close-form analytic solutions can be obtained [108], [109]. Methods relying on linear and Gaussian assumptions are susceptible to failure and collapse. Furthermore, a real world target may occasionally perform manoeuvring movements during its motion. A single fixed dynamical model for modeling the target motions is not enough to represent this behavior. To address this issue multiple model approaches are proposed to model highly maneuvering targets and the interacting multiple model (IMM) algorithm [40], [41], [42] is the most popular one among them. Moreover, in presence of clutter, some of the sensor measurements may not originate from the target-of-interest. In this case one has to solve the problem of data association. An effective approach in a Bayesian framework is that of probabilistic data association (PDA) [53], [110] for a single target in clutter and that of joint probabilistic data association (JPDA) [53], [111], [54] for multiple targets in clutter.

Recently, the IMM method, which is used to track highly maneuvering targets and PDA/JPDA methods, which are used to solve data association problems, are combined to tackle multiple maneuvering target tracking problems. In [112] the IMM algorithm is combined with the PDA filter in a multiple sensor scenario proposing a combined IMMPDA algorithm. In [113] multiple targets in clutter are considered using JPDA filter which, unlike the PDA filter, accounts for the interference from other targets. Various versions of IMMJPDA filters for multiple target tracking can be found in [114], [104], [115], [116]. In [114], an IMMJPDA-Coupled filter is developed for situations where the measurements of two targets are unresolved during periods of close encounter. In [104], an IMMJPDA uncoupled fixed-lag smoothing algorithm is developed with IMMJPDA uncoupled tracking as a special case. The fixed-lag smoothing algorithm is developed by applying IMM approach and JPDA technique to a state-augmented system. In [115], multi-scan information is incorporated in JPDA to solve the data association problem during the multiple maneuvering target tracking process. In [116], the JPDA and IMM algorithms are combined to tackle the track coalescence problem during multiple target tracking process.

Besides the IMMPDA/IMMJPDA algorithms, several different methods are proposed to solve the multiple maneuvering target tracking problem. In [117], the probability hypothesis density (PHD) is applied to jointly detect and track multiple maneuvering targets. In [118], genetic algorithm is utilized to solve the multiple target data association problem. In [119], a new MHT filter based on target existence probability is introduced to track multiple maneuvering targets. The probability that a target exists is estimated, and the estimate of the target kinematic state is subsequently conditioned on target existence.

In the above methods, the data models are assumed Gaussian and weakly nonlinear, and the Kalman filter/extended Kalman filter (KF/EKF) is used to perform target state estimation. However, the linear/linearized filter may not always be good especially in conditions where the state or measurement equation is nonlinear and the noises are non-Gaussian, for example, when the filter update is slow or the target maneuver is large.

More recently, nonlinear filtering techniques have been gaining more attention. The most common one among them is particle filter, a sampling based algorithm. Particle filter, which uses sequential Monte Carlo methods for on-line learning within a Bayesian framework, can be applied to any state-space models. Particle filter is more suitable than Kalman filter and EKF when dealing with non-linear and non-Gaussian estimation problems.

Moreover, in the above approaches multiple model methods are adopted to track the highly maneuvering targets. The possible multiple motion models and transition probability matrices are assumed as known. In practice, the dynamics is hard to break up into several different motion models and the model transition probabilities are difficult to obtain. A general model is needed to cope with the wide variety of motions exhibited by a maneuvering target.

In this work the equivalent-noise approach [95], [96], [97] is adopted, which uses one general model in maneuvering target tracking based on the assumption that the maneuver effect can be modeled by (part of) a white or colored noise process sufficiently well. The statistics of the equivalent noise are non-stationary in general. This fundamental assumption converts the problem of maneuvering target tracking to that of state estimation in the presence of non-stationary process noise. Almost all of the equivalent-noise approaches are limited to linear systems. In this work, the equivalent-noise approach is extended to the nonlinear and non-Gaussian systems. Particle filter methods are utilized to identify the non-stationary process noise.

The novelty of the proposed maneuvering target tracking method based on noise identification is that the posterior distribution of the process noise is not parametrically available and/or a priori fixed, but dynamically approximated using the particle filter algorithm. The process noise is modeled as a dynamic system and the state vector of the noise system is chosen as the noise vector. At the beginning of each time step, a set of process noise samples are drawn from a uniform distribution, which is noninformative and assumed as the prior distribution of the process noise system. The process noise samples are evaluated by the likelihood function including current measurements and are assigned with corresponding weights. The posterior distribution of the process noise system is approximated by the process noise samples and their associated weights. A new set of process noise samples are then generated from the approximate posterior distribution of process noise at the resampling stage. A standard particle filter for state estimation is then run using the new distributed process noise samples.

To deal with the data association between the targets and the available observations from observers, the particle filter based multi-scan JPDA filter is adopted. In the proposed approach, the distributions of interest are the marginal filtering distributions for each of the targets, which is approximated with particles. The multi-scan JPDA filter algorithm examines the joint association hypothesis in a multi-scan sliding window and calculates the posterior marginal probability based on the multi-scan joint association hypothesis. Compared with the single scan JPDA method, the multi-scan JPDA method uses richer information, which results in better estimated probabilities.

The proposed multiple maneuvering target tracking algorithm is a combination of the process noise identification method for modeling highly maneuvering target (introduced in Section 2.2), and the multi-scan JPDA algorithm for solving data association problem (introduced in Section 3.1), in a particle filter framework. The process noise identification process is effective in estimating both the maneuvering movement and the random acceleration of the target, avoiding the use of complicated multiple model approaches. The multi-scan JPDA is effective in maintaining the tracks of multiple targets using multiple scan information. The proposed algorithm is illustrated with an example involving tracking of two highly maneuvering, at times closely spaced and crossed, targets.

The particle filter based process noise identification method for tracking highly maneuvering target is introduced in Section 2.2. The data association method based on multi-scan JPDA is discussed in Section 3.1. The proposed multiple maneuvering target tracking algorithm, which is a combination of the above two algorithms, is introduced in Section 4.2. The simulation results of tracking two highly maneuvering targets are shown in Section 4.3. The conclusions are drawn in Section 4.4.

4.2 Multiple Maneuvering Target Tracking Algorithm

The proposed multiple maneuvering target tracking algorithm is a combination of the two methods proposed respectively in Section 2.2 and Section 3.1. The process noise identification process is effective in estimating both the maneuvering movement and the random acceleration of the target, while the multi-scan JPDA is effective in maintaining the track of multiple targets using multiple scan information. In the proposed multiple maneuvering target tracking algorithm, at the beginning of each time step, for each target model, the process noise identification method is used to estimate the maneuvering movement with random acceleration using one general model, then the particles of each target model are propagated to the next time step based on the new distributed process noise samples to obtain the predicted particles. In the update process, each predicated particle of one target model is assigned with a weight based on the multi-scan JPDA process. The steps involved in the proposed multiple maneuvering target tracking algorithm are listed in the following:

Algorithm 4.1: Multiple Maneuvering Target Tracking

1. At time step $k-1$, for $m = 1 \cdots M$,

   (a) Draw process noise samples $\{\hat{v}^j_{m,k-1} : j = 1, \ldots, NP\}$ from a uniform distribution $U(-d, d)$.

   (b) Calculate the intermediate particles $\{\mu^j_{m,k} : j = 1, \cdots, NP\}$ according to (4.1),

   $$\mu^j_{m,k} = f(\tilde{\tilde{x}}_{m,k-1}, \hat{v}^j_{m,k-1}). \tag{4.1}$$

   (c) Calculate the process noise sample weights $\{iw^j_{m,k} : j = 1, \cdots, NP\}$ as per (4.2) and normalize each weight.

   $$iw^j_{m,k} = p(z_k|\mu^j_{m,k}). \tag{4.2}$$

   (d) Resample the discrete distribution $\{iw^j_{m,k} : j = 1, \cdots, NP\}$, NP times to generate the new process noise samples $\{v^i_{m,k-1} : i = 1, \cdots, NP\}$, so that for any i, $Pr\{v^i_{m,k-1} = \hat{v}^j_{m,k-1}\} = iw^j_{m,k}$. Set the weights $iw^j_{m,k}$ to $\frac{1}{NP}$, $i = 1, ..., NP$.

   (e) Obtain the predicted particles $\{\hat{x}^i_{m,k} : i = 1, \cdots NP\}$ at time step k from the new process noise samples as per (4.3),

   $$\hat{x}^i_{m,k} = f(x^i_{m,k-1}, v^i_{m,k-1}). \tag{4.3}$$

2. At time step k, for $m = 1 \cdots M$, calculate $\tilde{x}_{m,k}$, the pre-approximation of $x_{m,k}$,
$$\tilde{x}_{m,k} = \sum_{i=1}^{NP} i w^i_{m,k} \hat{x}^i_{m,k}. \tag{4.4}$$

3. Enumerate all the valid joint measurement to target association events in the sliding window $k - L + 1 : k$ to form the set $\Lambda_{k-L+1:k}$.

4. For each $\lambda_{k-L+1:k} \in \Lambda_{k-L+1:k}$, compute the posterior probability of the joint association event of multiple scan,
$$\begin{aligned} p(\lambda_{k-L+1:k}|z_{1:k}) \quad \propto \quad & p(\lambda_{k-L+1:k}) \textstyle\prod_{s=1}^{L} [(V)^{-NC_{k_s}} \\ & \cdot \textstyle\prod_{j=1}^{NT_{k_s}} p(z_{j,k_s}|x_{\lambda_{j,k_s},k_s})]. \end{aligned} \tag{4.5}$$

5. For $m = 1 \cdots M, j = 1 \cdots N_k$, compute the marginal association posterior probability,
$$\begin{aligned} & \beta_{jm} \\ & = p(\lambda_{j,k} = m|z_{1:k}) \\ & = \textstyle\sum_{\{\lambda_{k-L+1:k} \in \Lambda_{k-L+1:k} \ : \ \lambda_{j,k}=m\}} p(\lambda_{k-L+1:k}|z_{1:k}). \end{aligned} \tag{4.6}$$

6. For $m = 1 \cdots M, i = 1 \cdots NP$, compute the target likelihood,
$$p_m(z_k|\hat{x}^i_{m,k}) = \beta_{0m} + \sum_{j=1}^{N_k} \beta_{jm} p(z_{j,k}|\hat{x}^i_{m,k}). \tag{4.7}$$

7. For $m = 1 \cdots M, i = 1 \cdots NP$, compute and normalize the particle weights,
$$\begin{aligned} & w^i_{m,k} \propto w^i_{m,k-1} \tfrac{p_m(z_k|\hat{x}^i_{m,k}) p_m(\hat{x}^i_{m,k}|x^i_{m,k-1})}{q_m(\hat{x}^i_{m,k}|x^i_{m,k-1}, z_k)}, \\ & \textstyle\sum_{i=1}^{NP} w^i_{m,k} = 1. \end{aligned} \tag{4.8}$$

8. Calculate $\tilde{\tilde{x}}_{m,k}$, the estimate of the true state $x_{m,k}$,
$$\tilde{\tilde{x}}_{m,k} = \sum_{i=1}^{NP} w^i_{m,k} \hat{x}^i_{m,k}. \tag{4.9}$$

9. Resample the discrete distribution $\{w^i_{m,k} : i = 1, \cdots, NP\}$ NP times to generate particles $\{x^j_{m,k} : j = 1, \cdots, NP\}$, so that for any j, $Pr\{x^j_{m,k} = \hat{x}^i_{m,k}\} = w^i_{m,k}$. Set the weights $w^i_{m,k}$ to $\frac{1}{NP}$, $i = 1, ..., NP$, and move to Stage 1.

Discussion

In contrast to (3.26), in (4.4) the pre-approximation of the true state, $\tilde{x}_{m,k}$, is calculated based on the weights of the process noise samples, $\{iw^i_{m,k} : i = 1, \cdots, NP\}$, and the predicted particles, $\{\hat{x}^i_{m,k} : i = 1, \cdots, NP\}$. In Section 3.1, the targets are assumed to perform nearly constant movements and the process noise of each target is assumed with known posterior distribution and small variance. The value of the ith predicted particle $\hat{x}^i_{m,k}$ does not change much compared with the ith particle $x^i_{m,k-1}$ from the previous time step. The predicted particles $\{\hat{x}^i_{m,k} : i = 1, \cdots, NP\}$ could be assumed approximately distributed according to $\{w^i_{m,k-1} : i = 1, \cdots, NP\}$, which leads to (3.26). However, in Section 4.2, the targets perform highly maneuvering movements and the equivalent process noise of each target is with unknown distribution. $\hat{x}^i_{m,k}$ and $x^i_{m,k-1}$ differ largely. From Stage 1d, it can be seen that the new process noise samples, $\{v^i_{m,k-1} : i = 1, \cdots, NP\}$, are distributed according to $\{iw^i_{m,k} : i = 1, \cdots, NP\}$ and the predicted particles $\{\hat{x}^i_{m,k} : i = 1, \cdots, NP\}$ are obtained form the new process noise samples (4.3). In (4.3), $\{x^i_{m,k-1} : i = 1, \cdots, NP\}$ are less variable compared to the new process noise samples. It could be assumed that $\{\hat{x}^i_{m,k} : i = 1, \cdots, NP\}$ are approximately distributed according to $\{iw^i_{m,k} : i = 1, \cdots, NP\}$. It can be concluded that in tracking maneuvering target condition, $\tilde{x}_{m,k}$ calculated via (4.4) is closer to the true state than that via (3.26).

4.3 Simulation Results and Analysis

We now consider tracking two highly maneuvering targets in clutter. The true trajectories and velocities of the targets are respectively shown in Fig. 4.1 and Fig. 4.2 in $x - y$ plane. The distance between the two targets as a function of time is shown in Fig. 4.3. Target one starts at location $[-310\ 310]$ in Cartesian coordinates in meters with the initial velocity (in m/s) $[10\ \ -400]$. Its trajectory is same with the example trajectory proposed in Section 2.2.5. Target two starts at location $[-310\ \ -20310]$

in Cartesian coordinates in meters with the initial velocity (in m/s) [10 400]. Its trajectory is: a straight line with constant velocity between 0 and 17 s, a coordinated turn ($0.09rad/s$) between 17 and 34 s, and a straight line constant velocity between 34 and 100 s.

A track-while-scan (TWS) radar is positioned at the origin of the plane (refer to Section 2.2.5).

The sampling interval is $\Delta T = 1s$ and it is assumed that the probability of detection $P_D = 0.9$ for the radar. For generating measurements in simulations, the clutter is assumed uniformly distributed with density $1 \times 10^{-6}/m^2$.

In the proposed method, each target model is assigned with 1000 particles. The length of the multiple scan sliding window (L) is chosen as 3. The bound of the process noise (d), which accounts for the uncertain dynamics, is chosen as $\{20\ m, 20\ m/s, 10\ m/s^2, 20\ m, 20\ m/s, 10\ m/s^2\}$. The algorithm is initialized with Gaussians around the initial states of the true targets.

A comparison to one of the IMMJPDA methods [104], is made in this work. The details regarding the IMMJPDA method may be found in [104].

The simulation results are obtained from 100 Monte Carlo runs. Fig. 4.4$\sim$ Fig. 4.6 and Fig. 4.7 $\sim$ Fig. 4.9 show one realization respectively performed by the IMMJPDA filter and the proposed method. The RMSE (respectively in position and velocity) at each time step for the two methods are shown in Fig. 4.10 $\sim$ Fig. 4.13. The performance of the two methods are also compared via global RMSE (in position and velocity), executing time, tracking loss rate and swap rate, which are listed in Table. 4.1. Moreover, for the proposed algorithm, the influence of particle number in its performance is studied and simulations are carried out based on different sample sizes. The results are listed in Table 4.2.

The simulation results show that the proposed algorithm is more accurate and robust than the popular IMMJPDA method, though it takes longer computing time. However, both the algorithms are implemented using Matlab. The computing time is considerably reduced when coded in C++. For the proposed method, it takes about 0.8 second for one time step in C++. In the scene of simulation, at the period of 30 $s \sim 50$ s, the two targets are very near to each other, which easily results in track swap. In the IMMJPDA method, the judgement of which measurements belong to which target depends on the information from current scan. If the measurements from two targets are very close, it is hard to distinguish the different targets based on the measurements from single scan, which leads to track swap. In the proposed method, the information from several previous scans is combined with the information from current scan to calculate the association probabilities. The decision of which measurements belong to which target based on the information from multiple scans will be more accurate than single scan method, which reduces the swap rate effectively. From Table 4.2, it can be seen that when the number of particles is increased, the performance of the proposed algorithm also increases. But when the number of particles exceeds some threshold (1000), the increase in the performance slows down.

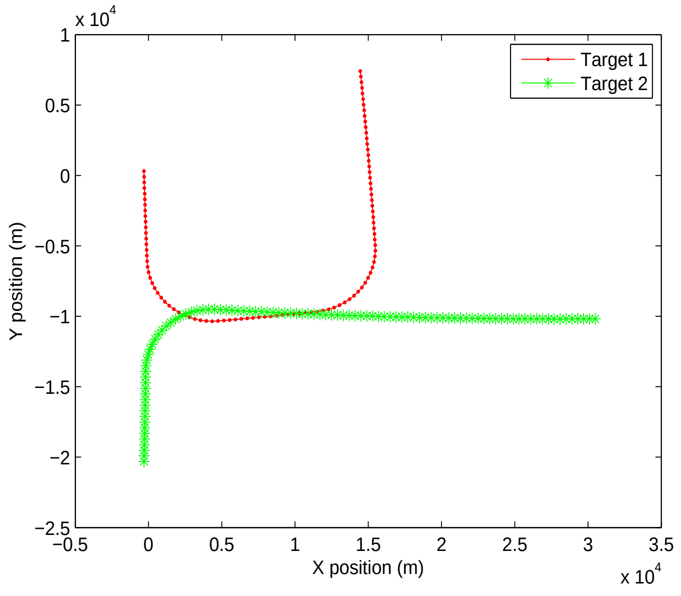

Figure 4.1: True trajectories of maneuvering targets

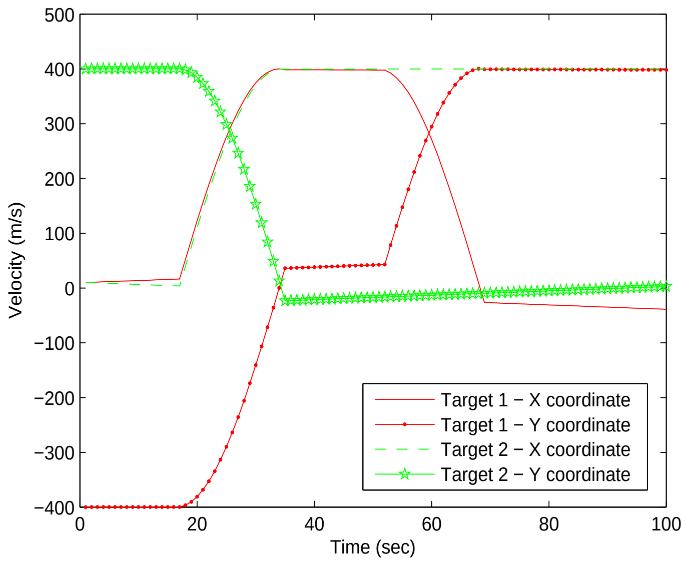

Figure 4.2: True velocities of maneuvering targets

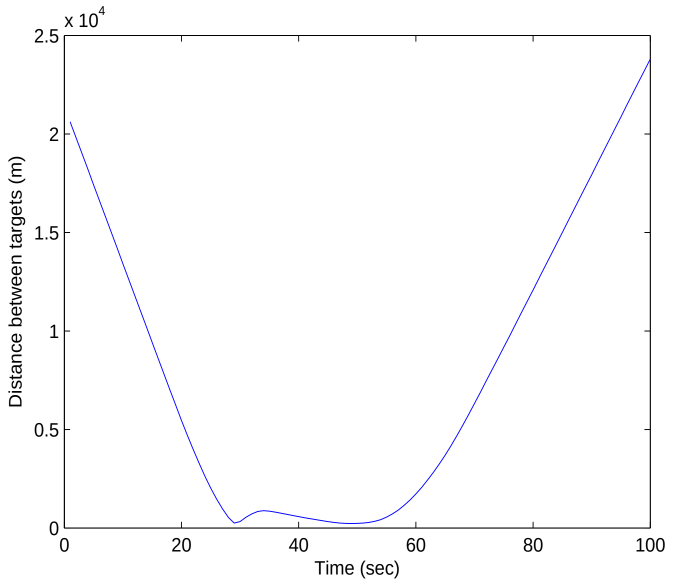

Figure 4.3: Distance between the targets

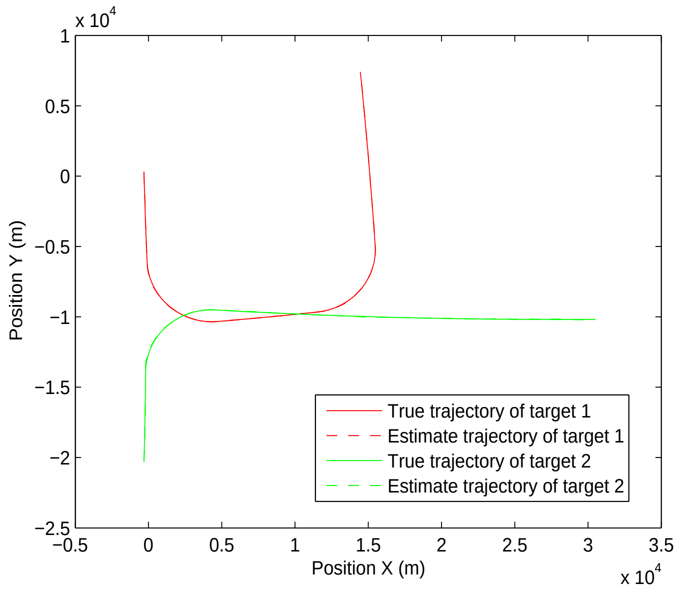

Figure 4.4: True and estimate trajectories of two maneuvering targets using IMMJPDA method

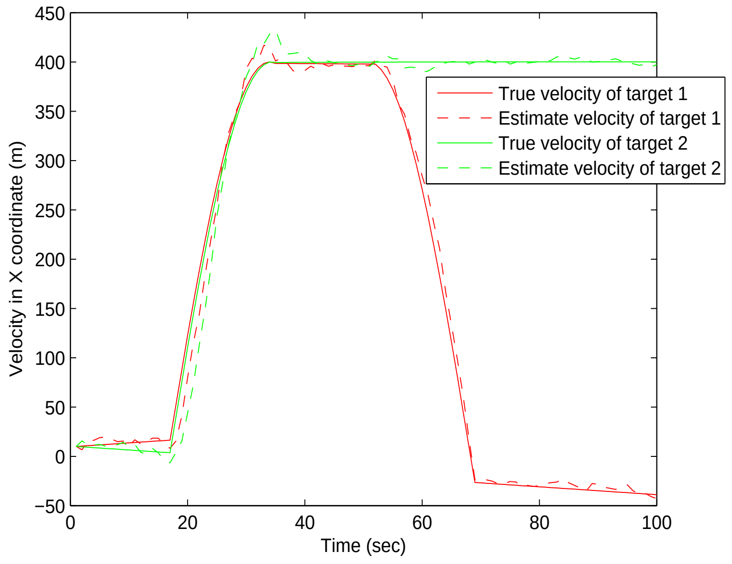

Figure 4.5: True and estimate velocities in X coordinate of two maneuvering targets using IMMJPDA method

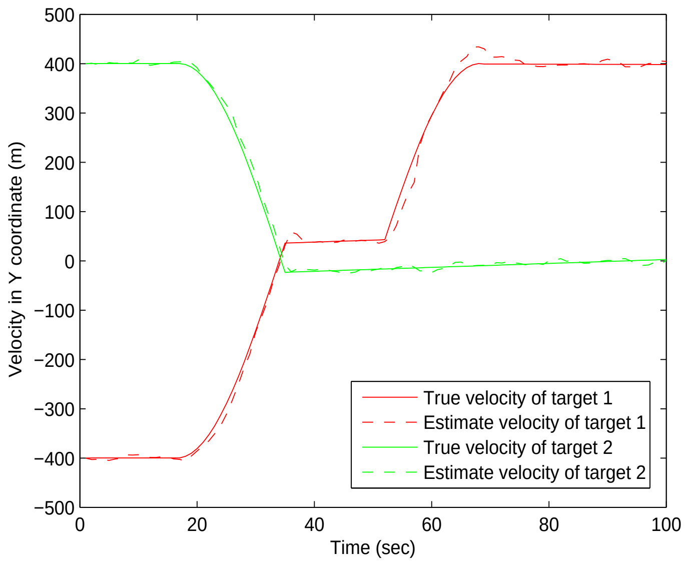

Figure 4.6: True and estimate velocities in Y coordinate of two maneuvering targets using IMMJPDA method

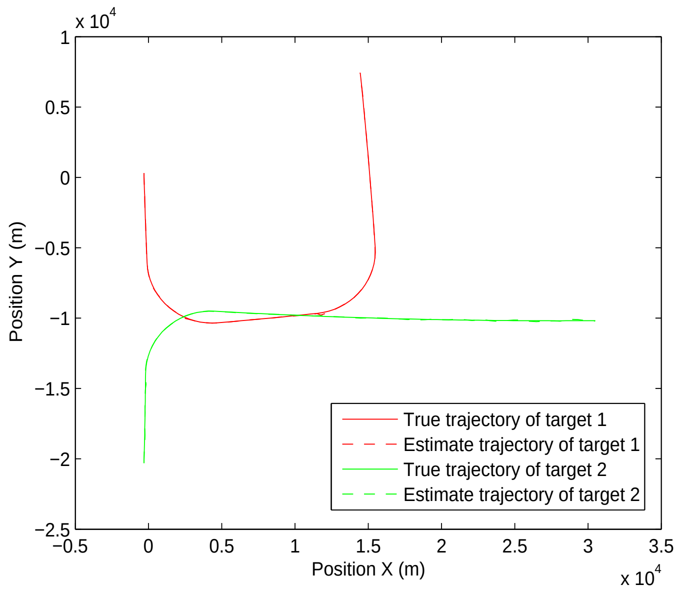

Figure 4.7: True and estimate trajectories of two maneuvering targets using the proposed method

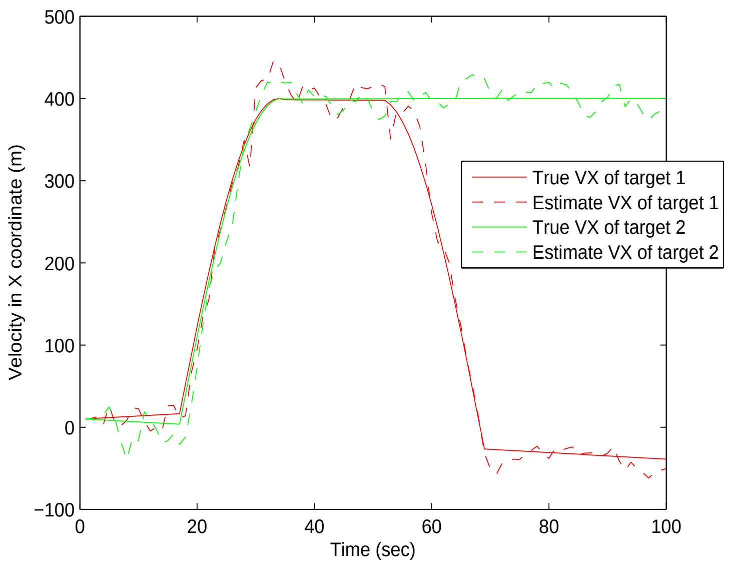

Figure 4.8: True and estimate velocities in X coordinate of two maneuvering targets using the proposed method

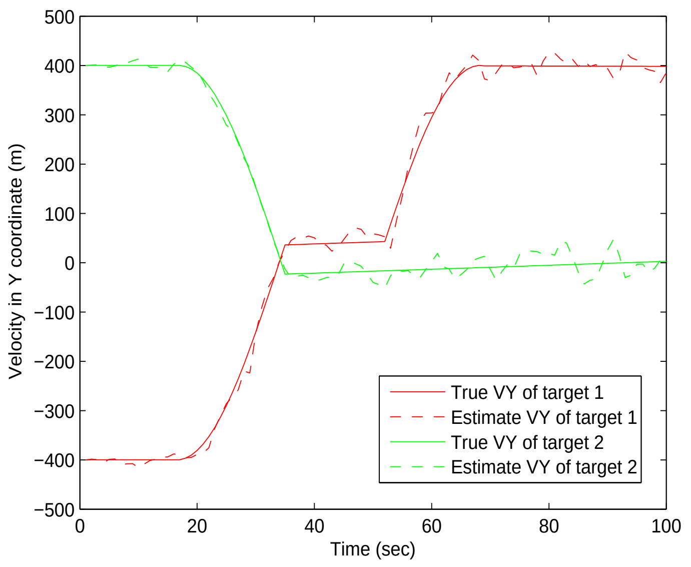

Figure 4.9: True and estimate velocities in Y coordinate of two maneuvering targets using the proposed method

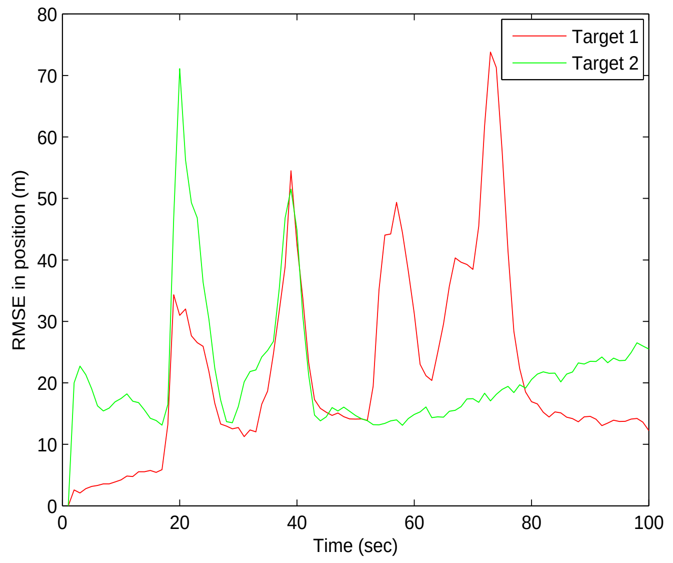

Figure 4.10: RMSE in position using IMMJPDA method

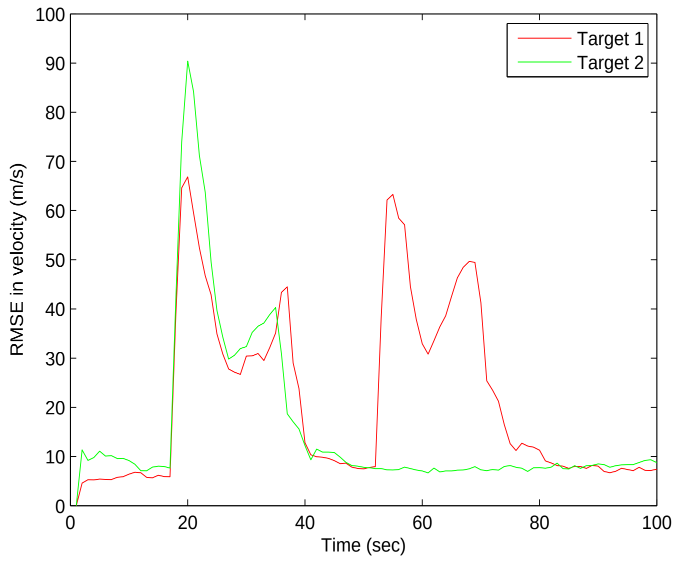

Figure 4.11: RMSE in velocity using IMMJPDA method

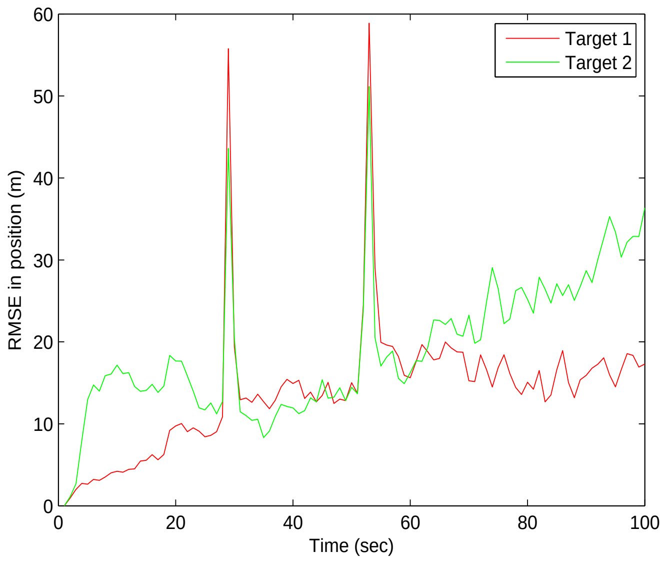

Figure 4.12: RMSE in position using the proposed method

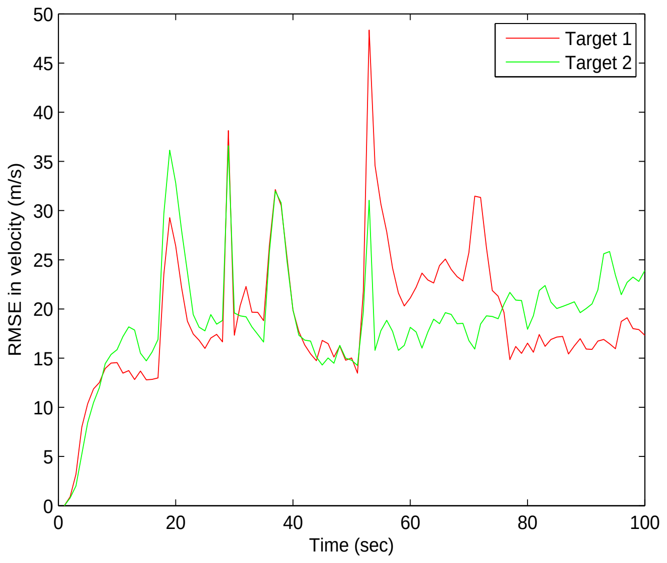

Figure 4.13: RMSE in velocity using the proposed method

Table 4.1: Performance Comparison

	RMSE in Position (m)	RMSE in Velocity (m/s)	Executing Time (s)	Tracking Loss Rate	Swap Rate
IMMJPDA Method	Target1: 26.5050, Target2: 23.8984	Target1: 28.2778, Target2: 23.4672	0.7565	10%	15%
Proposed Method	Target1: 16.3363, Target2: 20.9777	Target1: 20.2826, Target2: 19.8747	4.095	0	5%

Table 4.2: Influence of Particle Number in the Performance of the Proposed Algorithm for Tracking Multiple Maneuvering Target

Number of Particles	RMSE in Position (m)	Executing Time (s)	Tracking Loss Rate
200	NA	NA	100%
500	Target1: 18.4532, Target2: 22.671	2.9375	37%
1000	Target1: 16.3363, Target2: 20.9777	4.095	0
2000	Target1: 16.1285, Target2: 20.3747	20.2346	0

4.4 Conclusions

A new algorithm is proposed for the multiple maneuvering target tracking in particle filter framework. In order to track a highly maneuvering target, the particle filter based process noise identification method is proposed to estimate the equivalent process noise induced by both the maneuvering and random acceleration. The process noise is modeled as a dynamic system and the state vector of the noise system is chosen as the noise vector. The posterior distribution of the target noise state is dynamically approximated using the particle filter algorithm. Compared with the multiple model based methods for maneuvering target tracking, only one general model is adopted in the proposed method. In order to tackle the data association problem in multiple maneuvering target tracking, the particle filter based multi-scan JPDA filter is adopted. The marginal filtering distributions for each of the targets is approximated with particles. The proposed algorithm examines the joint association hypothesis in a multi-scan sliding window and calculates the marginal posterior probability based on the multi-scan joint association hypothesis. Compared with the single scan JPDA method, the multi-scan JPDA method uses richer information, which results in better estimated probabilities.

Chapter 5

A Random Object Tracking System Based on Multi-sensor Fusion

5.1 Introduction

The multi-sensor object tracking system has been widely used in different fields such as surveillance, automated guidance systems, and robotics in general. As robots are being deployed in everyday human environments, they have to perform increasingly interactive navigational tasks, such as leading, following, intercepting and avoiding obstacles. Object tracking has become an ubiquitous elementary task in the mobile robot application.

Recently particle filter methods have become popular tools to solve the tracking problem. The popularity stems from their simplicity, flexibility and ease of implementation, especially the ability to deal with non-linear and non-Gaussian estimation problem, which is a challenging one in multi-sensor object tracking applications. Particle filter uses sequential Monte Carlo methods for on-line learning within a Bayesian framework and can be applied to any state-space models. It represents the required posterior distribution by a scatter of particles which propagate through state space. The propagation and adaptation rules are chosen so that the combined weight of particles in a particular region approximates the integral of the posterior distribution over that region. A detailed introduction to particle filter is available in [30].

One important advantage of the particle filter framework is that it allows the information from different measurement sources to be fused in a general framework. In the literature, there are two main research applications of particle filter in sensor fusion: a) for sensor management [71], [120], [121] and b) for object tracking based on fused information [122], [123], [124], [125], [65]. In [122], the particle filter and support vector machine methods together provide a means of solving the distributed data fusion problem within a Bayesian framework. In [123], particle filter is used to provide a framework for integrating visual cues and maintain multiple hypotheses of target location in $3D$ space. In [124], particle filter is used to track manoeuvring targets in environments with clutter noise. The observation system is based on the assumption that the system is linear and Guassian. In [125] and [65], particle filter combined with Gibbs sampler is applied to track multiple moving objects. The targets are assumed to move at nearly constant velocities.

In this work, a particle filter based tracker that fuses color and sonar cues in a novel way is presented. More specifically, color is utilized as the main visual cue and is fused with sonar localization cues. The generic objective is to track a randomly moving object via the pan-tilt camera and sonar sensors installed on a mobile robot. When moving randomly, the object's position and velocity vary quickly and are hard to track. This leads to serious sample impoverishment in particle filter and then the tracking algorithm fails. An improved particle filter with a new resampling algorithm is proposed to tackle this issue.

The proposed algorithm is implemented on a mobile robot, which is equipped with pan-tilt camera and sonar sensors. The mobile robot continuously follows the object with the help of the pan-tilt camera by keeping the object at the center of the image.

The robot is capable of continuously tracking a human's random movement at walking rate. Several similar experiments are reported in [126] ,[127] and [128], where the first two deals with the mobile robot tracking problem, and the last one deals with the sensor fusion problem. [126] investigates visual head tracking and person following with a mobile robot. It is mainly concerned with human head tracking by using skin color cues inside and around simple silhouette shapes in the context of face. Since an existing color histogram is used to model the skin color, the algorithm is unable to deal with other moving objects with different colors and shapes. However, the proposed algorithm can be applied to different color objects since the reference color model is constructed during the tracking process through the automatic object detection module. [127] describes one solution for the problem of pursuit of objects moving on a plane by using a mobile robot and an active vision system. This approach deals with the interaction of different control systems using visual feedback and it is accomplished by the implementation of a visual gaze holding process interacting cooperatively with the control of the trajectory of a mobile robot. [127] focuses on system integration and controller design while choosing simple $\alpha - \beta$ tracker assuming that the target has uniform acceleration. However, our method uses the particle filter based tracker to tackle the random movement of the object. [128] presents a particle filter based visual tracker that fuses three cues in a novel way: color, motion and sound. The generic importance sampling mechanism is introduced for data fusion and applied for fusing color either with stereo sound (for teleconferencing) or with motion cues (for surveillance) using a still camera. However, the pan-tilt camera used in our work will be more effective if it is used for teleconferencing or for surveillance: the surveillance area could be expanded via the camera's panning and tilting movements, and the object of interest could always be centered within the image frame in teleconferencing.

The rest of this chapter is organized as follows. The sensor fusion tracker based on particle filter is introduced in section 5.2. In Section 5.3, the improved resampling algorithm is introduced. Experimental results on a mobile robot are provided in

Section 5.4 and conclusions are drawn in Section 5.5.

5.2 Sensor Fusion Tracker

A tracker that fuses color and sonar cues is presented in Fig. 5.1. The inputs of the sensor fusion system are the color cues extracted from sequence of captured images and the sonar localization cues, obtained by measuring the distance between the robot and the moving object. The color localization cues are used to locate the moving object within the image plane. The color cues are represented via the dichromatic r-g-b-color space ($r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B}, b = \frac{B}{R+G+B}$), which is independent from variations in luminance. R, G and B denote the primary colors Red, Green and Blue. For each pixel in the image, its RGB values are read from the captured image frame. The $r - g - b$ color cues are remarkably persistent and robust to changes in pose and illumination. They are, however, more prone to ambiguity, especially if the scene contains other objects characterized by a color distribution similar to that of the object of interest. The sonar localization cues are very discriminant and can be used to compensate for the color cues.

Both the color cues and the sonar cues are verified by the reference model during the tracking process. The reference model is associated with the moving object and presents some characteristics of it (color), which are obtained from the initial automatic detection module introduced in Section 5.2.1.

The outputs of the sensor fusion system are the estimates of the object's center position and size in the image plane, and its distance to the mobile robot.

In the particle filter framework, the outputs of the sensor fusion system are chosen as the state vector, and the reference model variables are chosen as the observation variables. A likelihood model is constructed for each of the color cues. These models are assumed mutually independent, considering that any correlation that may exist between the color and distance of an object is likely to be weak.

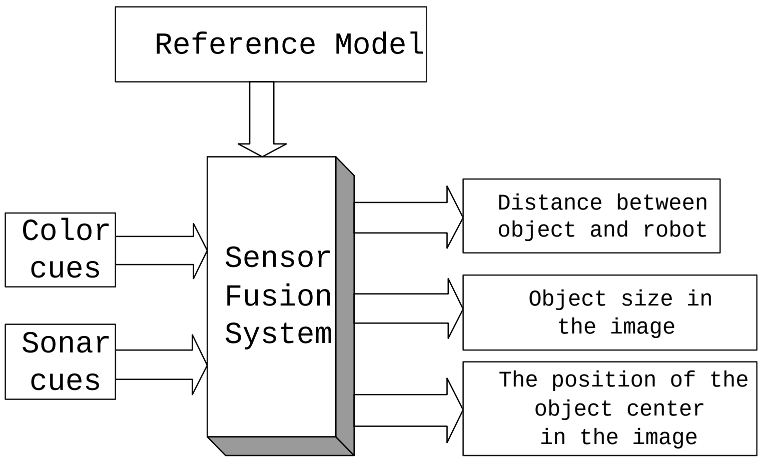
Reference Model
Color cues
Sonar cues
Sensor Fusion System
Distance between object and robot
Object size in the image
The position of the object center in the image

Figure 5.1: Sensor fusion system

5.2.1 Moving Object Detection Module

In this section, we describe the procedures of detecting a moving object through a sequence of images taken by a stationary pan-tilt camera, and then obtaining the reference model associated with the moving object.

Initially, the pan-tilt camera is kept stationary. Sequence of images taken by the camera are transformed to gray images and processed by image differencing method. The absolute value substraction (pixel by pixel) of two gray images respectively at time step $k + 1$ and k generates the image of differences. Noises are reduced via median filter in the differences image. If there is no moving object in the scene, the intensity of each pixel in the differences image remains nearly constant in consecutive frames. From frame to frame, the pixel intensity sum of the differences image vary around a range of values. In the experiments conducted, 100 differences images are used to estimate the range of the pixel intensity sum. The upper limit of the pixel intensity sum is chosen as the threshold to detect the moving object, which is defined as T_{moving}. When the object begins moving, the two images captured just before and after the beginning of the motion, will have large difference in their pixel intensity distributions. So the resulted differences image will have a large pixel intensity sum, which will exceed the threshold T_{moving}, resulting in the detection of the motion of the object.

A labelling method is used in the differences image to connect the components. The classic labelling method is used, which makes only two passes through the image, but requires a large global table to record the equivalences. The moving object is then identified by selecting the largest component in the differences image. Then in the corresponding color image, the average colors of the pixels within the moving object region, (r_f, g_f, b_f), are obtained and are used to compose the reference model. The initial state vector is then generated which comprises of the initial object center, the initial height and width of the moving object in the image plane, and the distance between the robot and the moving object. Since 16 sonar sensors constitute a 360 degree description of the robot's surroundings, it is possible to assign one sonar sensor to a specific point in the image plane (explained in Section 5.4.2). Once the initial object in the image is detected, the sonar sensor corresponding to that is identified and the distance to the object is measured. The camera then pans and tilts to center the moving object within the image plane.

The image differencing method is not suitable when the camera begins to move. The tracking process based on particle filter is then initiated as described in Section 5.2.2.

5.2.2 Particle Filter Based Sensor Fusion Tracker

In the particle filter based tracking system, the state vector is chosen as $\chi_k = [\Delta x, \Delta y, h, l, d]_k^T$, where Δx and Δy are the x and y components of the distance between the center of the moving object, $\{cx_{obj}, cy_{obj}\}$, and the center of the image, $\{cx_{img}, cy_{img}\}$, both in the image coordinate frame. h and l are the height and width of the rectangle bounding box of the object in the image plane. d is the distance between the robot and the moving object. k denotes time step k and T denotes transpose. The NP particles are defined as $\{S_k^i = [\Delta x^i, \Delta y^i, h^i, l^i, d^i]_k^T, i = 1, \cdots, NP\}$, and each particle is corresponding to a candidate region in the image, which centers at (cx^i_{obj}, cy^i_{obj}), with h^i and l^i as its height and width respectively. $(cx^i_{obj\_k}, cy^i_{obj\_k})$ can be obtained via (5.1) and (5.2),

$$cx^i_{obj\_k} = \Delta x^i_k + cx_{img}, \tag{5.1}$$

$$cy^i_{obj\_k} = \Delta y^i_k + cy_{img}. \tag{5.2}$$

The candidate image region corresponding to the *ith* particle is defined as the *ith* particle-object.

At each time step, the particle filter outputs the estimated state vector $[\widetilde{\Delta x}, \widetilde{\Delta y}, \tilde{h}, \tilde{l}, \tilde{d}]^T_k$ based on all the particles $\{S^i_k = [\Delta x^i, \Delta y^i, h^i, l^i, d^i]^T_k, i = 1, \cdots, NP\}$ and their associated weights $\{w^i_k, i = 1, \cdots, NP\}$ as per (5.3) $\sim$ (5.7).

$$\widetilde{\Delta x}_k = \Sigma^{NP}_{i=1} \Delta x^i_k w^i_k \tag{5.3}$$

$$\widetilde{\Delta y}_k = \Sigma^{NP}_{i=1} \Delta y^i_k w^i_k \tag{5.4}$$

$$\tilde{h}_k = \Sigma^{NP}_{i=1} h^i_k w^i_k \tag{5.5}$$

$$\tilde{l}_k = \Sigma^{NP}_{i=1} l^i_k w^i_k \tag{5.6}$$

$$\tilde{d}_k = \Sigma^{NP}_{i=1} d^i_k w^i_k \tag{5.7}$$

The estimate of the center of the moving object in image coordinate frame is:

$$\widetilde{cx_{obj\_k}} = \widetilde{\Delta x}_k + cx_{img}, \tag{5.8}$$

$$\widetilde{cy_{obj\_k}} = \widetilde{\Delta y}_k + cy_{img}. \tag{5.9}$$

According to [127], the pan angle, $\Delta\theta_x$, and tilting angle, $\Delta\theta_y$, at time k by which the camera pans and tilts to center the object within the image is:

$$\Delta\theta_{x\_k} = \frac{\widetilde{cx_{obj\_k}} - cx_{img}}{S_x f} = \frac{\widetilde{\Delta x}_k}{S_x f}, \tag{5.10}$$

$$\Delta\theta_{y\_k} = \frac{\widetilde{cy_{obj\_k}} - cy_{img}}{S_y f} = \frac{\widetilde{\Delta y}_k}{S_y f}, \tag{5.11}$$

where S_x and S_y are the scale factors for the x and y-axes respectively, and f is the camera focal length. At each time step, the particle filter output state variables, (5.3) and (5.4), are the inputs to the pan-tilt camera controller, and there by, the vision system and pan-tilt camera control system are integrated effectively. The observation vector is defined as $z_k = [r_f, g_f, b_f, d_f]_k^T$, where $[r_f, g_f, b_f]_k^T$ is obtained from the reference model and $d_{f\_k} = \tilde{d}_{k-1}$. The value of d_{f_k} is chosen considering that the distance between the moving object and the robot do not change much between two consecutive frames.

The tracking process based on particle filter begins with the initialization stage. Particles are drawn around the initial state vector $[\Delta x, \Delta y, h, l, d]_0^T$, which is obtained through the moving object detection procedure.

In the prediction stage, the particles are propagated through the dynamic model $\chi_k = \Phi\chi_{k-1} + \nu_k$, where Φ is the transition matrix representing dynamic characteristics of the randomly moving object and $\nu_k = [\nu_{\Delta x}, \nu_{\Delta y}, \nu_h, \nu_l, \nu_d]_k^T$ is the zero mean Gaussian white noise process with covariance $Q : E[\nu_k\ \nu_j^T] = Q\delta_{jk}$, where,

$$Q = \begin{bmatrix} \sigma_{\Delta x}^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{\Delta y}^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_h^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_l^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_d^2 \end{bmatrix}. \tag{5.12}$$

$\sigma_{\Delta x}$ is the standard deviation associated with the Δx component of the state vector χ, and the similar definitions hold for the other standard deviations.

In the update stage, the likelihood models, respectively for the color cues and sonar distance cues, are constructed and the weight of each particle is obtained through the product of these two likelihoods.

The color cues likelihood model is constructed by comparing the average colors of the pixels within the particle-object region with the color reference model (from the observation vector). The smaller the discrepancy between the particle-object color and reference color model, the higher the probability for the particle-object being a "true" object.

The color cues likelihood model for the *ith* particle-object is represented as:

$$p^i_{color\_k} = p(r_{f\_k}, g_{f\_k}, b_{f\_k} | \bar{r}^i_k, \bar{g}^i_k, \bar{b}^i_k), \tag{5.13}$$

where $(\bar{r}^i_k, \bar{g}^i_k, \bar{b}^i_k)$ are the average colors of the pixels within the *ith* particle-object.

The size of the moving object changes largely during tracking process. To tackle this issue, a small Gaussian color model is placed around each pixel of the particle-object, and the size of the particle-object is tailored based on the color evaluation of pixels. The color of the *jth* pixel of *ith* particle-object at time k is represented as $(r^{i,j}_{p\_k}, g^{i,j}_{p\_k}, b^{i,j}_{p\_k})$. A weight $w^{i,j}_{p\_k}$ is assigned to the *jth* pixel of the *ith* particle-object as:

$$w^{i,j}_{p\_k} = N(|r_{f\_k} - r^{i,j}_{p\_k}|; 0, \sigma^2_r) \cdot N(|g_{f\_k} - g^{i,j}_{p\_k}|; 0, \sigma^2_g) \cdot N(|b_{f\_k} - b^{i,j}_{p\_k}|; 0, \sigma^2_b), \tag{5.14}$$

where, N represents the Gaussian distribution and, σ^2_r, σ^2_g and σ^2_b represent respectively the variances of the $r^{i,j}_{p\_k}, g^{i,j}_{p\_k}$ and $b^{i,j}_{p\_k}$ variables. Using the general variance calculation method [129], σ^2_r, σ^2_g and σ^2_b are calculated based on the pixels within the initial object region in the color image (Section 5.2.1).

From (5.14), it is observed that $w^{i,j}_{p\_k}$ is always positive and the smaller the discrepancy between the candidate pixel color and reference color models, the larger the $w^{i,j}_{p\_k}$ is. A threshold T_{pixel} is set to sort the pixels: the pixels with weights larger than T_{pixel} are retained as "true" pixels and others are eliminated. The choice of T_{pixel} is a compromise: too large a value would lead to the loss of "true" pixels while too small a value would misunderstand "wrong" pixels as "true" pixels. In this experiment, T_{pixel} is chosen experimentally and is set as 0.5. The remaining pixels with weights larger than T_{pixel} are labeled and bounded by a rectangular box, with (h^i_k, l^i_k) being replaced by the size of the *ith* tailored particle-object.

The tailored particle-objects are sorted based on their sizes in comparison to the estimated size $(\tilde{h}_{k-1}, \tilde{l}_{k-1})$ from the previous time step assuming that the object sizes do not change much between two consecutive frames.

$$\begin{cases} ith\ particle\text{-}object\ is\ retained & IF\ |h_k^i - \tilde{h}_{k-1}| < T_{\Delta h}\ \ AND\ \ |l_k^i - \tilde{l}_{k-1}| < T_{\Delta l}, \\ ith\ particle\text{-}object\ is\ eliminated & Otherwise, \end{cases} \tag{5.15}$$

where $T_{\Delta h}$ and $T_{\Delta l}$ are the thresholds for the object size difference between two consecutive frames. The values of of $T_{\Delta h}$ and $T_{\Delta l}$ are set experimentally (5.16).

$$\begin{cases} T_{\Delta h} = 0.1 * \tilde{h}_{k-1} \\ T_{\Delta l} = 0.1 * \tilde{l}_{k-1} \end{cases} \tag{5.16}$$

The color cues likelihood model can then be represented as:

$$p^i_{color\_k} = \begin{cases} \frac{1}{M^i_{p\_k}} \sum_{j=1}^{M^i_{p\_k}} w^{i,j}_{p\_k} & \text{IF ith particle-object is retained,} \\ 0 & \text{IF ith particle-object is eliminated,} \end{cases} \tag{5.17}$$

where, $M^i_{p\_k}$ is the number of remaining pixels of the retained *ith* particle-object at time step k. The $p^i_{color\_k}$ of the eliminated particle-objects are set to zeros, which reduces the number of particles effectively.

The sonar distance cues likelihood model is represented as:

$$p^i_{distance\_k} = N(|d_{f\_k} - d^i_k|; 0, \sigma_d^2), \tag{5.18}$$

where d_{f_k} is obtained from the observation vector, d^i_k is the distance between the robot and the *ith* particle-object, and, σ_d^2 is the variance of the distance variable d^i_k.

The weight of the *ith* particle at time k is obtained through the product of the two likelihoods based on the assumption that the two likelihood models are mutually independent.

$$w^i_k = p^i_{color\_k} \cdot p^i_{distance\_k} \tag{5.19}$$

The resulted weight of each particle is normalized as,

$$w^i_k = \frac{w^i_k}{\sum_{i=1}^{NP} w^i_k}. \tag{5.20}$$

The state vector is estimated based on the particles and associated weights via (5.3) $\sim$ (5.7).

The above initialization, prediction and update stages form a single iteration of the recursive algorithm. However, after a few iterations, degeneracy problem occurs, where all but one particle have negligible weights. It is shown in [130] that the variance of the importance weights can only increase over time, and thus it is impossible to avoid the degeneracy phenomenon. Resampling is a common method to reduce the effects of degeneracy. It eliminates particles that have smaller weights and duplicates particles with larger weights many times thus reducing the computation burden. The following resampling stage is added to reduce the degeneracy problem:

Algorithm 5.1: Resampling Algorithm

```
$[\{\hat{S}_k^j, w_k^j\}_{j=1}^{NP}] = RESAMPLING[\{S_k^i, w_k^i\}_{i=1}^{NP}]$

• Calculate $M_{eff}$, $M_{eff} = \frac{1}{\sum_{i=1}^{NP}(w_k^i)^2}$
• IF $M_{eff} < T_{degeneracy}$
    – resample the discrete distribution $\{w_k^i : i = 1, \cdots, NP\}$ $NP$ times to generate particles $\{\hat{S}_k^j : j = 1, \cdots, NP\}$, so that for any $j$, $Pr\{\hat{S}_k^j = S_k^i\} = w_k^i$.
    – All the new particles are assigned with the same weight $\frac{1}{NP}$.
• ELSE
    – $\hat{S}_k^i = S_k^i, i = 1, \cdots, NP$
• END IF
• Move to the prediction stage.
```

M_{eff} is the effective sample size, which is used to evaluate degeneracy with small M_{eff} indicating severe degeneracy. Whenever significant degeneracy is observed (when M_{eff} falls below the threshold $T_{degeneracy}$), the resampling stage is used to reduce the effects of degeneracy. $T_{degeneracy}$ is used as an indication of the occurrence of the degeneracy. Too large $T_{degeneracy}$ induces unnecessary resampling steps which will reduce the number of distinctive particles and introduce extra Monte Carlo variation. Too small $T_{degeneracy}$ will miss onset of degeneracy.

Although the resampling step reduces the effects of the degeneracy problem, it introduces other practical problems. The particles that have high weights are statistically selected many times. This leads to loss of diversity among the particles as the resultant samples will contain many repeated points which results in sample impoverishment. Sample impoverishment leads to failure in tracking since less diverse particles are used to represent the uncertain dynamics of the moving object. Especially when tracking a randomly moving object, whose position, velocity and acceleration vary quickly, sample impoverishment becomes very serious (all particles collapse to a single point within a few iterations) and then the tracking algorithm fails. An improved particle filter with a new resampling algorithm is proposed to tackle this issue. After the traditional resampling procedure, an adaptive diversing procedure is added to draw new particles from the neighborhoods of the focused particles, which enriches the diversity of particles.

5.3 Improved Resampling Algorithm

According to Bayesian theory, the prior distribution of parameters, on which there is no information, could be considered as a uniform distribution. In the diversing procedure, the new particles are assumed uniformly distributed in the neighborhoods of the previous resampled particles and are sampled from $U(\chi_k^{i:l} - \alpha \cdot \sigma_{\chi^l}, \chi_k^{i:l} + \alpha \cdot \sigma_{\chi^l})$, where σ_{χ^l} is the standard deviation of the lth state variable in the state vector. α determines the size of the sampling region and is adapted to $1/M_{eff}$ in (5.21),

$$\alpha = \frac{K}{M_{eff}}, \tag{5.21}$$

where K is a constant tuning parameter. It can be seen from (5.21) that the smaller the M_{eff} is, the larger the sampling area size will be. When the resampled particles are more focused and in the subsequent diversing procedure the sampling region is expanded to obtain more "diverse" particles. In this work, K is set to $\frac{1}{10}NP$. Clearly the choice of K is a compromise: too large a value blurs the posterior distribution and too small a value produces tight clusters of points around the original samples.

Fig. 5.2 shows sample impoverishment in the traditional resampling method, while in the improved resampling method, as shown in Fig. 5.3, the particle distribution area is expanded at each time step and sample impoverishment is eliminated.

The steps involved in the improved resampling algorithm is provided in what follows. When the effective sample size M_{eff} is less than $T_{degeneracy}$, the particles with smaller weights are eliminated and, those with larger weights are retained and duplicated as in traditional resampling. All the resulting particles are assigned with the same weight $\frac{1}{NP}$. In the subsequent diversing step, new particles are drawn from the neighborhoods of the previously resampled particles based on a uniform distribution.

The new resampling algorithm reduces sample impoverishment introduced by the traditional resampling method effectively and it obtains good results in the application of tracking a randomly moving object (Section 5.4).

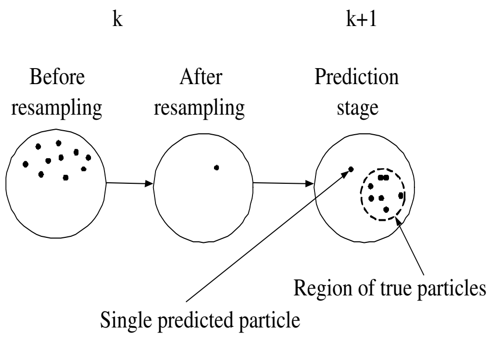

Figure 5.2: Traditional resampling method

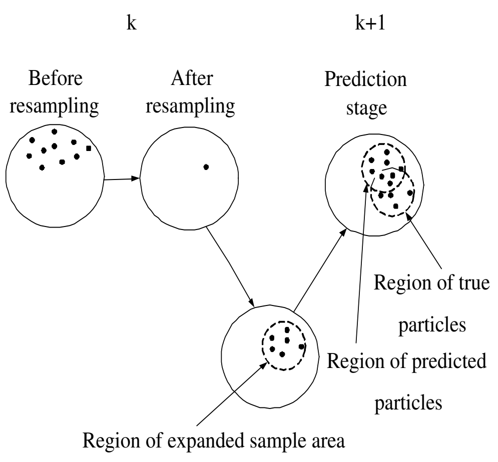

Figure 5.3: Improved resampling method

5.4 Experimental Results

5.4.1 Physical Structure of the Mobile Robot

The proposed system is implemented on a Magellan Pro robot. Magellan Pro mobile robot is part of iRobot's Research Robot indoor family, cylindrical in shape with a height of 25.4 *cm* and diameter of 40.6 *cm*. It is designed with a dense IR and sonar sensor coverage, thus offering a 360^o-view of the robot's surroundings. The robot also has an on-board PC (Pentium II) with Red Hat Linux 6.2 and Mobility software installed. In this project, a camera and video capture card are used to extend the perceptual capabilities of the robot.

The visual system consists of the Hauppage frame grabber with a BT878 chip set and the Sony EVI-D30 pan-tilt camera. The video interface is v4l, Video for Linux. Communication with the robot is done using the BreezeCom BreezeNet PRO 1.1 wireless LAN device.

5.4.2 3-D Geometry Relationship of the Mobile Robot System

Since the 16 sonar sensors constitute a 360 degree description of the robot's surroundings, it is possible to assign one sonar sensor to a specific point in the image plane. The sonar sensor corresponding to the *ith* particle-object in the image plane can then be decided. In Fig. 5.4, the camera image plane is perpendicular to the top plane of the robot. The projection of the image center R on the robot top plane coincides with R', the center of the robot top plane. The image point P_i is the center of the *ith* particle-object. Δx^i is the x component of the distance between P_i and the image plane center R. To find the relation between a two-dimensional (2-D) point in the image plane with its corresponding 3-D point in space, a perspective model is used. The perspective model consists of the image plane, the focus of projection O and the optical axis Oa_3, which goes through the image plane center R. $|OR|$ is the focal length. The 3-D point P_o corresponding to image point P_i must lie on the projection line OP_i. To simplify the derivation, the lines and points in the 3-D space are projected on to the robot top plane. Points in the robot top plane, O', P_i', R', P_o' and a_3' are respectively the projection points of the 3-D points O, P_i, R, P_o and a_3 (Fig. 5.4). Fig. 5.5 is the top view of the robot top plane with a_1 and a_2 as the vertical and horizontal axes. In the robot top plane, the angles defined in clockwise direction with respect to a_1 are positive.

In Fig. 5.5, when the camera is in the original position, the image plane projection (bold dash line) coincides with a_2 and, $O'a_3'$ coincides with a_1. For instance, consider the situation when the camera has panned by an angle A (the angle between a_1 and $O'a_3'$). The sonar sensor which faces the object with its central axis passing through the center of the object, receives strong reflection signal and reports the correct distance. On the 2-D robot top plane, the sonar with its central axis projection (the line connecting the projection of sonar sensor center and R') nearest to the object's center projection P_o', corresponds to the object. The corresponding sonar sensor is identified by comparing the angle (D') (between the line $R'P'_o$ and axis a_1) with the 16 sonar sensor angles and finding the one with the minimum angle difference. The sonar sensor angle is defined as the angle between the sonar central axis projection and axis a_1 (ie. the angle C in Fig. 5.5). Since the exact position of the 3-D point P_o is not known, its projection point P'_o is not known. Angle D' is then approximately estimated. In Fig. 5.5, $D' = D + \angle R'P'_oJ$, where D is the angle between line $O'P'_i$ and axis a_1. In the triangle $R'O'P'_i$, $|O'R'| = f$ and $|R'P'_i| = |\Delta x^i|$, which are far smaller in size compared to the distance $|R'P'_o|$, and as a result $|R'J|$ is much smaller than $|R'P'_o|$ and $|JP'_o|$. It is then reasonable to assume that $\angle R'P'_oJ \approx 0$, and $D' \approx D$. In Fig.5.5, $D = A + B$, where B is the angle between line $O'P'_i$ and axis $O'a'_3$. B can be estimated via the pan angle $\Delta\theta^i_{x\_k}$ for the ith particle-object as,

$$B = \Delta\theta^i_{x\_k} = \frac{cx^i_{obj\_k} - cx_{img}}{S_x f} = \frac{\Delta x^i_k}{S_x f}. \tag{5.22}$$

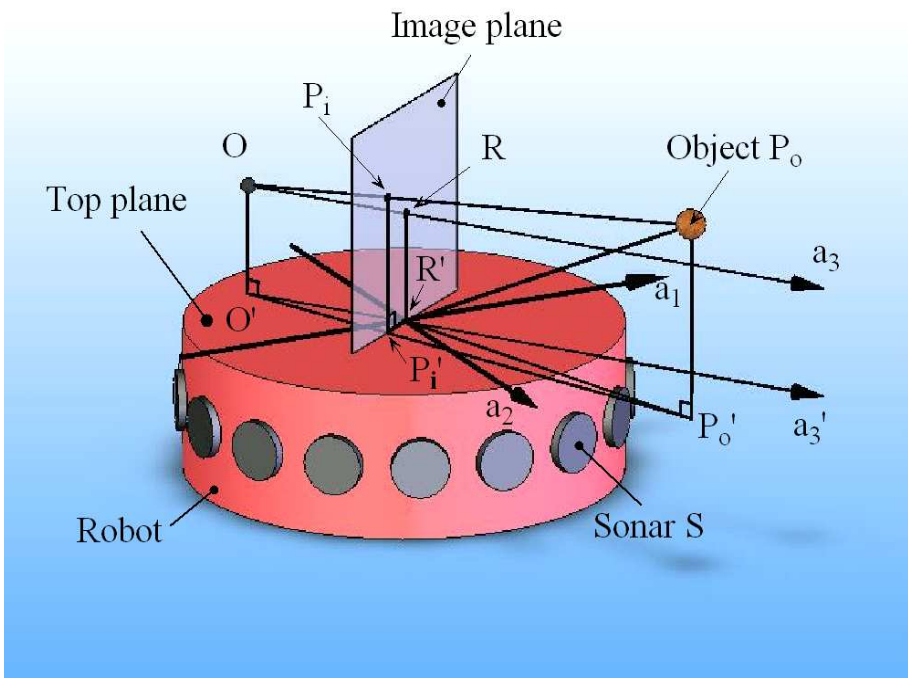

Figure 5.4: Geometry relationship in 3-D space

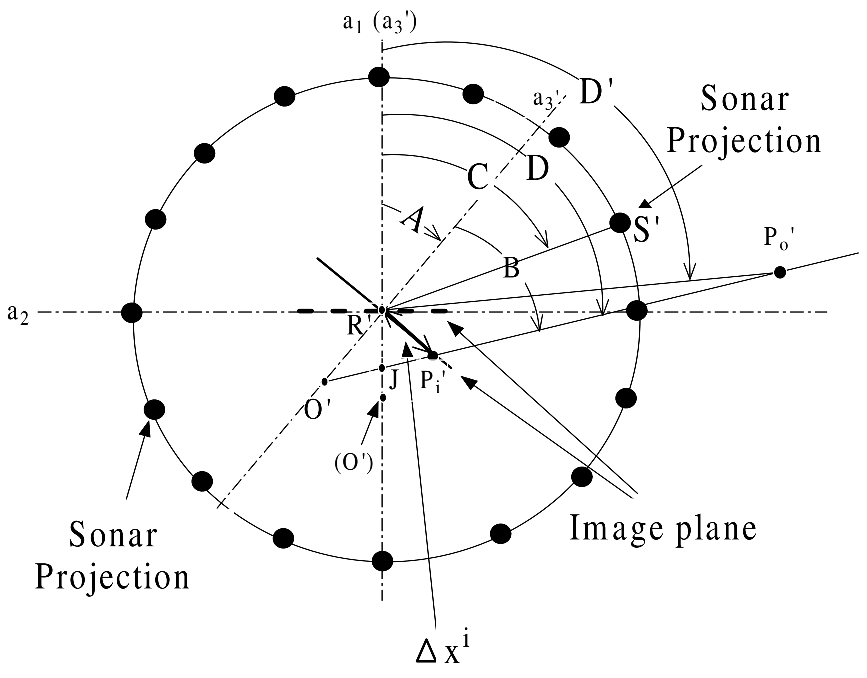

Figure 5.5: Top view of the robot

5.4.3 Logic Architecture of the Mobile Robot Tracking System

The logic architecture of the proposed system is shown in Fig. 5.6. Sequence of images taken by the pan-tilt camera are input to the moving object detection module to obtain the reference model and the initial state vector. The color cues extracted from the raw images and, the distance cues from sonar sensors and the reference model are input to the sensor fusion system. The outputs of the sensor fusion system, the estimated x and y components of the distance between the object-center and the image-center, and the estimated distance between the robot and the moving object, are passed on to the camera controller and robot trajectory controller respectively. The camera controller controls the pan-tilt camera to center the object in the image plane. The trajectory controller commands the robot to follow the moving object, maintaining the distance and orientation of the robot towards the target.

5.4.4 Experimental Results and Analysis

The experimental parameters are listed in Table 5.1.

The experiments, in which the mobile robot tracks a randomly moving person, are carried out in a laboratory environment. The experimental results are shown in Fig. 5.7 $\sim$ Fig. 5.10. The red crosses denote centers of particle-objects and the blue dot denotes the estimated center of the object. The white rectangle denotes the estimated size of the object. Fig. 5.7 shows the tracking result using the traditional resampling method. In frames 1 to 4, particles are distributed around the center of the image when the tracked objects (human legs) are in the middle of the image. When the legs move to the left in frame 5, most of the particles could not follow the object and are eliminated through traditional resampling method, which leads to dramatic decrease in particle number (frame 6). In frame 7, only three particles are

Table 5.1: Simulation parameters

Simulation Parameter	Value
Number of particles: NP	30
Diameter of the robot's top plane	40.6 cm
Maximum detecting range of sonar	3 m
Maximum panning angle (speed) and tilting angle (speed)	$\pm 100\ deg(80\ deg/s), \pm 25\ deg(50\ deg/s)$
Focus length of camera: f	5.4 mm
Width and height of image sensor	4.8 mm and 3.6 mm
Width and height of image plane	160 $pixel$ and 120 $pixel$
Scale factors: S_x and S_y	23.622 and 23.622
Frame grabber rate	25 $frame/s$
Time interval between two consecutive frames after image processing: ΔT	$\frac{1}{20}\ s$
Threshold T_{moving}	3000
Threshold T_{pixel}	0.5
Threshold $T_{\Delta h}$ and $T_{\Delta l}$	$0.1 * \tilde{h}_{k-1}$ and $0.1 * \tilde{l}_{k-1}$
Variances of the $r^{i,j}_{p\_k}$, $g^{i,j}_{p\_k}$ and $b^{i,j}_{p\_k}$ variables: σ_r^2, σ_g^2, σ_b^2	$0.01, 0.01, 0.01$
Variance of the distance variable: σ_d^2	0.1
Tuning parameter: K	$\frac{1}{10}NP$
Variance matrix of process noise: Q	$diag\{5\ 5\ 10\ 10\ 0.1\}$
Threshold $T_{degeneracy}$	3

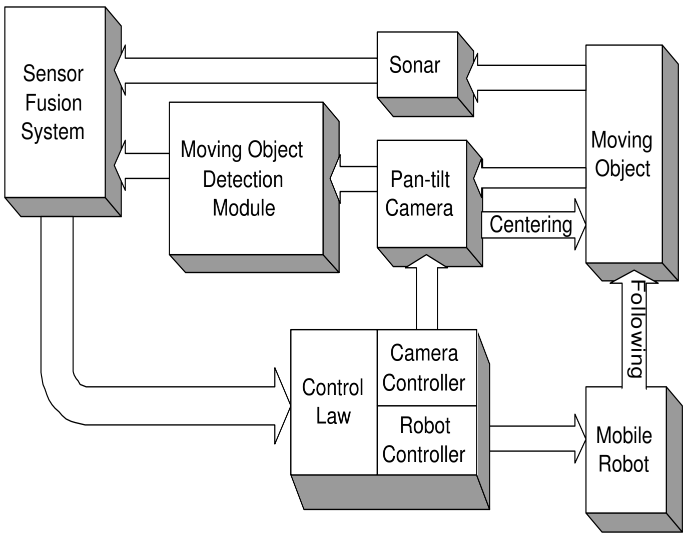

Figure 5.6: Architecture of the robot tracking system

left. Tracking process fails in frame 8 due to sample impoverishment. Fig. 5.8 shows the tracking result using the proposed resampling method. When the legs move to the left (frames 1 and 2), most of the particles could not follow the object (frame 2). The proposed resampling method utilized the adaptive diversing procedure to draw new particles from the neighborhoods of the previously focused resampled particles approximating the posterior distribution of the target state. In frame 3, the particle number increases and particles focus on the tracked object region. Similarly, when the legs move to the right quickly from frame 4 to frame 6, the particles follow the object's movement quickly and smoothly. Fig. 5.9 shows a sequence of interesting images obtained when the mobile robot follows a person who performs the following movements: crouches to pick up a box in the floor, stands up and walks towards a desk, and puts the box on the desktop. The robot watches the movements and approaches the person. In Fig. 5.9 the size of the tracked object varies largely from frame to frame, while the blue dot is always on the tracked object as the pan-tilt camera is able to lock the object successfully. In Fig. 5.10 after a full occlusion, the tracker recovers fast.

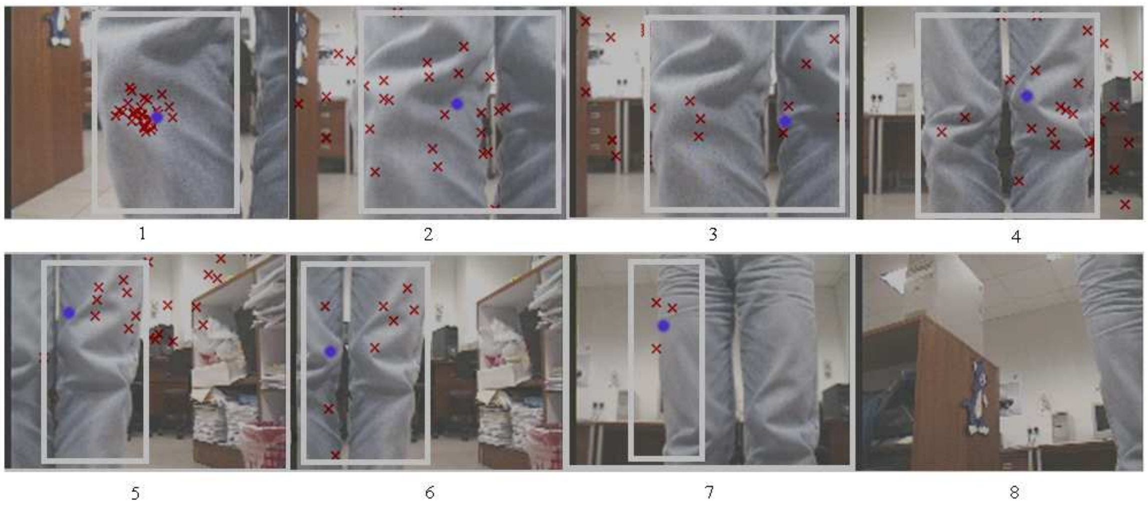

Figure 5.7: Tracking result using traditional resampling method

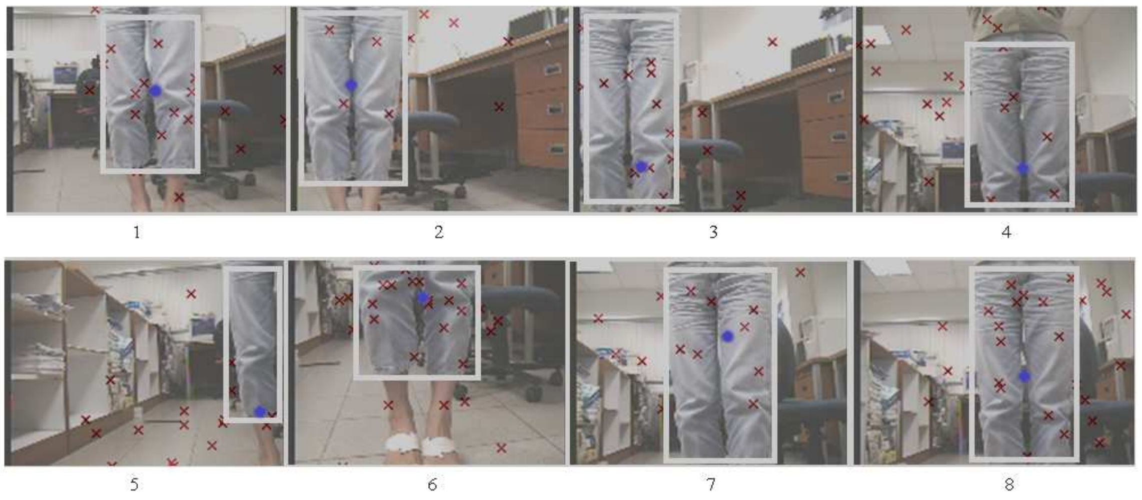

Figure 5.8: Tracking result using new resampling method

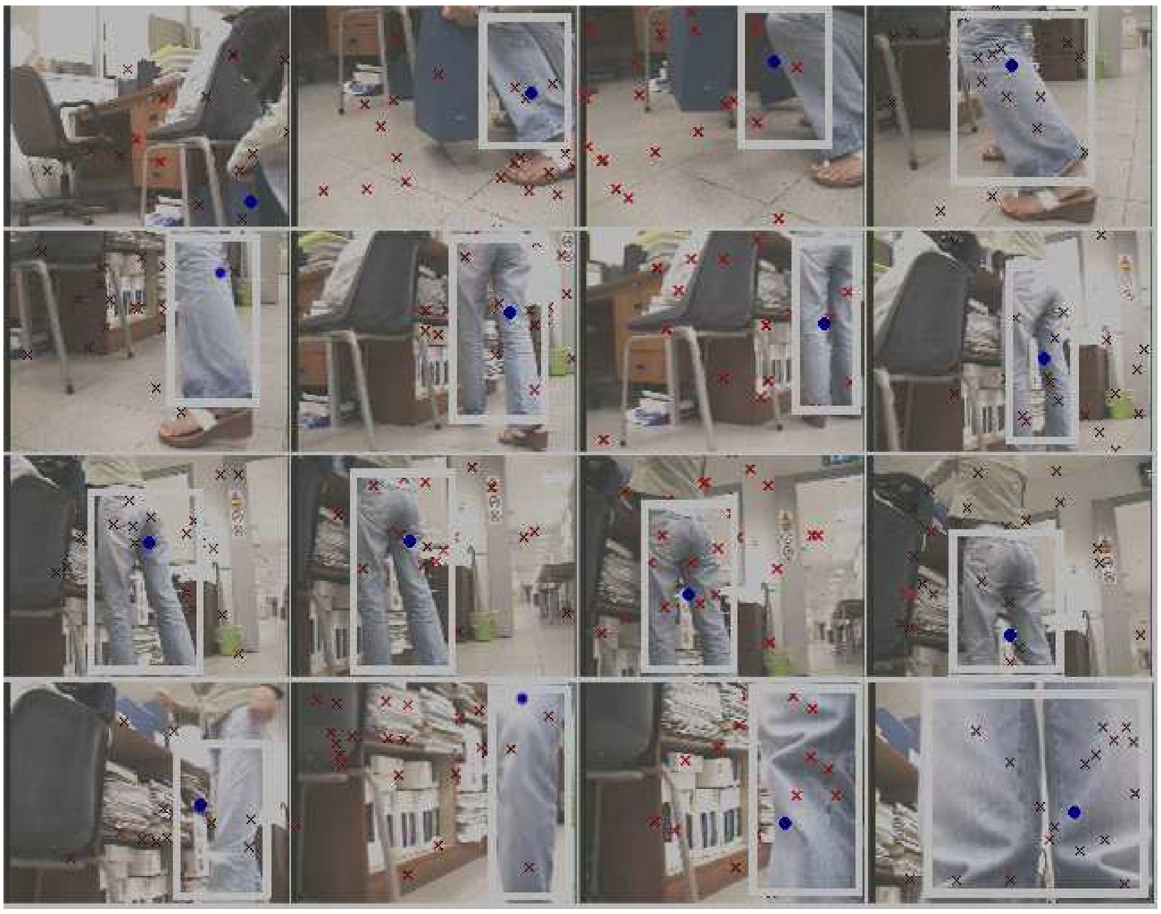
Figure 5.9: Tracking result with random movement

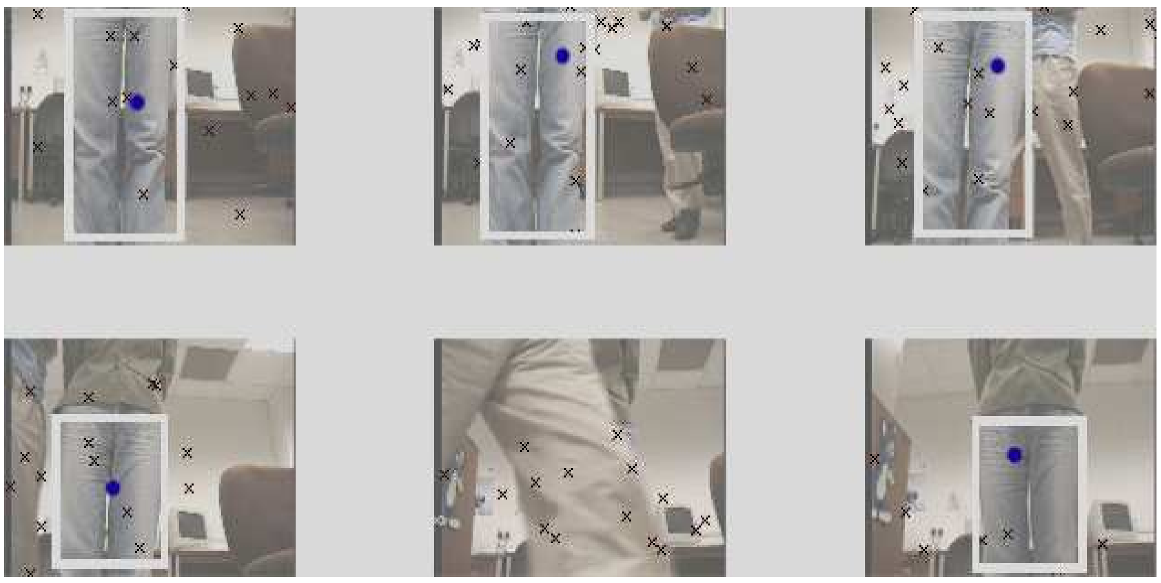

Figure 5.10: Tracking result with full occlusion

5.4.5 Upper Velocity Estimation

The above experiments are carried out when the person moves at normal walking speed. When the object moves faster, tracking fails as the object falls out of the camera's field of view. Though the camera can pan and tilt to center the object, due to the response time-limit of the camera, it is required that the moving object should be present in two consecutive image frames even if the camera is stationary. The motion of the object along a direction perpendicular to the optical axis of the camera is the fastest way an object can leave out the camera's field of view. To guarantee that the object appears in two consecutive image frames, the object's position difference along the direction perpendicular to the optical axis of the camera during two consecutive frames can not exceed the width of the camera's field of view.

According to the general optics theory,

$$\frac{H_i}{H_o} = \frac{D_i}{D_o}, \tag{5.23}$$

where H_i and H_o respectively represent the sizes of the object in the image frame and the real object. D_i denotes the distance from the image plane to the rear principle plane of the lens. D_o denotes the distance from the object plane to the front principle plane of the lens. When H_i is chosen as the width of the image plane W_i, the corresponding H_o will be the width of the field of view, W_{FOV}, given some specified distance D_o.

$$W_{FOV} = \frac{D_o}{D_i} W_i \tag{5.24}$$

The upper velocity limit V_{upper} of a moving object that can be tracked is obtained through (5.25),

$$V_{upper} = \frac{W_{FOV}}{\Delta T} = \frac{D_o W_i}{D_i \Delta T} = \frac{0.5\ m \times 4.8\ mm}{3\ cm \times \frac{1}{20}\ s^{-1}} = 1.6\ m/s, \tag{5.25}$$

where ΔT is the time interval between two consecutive frames after the image processing. In the experiment, D_o is the distance kept between the robot and the moving object and is chosen approximately as 0.5 m. Since the exact value of D_i is not known, it is approximated to 3 cm. To increase the upper velocity limit of the moving object, improvements in hardware, by increasing the computing power of the onboard computer and by reducing the response time of the pan-tilt camera can be considered.

5.5 Conclusions

In this chapter, a real-time algorithm to track a randomly moving object based on information received from multiple sensors is proposed in the particle filter framework. A new resampling algorithm is proposed to tackle sample impoverishment. After the traditional resampling procedure, an adaptive diversing procedure is added to draw new particles from the neighborhoods of the focused particles, which enriches the diversity of particles. The particle filter with the new resampling algorithm is able to track a randomly moving object. A mobile robot real-time tracking system, composed of vision, sensor fusion and control subsystems, is used to verify the effectiveness of the proposed algorithm. The experimental results show the capabilities of the mobile robot to continuously and smoothly follow a randomly moving object (a human subject) at a reasonable walking rate.

Chapter 6

Summary and Proposals

This chapter consists of two major portions, namely, summary of the works in Section 6.1 and some proposals for future works in Section 6.2.

6.1 Summary of the Works

In this thesis, effective particle filter based methods have been developed for target tracking applications, which include single maneuvering target tracking, multiple target tracking and multiple maneuvering target tracking. Furthermore, a real-time target tracking system based on multi-sensor fusion is implemented on a mobile platform, the Magellan robot.

Firstly, two different methods, MCMC based particle filter and process noise estimation based particle filter, are proposed to tackle the maneuvering target tracking problem.

In the MCMC based particle filter method, the maneuvering movements are tracked through moving the particles towards the target posterior distribution via Markov chain Monte Carlo (MCMC) sampling. Two new sampling methods are adopted to speed up the MCMC convergence rate: the adaptive MCMC sampling and the interacting MCMC sampling. The proposed adaptive MCMC based particle filter method, which is the combination of the adaptive Metropolis (AM) method and the importance sampling method, tackles the real-time tracking problem effectively. As for the interacting MCMC based particle filter method, the importance sampling is replaced with interacting MCMC sampling, which avoids the sample impoverishment while accelerating the MCMC convergence rate.

The second method deals with the maneuvering target tracking problem using the equivalent-noise method, in which the maneuvering effect is modeled by (part of) a white or colored noise process. The proposed method focuses on the identification of the equivalent process noise: the process noise is modeled as a dynamic system and a sampling based algorithm is proposed in the particle filter framework to deal with process noise identification problem.

Secondly, two different algorithms are presented for multiple target tracking problem. The first algorithm is named the particle filter based multi-scan joint probabilistic data association (MS-JPDA) filter and is an extension of the single scan JPDA methods. In contrast to the single scan JPDA methods, the proposed multi-scan JPDA method examines the joint association hypothesis in a multi-scan sliding window and calculates the posterior marginal probability based on the multi-scan joint association hypothesis. The multi-scan JPDA method uses more scans of measurements with more information, which results in better computed probabilities.

The second algorithm, named multi-scan mixture particle filter, applies the particle filter algorithm directly in the multiple target tracking process and avoids the data association problem. The posterior distribution of target state is a multi-mode distribution and each mode corresponds to either a target or clutter. In order to distinguish the targets from the clutters, multi-scan information is incorporated. Moreover, to deal with the new target appearance problem, a set of new particles are sampled from the likelihood model (according to the most recent measurements) to detect the new modes appeared at each time step.

Thirdly, a new algorithm is proposed to cope with the multiple maneuvering target tracking problem, which is the combination of the process noise identification method for modeling highly maneuvering target, and the multi-scan JPDA algorithm for solving data association problem, in a particle filter framework. The process noise identification process is effective in estimating both the maneuvering movement and the random acceleration of the target, avoiding the use of complicated multiple model approaches. The multi-scan JPDA is effective in maintaining the tracks of multiple targets using multiple scan information.

Finally, a target tracking system based on multi-sensor fusion is implemented on a mobile robot. Experiments are carried out to verify the proposed adaptive resampling algorithm. The experimental results show that the robot is capable of continuously tracking a human's random movement at walking rate.

6.2 Further Research

Implementation of multiple object tracking system based on multi-sensor fusion using mobile robot is a natural extension of the work. Some research topics are proposed for the implementation of the multiple object tracking system:

1. Data association method. The key problem in multiple target tracking is the data association problem. The particle filter based multi-scan JPDA algorithm (proposed in Section 3.1) can be adopted to solve the data association problem, in which each of the tracking targets is assigned with a corresponding particle filter. The distribution of interest is the marginal filtering distribution for each of the targets, which is approximated with particles. Compared with the single scan JPDA methods, the multi-scan JPDA method uses richer information, which results in better estimated probabilities. However, for the multi-scan JPDA filter, examining the joint association hypothesis in a multi-scan sliding window and calculating the posterior marginal probability based on the multi-scan joint association hypothesis are very complex and time-consuming work.

A hypothesis reduction method should be utilized to speed up the algorithm to keep up with the real-time tracking process.

2. State estimation method. The proposed particle filter method with the new adaptive resampling algorithm (in Section 5.3) is considered to deal with the problem of tracking multiple objects' random movements.

3. Design of the control system. The hierarchical control system can be adopted to establish tracking process using different degrees of freedom on the vision system and the movement of the mobile robot. Since the inertia of the mobile robot is greater than the camera inertia, this type of control simulates a control system with different levels. The inmost level comprises pan-tilt camera, responsible for tracking the target in real-time. This sub-system controls the camera's position to maintain the visual system fixed on the target. At the outmost level is the mobile robot sub-system that provides the compensation for the orientation of the vision system and also controls the orientation and distance to the target.

4. Integrated system implementation. The integrated system is based on the co-operation between different control systems and sensor systems. It deals with the interaction of different control systems using visual feedback and it can be accomplished by the implementation of a visual gaze holding process interacting cooperatively with the control of the trajectory of a mobile robot. These two systems are integrated to follow a moving object at constant distance and orientation with respect to the mobile robot. The orientation and the position of the vision system running a gaze holding process give the feedback signals to the control system that tracks the target in real-time. The visual fixation, visual smooth pursuit, navigation using visual feedback and compensation for system's movements can be concerned in the future work.

Bibliography

[1] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* NJ, Prentice-Hall, 1979.

[2] N.J.Gordon and D.J.Smith, "Improved approach to nonlinear/non-Guassian Bayesian state estimation," *IEE Proceedings-F*, vol. 140, pp. 107–113, 1993.

[3] G. Kitagawa and W. Gersch, *Smoothness Priors Analysis of Time Series.* New York: Springer-Verlag, 1996.

[4] A.Doucet, "On sequential Monte Carlo methods for Bayesian filtering," *Statistics and Computing,*, vol. 10, pp. 197–208, 2000.

[5] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, *Sequential Monte Carlo Methods in Practice.* New York: Springer-Verlag, 2001.

[6] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, 1998.

[7] M. Isard and A. Blake, "CONDENSATION conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998.

[8] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear system.," *In Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defence Sensing, Simulation and Controls*, vol. Multi Sensor Fusion, Tracking and Resource Management II, 1997.

[9] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, pp. 401–422, 2004.

[10] D. L. Alspach and H. W. Sorenson, "Nonlinear Bayesian estimation using Gaussian sum approximation," *IEEE Transactions on Automatic Control,*, vol. 14, pp. 439–448, 1972.

[11] G. Kitagawa, "Non-Gaussian state-space modeling of nonstationary time series (with discussion)," *Journal of the American Statistical Association,*, vol. 82, pp. 1032–1063, 1987.

[12] J. M. Hammersley and K. W. Morton, "Poor man's Monte Carlo," *J. Roy.Stat. Soc., Series B*, vol. 16, no. 1, pp. 23–38, 1954.

[13] M. N. Rosenbluth and A. W. Rosenbluth, "Monte Carlo calculation of the average extension of molecular chains," *J. Chem. Phys.*, vol. 23, no. 2, pp. 356–359, 1956.

[14] D. Crisan, P. D. Moral, and T. J. Lyons, "Non-linear filtering using branching and interacting particle systems," *Markov Processes Related Fields*, vol. 5, pp. 293–319, 1999.

[15] P. D. Moral, "Non-linear filtering: Interacting particle solution," *Markov Processes Related Fields*, vol. 2, pp. 555–580.

[16] D. K. K. Kanazawa and S. J. Russell, "Stochastic simulation algorithms for dynamic probabilistic networks," *in Proc. Eleventh Annu. Conf. Uncertainty AI*, pp. 346–351, 1995.

[17] J. S. Liu, *Monte Carlo Strategies in Scientific Computing.* New York: Springer Press, 2001.

[18] N. Bergman, L. Ljung, and F. Gustafsson, "Terrain navigation using Bayesian statistics," *IEEE Control Systems Magazine*, vol. 19, pp. 33–40, 1999.

[19] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Transactions on Signal Processing*, vol. 49, pp. 613–624, 2001.

[20] P. D. Moral, "Measure valued processes and interacting particle systems: Application to nonlinear filtering problems," *Ann. Appl. Probab.*, vol. 8, pp. 438–495, 1998.

[21] R. van der Merwe, A. Doucet, J. F. G. de Freitas, and E. Wan, "The unscented particle filter," *Adv. Neural Inform. Process. Syst.*, 2000.

[22] N. Oudjane and C. Musso, "Progressive correction for regularized particle filters," *Proc. 3rd Int. Conf. Inform. Fusion*, 2000.

[23] J. MacCormick and A. Blake, "A probabilistic exclusion principle for tracking multiple objects," *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 1, pp. 572–578, 1999.

[24] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Proc.-Radar,Sonar Navig.*, vol. 146, 1999.

[25] B. Ripley, *Stochastic Simulation.* New York: Wiley, 1987.

[26] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models.," *Journal of Computational and Graphical Statistics,*, vol. 5, pp. 1–25, 1996.

[27] A. D. S. Godsill and M. West, "Methodology for Monte Carlo smoothing with application to time-varying autoregressions," *in Proc. Int. Symp. Frontiers Time Series Modeling*, 2000.

[28] B. P. Carlin, N. G. Polson, and D. S. Stoffer, "A Monte Carlo approach to nonnormal and nonlinear state-space modeling," *J. Amer. Statist. Assoc.*, vol. 87, pp. 493–500, 1992.

[29] W. R. Gilks and C. Berzuini, “Following a moving target - Monte Carlo inference for dynamic Bayesian models,” *Journal of Royal Statistical Society*, vol. 63, pp. 127–146, 2001.

[30] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, 2002.

[31] M. K. Pitt and N. Shephard, “Filtering via simulation: Auxiliary particle filter,” *Journal of the American Statistical Association*, vol. 94, pp. 590–599, 1999.

[32] V. Aidala and J. Davis, “The utilization of data measurement residuals for adaptive Kalman filtering,” *OCEANS*, vol. 5, pp. 450–460, 1973.

[33] R. Kirlin and A. Moghaddamjoo, “Robust adaptive Kalman filtering for systems with unknown step inputs and non-Gaussian measurement errors,” *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP*, vol. 10, pp. 157–160, 1985.

[34] M. Efe and D. Atherton, “Maneuvering target tracking with an adaptive Kalman filter,” *Proceedings of the 37th IEEE Conference on Decision and Control*, vol. 1, pp. 737–742, 1998.

[35] H. Lee and M.-J. Tahk, “Generalized input-estimation technique for tracking maneuvering targets,” *IEEE Trans. Aerospace and Electronic Systems*, vol. 35, pp. 1388–1402, Oct. 1999.

[36] Y. T. Chan and F. Couture, “Manoeuvre detection and track correction by input estimation,” *IEE Proceedings of Radar and Signal Processing*, vol. 140, pp. 21–28, Feb. 1993.

[37] Y. H. Park, J. H. Seo, and J. G. Lee, “Tracking using the variable-dimension filter with input estimation,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 31, pp. 399–408, Jan. 1995.

[38] A. T. Alouani, P. Xia, T. R. Rice, and W. D. Blair, “A two-stage Kalman estimator for tracking maneuvering targets,” in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, pp. 761–766, Oct. 1991.

[39] J. Tugnait, “Detection and estimation for abruptly changing systems,” *Automatica*, vol. 18, no. 5, pp. 607–615, 1982.

[40] H. A. P. Blom and Y. Bar-Shalom, “The interacting multiple model algorithm for systems with a jump-linear smoothing application,” *IEEE Transaction on Automatic Control*, vol. AC-33, no. 8, pp. 780–783, 1988.

[41] Y. Bar-Shalom, K. Chan, and H. Blom, “Tracking a maneuvering target using input estimation versus the interacting multiple model algorithm,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 25, no. 2, pp. 296–300, 1989.

[42] A. Munir and P. Artherton, “Adaptive interacting multiple model algorithm for tracking a maneuvering target,” *IEE Proc., Radar Sonar Navig.*, vol. 142, no. 1, pp. 11–17, 1995.

[43] R. Karlsson and F. Gustafsson, “Range estimation using angle-only target tracking with particle filters,” *Proceedings of American Control Conference*, vol. 5, pp. 3743–3748, 2001.

[44] N. Ikoma, N. Ichimura, T. Higuchi, and H. Maeda, “Maneuvering target tracking by using particle filter,” *IFSA World Congress and 20th NAFIPS International Conference*, vol. 4, pp. 2223–2228, 2001.

[45] R. Karlsson and N. Bergman, "Auxiliary particle filters for tracking a maneuvering target," *Proceedings of the 39th IEEE Conference on Decision and Control*, vol. 4, pp. 3891–3895, 2000.

[46] N. Ikoma, T. Higuchi, and H. Maeda, "Tracking of maneuvering target by using switching structure and heavy-tailed distribution with particle filter method," *Proceedings of the 2002 International Conference on Control Applications*, vol. 2, pp. 1282–1287, 2002.

[47] W. Malcolm, A. Doucet, and S. Zollo, "Sequential Monte Carlo tracking schemes for maneuvering targets with passive ranging," *Proceedings of the Fifth International Conference on Information Fusion*, vol. 1, pp. 482–488, 2002.

[48] M. Morelande and S. Challa, "Manoeuvring target tracking in clutter using particle filters," *IEEE Trans. Aerospace and Electronic Systems*, vol. 41, pp. 252–270, Jan. 2005.

[49] S. J. Godsill, J. Vermaak, W. Ng, and J. F. Li, "Models and algorithms for tracking of maneuvering objects using variable rate particle filters," *Proceedings of the IEEE*, vol. 95, pp. 925–952, May 2007.

[50] F. Opitz, "Fusion of active phased array radars within a quick reaction time using multidimensional data association," *Proceedings of the Sixth International Conference of Information Fusion*, vol. 1, pp. 342–349, 2003.

[51] T. Kirubarajan, H. Wang, Y. Bar-Shalom, and K. R. Pattipati, "Efficient multisensor fusion using multidimensional data association," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, pp. 386–400, 2001.

[52] D. Read, "An algorithm for tracking multiple targets," *IEEE Transactions on Automation and Control*, vol. 24, pp. 84–90, 1979.

[53] Y. Bar-Shalom and T. Fortmann, *Tracking and data association.* Orlando, FL: Academic Press, 1988.

[54] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. 8, pp. 173–184, 1983.

[55] L. Y. Pao, "Multisensor multitarget mixture reduction algorithms for target tracking," *AIAA Journal of Guidance, Control and Dynamics*, vol. 17, pp. 1205–1211, 1994.

[56] D. J. Salmond, "Mixture reduction algorithms for target tracking in clutter," *In O. E. Drummond (Ed.), Signal and Data Processing of Small Targets, SPIE*, vol. 1305, pp. 434–445, 1990.

[57] H. Gauvrit, J.-P. Le Cadre, and C. Jauffret, "A formulation of multitarget tracking as an incomplete data problem," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, pp. 1242–1257, 1997.

[58] R. L. Streit and T. E. Luginbuhl, "Maximum likelihood method for probabilistic multi-hypothesis tracking," *In O. E. Drummond (Ed.), Signal and Data Processing of Small Targets, SPIE*, vol. 2235, 1994.

[59] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, pp. 1–38, 1977.

[60] P. Willett, Y. Ruan, and R. Streit, "PMHT: Problems and some solutions," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, pp. 738–754, 2002.

[61] D. Avitzour, "Stochastic simulation Bayesian approach to multitarget approach," *Proc. Inst. Elect. Eng. - Radar, Sonar, Navig.*, vol. 142, pp. 41–44, 1995.

[62] J. M. Cormick and M. Isard, "Bramble: A Bayesian multiple-blob tracker," *Proceedings of the Eighth International Conference on Computer Vision*, vol. 2, pp. 34–41, 2001.

[63] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, "Tracking multiple moving targets with a mobile robot using particle filters and statistical data association," *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1665–1670, 2001.

[64] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filter for simultaneous mobile robot localization and people-tracking," *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 695–701, 2002.

[65] C. Hue, J.-P. Le Cadre, and P. Perez, "Tracking multiple objects with particle filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, pp. 791–812, 2002.

[66] M. Orton and W. Fitzgerald, "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Transactions on Signal Processing*, vol. 50, pp. 216–223, 2002.

[67] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of the image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[68] F. Dellaert, S. M. Seitz, C. Thorpe, and S. Thrun, "EM, MCMC, and chain flipping for structure from motion with unknown correspondence," *Machine Learning*, vol. 50, pp. 45–71, 2003.

[69] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov Chain Monte Carlo in Practice*. London: Chapman and Hall, 1996.

[70] N. J. Gordon and A. Doucet, "Sequential Monte Carlo for manoeuvering target tracking in clutter," *In O. E. Drummond (Ed.), Signal and Data Processing of Small Targets, SPIE*, vol. 3809, pp. 493–500, 1999.

[71] A. Doucet, B.-N. Vo, C. Andrieu, and M. Davy, "Particle filtering for multi-target tracking and sensor management," *Proceedings of the Fifth International Conference on Information Fusion*, vol. 1, pp. 474–481, 2002.

[72] N. Ikoma and S. J. Godsill, "Extended object tracking with unknown association, missing observations, and clutter using particle filters," *In Proceedings of the 12th IEEE Workshop on Statistical Signal Processing*, pp. 485–488, 2003.

[73] D. Schulz, W. Burgard, and D. Fox, "People tracking with mobile robots using sample-based joint probabilistic data association filters," *International Journal of Robotics Research*, vol. 22, 2003.

[74] O. Frank, J. Nieto, J. Guivant, and S. Scheding, "Multiple target tracking using sequential Monte Carlo methods and statistical data association," *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[75] Y. Boers, "A multi-target track before detect application," *In Proceedings of the IEEE Workshop on Multi-Object Tracking*, 2003.

[76] D. J. Salmond and H. Birch, "A particle filter for track-before-detect," *In Proceedings of the American Control Conference*, pp. 3755–3760, 2001.

[77] J. Vermaak, A. Doucet, and P. Perez, "Maintaining multi-modality through mixture tracking," *Proc. IEEE. ICCV*, 2003.

[78] J. Vermaak, S. J. Godsill, and P. P. Erez, "Monte Carlo filtering for multi-target tracking and data association," *IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS*, vol. 41, pp. 309–332, 2005.

[79] C. Musso, N. Oudjane, and F. LeGland, *Improving regularised particle filters in Sequential Monte Carlo Methods in Practice.* New York: Springer-Verlag, 2001.

[80] H. Haario, E. Saksman, and J. Tamminen, "An adaptive Metropolis algorithm," *Bernoulli*, vol. 7, pp. 223–242, 2001.

[81] Z. Khan, T. Balch, and F. Dellaert, "MCMC-based particle filtering for tracking a variable number of interacting targets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1805–1819, 2005.

[82] W. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[83] A. Gelman, G. O. Roberts, and W. R. Gilks, "Efficient Metropolis jumping rules," in *Bayesian Statistics 5* (J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, eds.), Oxford University Press, 1996.

[84] P. Angeline, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences," in *Evolutionary Programming VII* (V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, eds.), pp. 601–610, Springer-Verlag, 1998.

[85] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *IEEE Int. Conf. Neural Networks*, (Perth, Australia), pp. 1942–1948, Nov. 1995.

[86] J. Kennedy, "The particle swarm: Social adaptation of knowledge," in *Int. Conf. Evolutionary Computation*, (Indianapolis, IN), pp. 303–308, Apr. 1997.

[87] M. Clerc and J. Kennedy, “The particle swarmexplosion, stability, and convergence in a multidimensional complex space,” *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, vol. 6, pp. 58–73, 2002.

[88] S. W. Yeom, T. Kirubarajan, and Y. Bar-Shalom, “Track segment association, fine-step IMM and initialization with Doppler for improved track performance,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 293 – 309, 2004.

[89] J. Tugnait, “Tracking of multiple maneuvering targets in clutter using multiple sensors, IMM, and JPDA coupled filtering,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 320–330, 2004.

[90] M. Farmer, R.-L. Hsu, and A. Jain, “Interacting multiple model (IMM) Kalman filters for robust high speed human motion tracking,” *16th International Conference on Pattern Recognition*, vol. 2, pp. 20–23, 2002.

[91] L. Johnston and V. Krishnamurthy, “An improvement to the interacting multiple model (IMM) algorithm,” *IEEE Transactions on Signal Processing*, vol. 49, pp. 2909–2923, 2002.

[92] B. W. Ahn, J. W. Choi, T. H. Fang, and T. L. Song, “A modified variable dimension filter with input estimation for maneuvering target tracking,” *American Control Conference*, vol. 2, pp. 1266–1271, 2003.

[93] J. Cloutier, C.-F. Lin, and C. Yang, “Enhanced variable dimension filter for maneuvering target tracking,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 29, pp. 786–797, 1993.

[94] F.-B. Duh and C.-T. Lin, “Tracking a maneuvering target using neural fuzzy network,” *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 34, pp. 16–33, 2004.

[95] A. H. Jazwinski, *Stochastic processes and filtering theory.* New York: Academic Press, 1970.

[96] X. R. Li and Y. Bar-Shalom, "A recursive multiple model approach to noise identification," *IEEE Trans. Aerospace and Electronic Systems*, vol. 30, pp. 671–684, July 1994.

[97] X. R. Li and Y. Bar-Shalom, "A recursive hybrid system approach to noise identification," *Proceedings of the first IEEE Conference on Control Applications*, pp. 847–852, 1992.

[98] P.-O. Gutman and M. Velger, "Tracking targets using adaptive Kalman filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 26, pp. 691–699, 1990.

[99] P. Weston and J. Norton, "Process-noise models for particle filtering under bounded forcing with unknown distribution," *American Control Conference*, vol. 5, pp. 3749–3754, 2001.

[100] P. Djuric and J. Miguez, "Sequential particle filtering in the presence of additive Gaussian noise with unknown parameters," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1621–1624, 2002.

[101] R. Singh and B. Raj, "Tracking noise via dynamical systems with a continuum of states," *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. I–396–I–399, 2003.

[102] J. H. Kotecha and P. M. Djuric, "Gaussian particle filtering," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2592–2601, 2003.

[103] J. H. Kotecha and P. M. Djuric, "Gaussian sum particle filtering," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2602–2612, 2003.

[104] B.Chen and T. J.K., "Tracking of multiple maneuvering targets in clutter using IMM/JPDA filtering and fixed-lag smoothing," *Automatica*, vol. 37, pp. 239–249, Feb. 2001.

[105] C. Chang and R. Ansari, "Kernel particle filter for visual tracking," *IEEE Signal Processing Letters*, vol. 12, no. 3, pp. 242–245, 2005.

[106] J. L. Bentley, "Multidimensional divide and conquer," *Communications of the ACM*, vol. 23(4), 1980.

[107] A. W. Moore, *Efficient Memory-based Learning for Robot Control.* PhD thesis, Cambridge Univ., 1990.

[108] X. R. Li, Y. Bar-Shalom, and T. Kirubarajan, *Estimation, Tracking and Navigation: Theory, Algorithms and Software.* New York: John Wiley & Sons, June 2001.

[109] Y. Bar-Shalom and W. D. Blair, *Multitarget-Multisensor Tracking: Applications and Advances*, vol. III. Norwood, MS: Archtech House, 2000.

[110] Y. Bar-Shalom and X. R. Li, *Estimation and Tracking: Principles, Techniques and Software.* Norwood, MA: Archtech House, 1993.

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

[112] A. Houles and Y. Bar-Shalom, "Multisensor tracking of a maneuvering target in clutter," *IEEE Trans. Aerospace and Electronic Systems*, vol. 25, pp. 176–188, Mar. 1989.

[113] H. Blom and E. Bloem, "Interacting multiple model joint probabilistic data association avoiding track coalescence," in *Proceedings. 41st IEEE Conference on Decision and Control*, vol. 3, pp. 3408–3415, Dec. 2002.

[114] Y. Bar-Shalom, K. Chang, and H. Blom, "Tracking splitting targets in clutter by using an interacting multiple model joint probabilistic data association filter," in *Multitarget multisensor tracking: applications and advances* (Y. Bar-Shalom, ed.), vol. II, pp. 93–110, Artech House, 1992.

[115] S. Puranik and J. K. Tugnait, "Tracking of multiple maneuvering targets using multiscan JPDA and IMM filtering," in *Proceedings of the 2004 American Control Conference*, vol. 5, pp. 4096–4101, July 2004.

[116] H. A. P. Blom and E. A. Bloem, "Combining IMM and JPDA for tracking multiple maneuvering targets in clutter," in *Proceedings of the Fifth International Conference on Information Fusion*, vol. 1, pp. 705–712, July 2002.

[117] B. Vo and W.-K. Ma, "Joint detection and tracking of multiple maneuvering targets in clutter using random finite sets," in *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference*, vol. 2, pp. 1485–1490, Dec. 2004.

[118] J. H. Wang, W. T. Liu, M. Wang, B. Zhang, and J. Wang, "Multiple maneuvering target data association based on genetic algorithms," in *The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005.*, vol. 2, pp. 1011–1014, Feb. 2005.

[119] D. Musicki and R. Evans, "Target existence based MHT," in *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05.*, pp. 1228–1233, Dec. 2005.

[120] A. Marrs, "Asynchronous multi-sensor tracking in clutter with uncertain sensor locations using Bayesian sequential Monte Carlo methods," *Aerospace Conference, IEEE Proceedings.*, vol. 5, pp. 2171–2178, 2001.

[121] M. Hernandez, "Efficient data fusion for multi-sensor management," *Aerospace Conference, IEEE Proceedings.*, vol. 5, pp. 2161–2169, 2001.

[122] S. Challa, M. Palaniswami, and A. Shilton, "Distributed data fusion using support vector machines," *Proceedings of the Fifth International Conference on Information Fusion*, vol. 2, pp. 881–885, 2002.

[123] G. Loy, L. Fletcher, N. Apostoloff, and A. Zelinsky, "An adaptive fusion architecture for target tracking," *Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 248–253, May 2002.

[124] A. Doucet and N. Gordon, "Efficient particle filters for tracking manoeuvring targets in clutter," *IEE Colloquium on Target Tracking: Algorithms and Applications*, pp. 4/1–4/5, 1999.

[125] C. Hue, L. C. J.-P., and P. Perez, "Sequential Monte Carlo methods for multiple target tracking and data fusion," *IEEE Transactions on Signal Processing*, vol. 50, pp. 309–325, 2002.

[126] B. Kwolek, "Person following and mobile camera localization using particle filters," *Fourth International Workshop on Robot Motion and Control*, pp. 265–270, 2004.

[127] J. Dias, C. Paredes, I. Fonseca, H. Araujo, J. Batista, and A. T. Almeida, "Simulating pursuit with machine experiments with robots and artificial vision," *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 1–18, 1998.

[128] P. Prez, J. Vermaak, and A. Blake, "Data fusion for visual tracking with particles," *Proceedings of the IEEE*, vol. 92, pp. 495–513, 2004.

[129] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis.* New Jersey: Prentice Hall, 2002.

[130] A. Doucet, "On sequential Monte Carlo methods for Bayesian filtering," *Tech. Rep.,Dept., Eng., Univ. Cambridge, UK*, 1998.

Author's Publications

Published Papers:

[1] Vadakkepat, P. and Jing, L, "Improved Particle Filter in Sensor Fusion for Tracking Randomly Moving Object," *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 5, pp. 1823-1832, October 2006.

[2] Liu Jing and Prahlad Vadakkepat, "Improved Particle Filter in Sensor Fusion for Tracking Random Moving Object," *Proc. of the 21st IEEE Instrumentation and Measurement Technology Conference*, pp. 476-481, Italy, May 2004.

[3] Liu Jing and Prahlad Vadakkepat, "Multiple Targets Tracking by Optimized Particle Filter Based on Multi-scan JPDA," *Proc. of the 21st IEEE Instrumentation and Measurement Technology Conference*, pp. 303-308, Italy, May 2004.

[4] Liu Jing and Prahlad Vadakkepat, "Adaptive Particle Filter in Sensor Fusion for Tracking Moving Object with Uncertain Dynamics," *The 5th International Conference on Simulated Evolution And Learning (SEAL04)*, Korea, October 2004.

Submitted Papers:

[1] Liu Jing and Prahlad Vadakkepat, "Maneuvering Target Tracking Based on Process Noise Identification Using Particle Filter," *IEE Proceedings-Vision, Image, and Signal Processing*, Aug., 2006.

[2] Liu Jing and Prahlad Vadakkepat, "Multiple Maneuvering Target Tracking By Improved Particle Filter Based on Multi-scan JPDA," *Automatica*, Sept., 2006.

[3] Prahlad Vadakkepat, Peter Lim, Liyanage C. De Silva, Li Li Ling and Liu Jing, "Multi-Modal Approach to Human Face Detection and Tracking," *IEEE Transactions on Industrial Electronics*, Sept., 2006.

[4] Liu Jing and Prahlad Vadakkepat, "Interacting MCMC Particle Filter for Tracking Maneuvering Target," *Digital Signal Processing*, Nov., 2006.

[5] Liu Jing and Prahlad Vadakkepat, "Adaptive MCMC Based Particle Filter for Tracking Random Moving Object via Sensor Fusion," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, Nov., 2006.