MULTI-TARGET PARTICLE FILTER BASED TRACK BEFORE DETECT ALGORITHMS FOR SPAWNING TARGETS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MEHMET EYILI

IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JUNE 2014

Approval of the thesis:

MULTI-TARGET PARTICLE FILTER BASED TRACK BEFORE DETECT ALGORITHMS FOR SPAWNING TARGETS

submitted by **MEHMET EYİLİ** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Mübeccel Demirekler
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Mustafa Kuzuoğlu
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Mübeccel Demirekler
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Seyit Sencer Koç
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. Umut Orguner
Electrical and Electronics Engineering Dept., METU _____

Elif Yavuztürk, M.Sc. in EEE
REHIS, ASELSAN _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : MEHMET EYİLİ

Signature :

ABSTRACT

MULTI-TARGET PARTICLE FILTER BASED TRACK BEFORE DETECT ALGORITHMS FOR SPAWNING TARGETS

EYİLİ, Mehmet
M. Sc., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Mübeccel DEMİREKLER
June 2014, 153 pages

In this work, a Track Before Detect (TBD) approach is proposed for tracking and detection of the spawning targets on the basis of raw radar measurements. The principle of this approach is mainly constructed by multi-model particle filter method. In contrast to the related works in the literature, a novel reduced order dynamic model is introduced and the information about bearing angle derived from the radar measurements is not used in this model to improve the efficiency of the particle filter. Moreover, a new process noise identification method [1] proposed for the classical target tracking is adapted to the TBD framework. The process noise identification is used for the state estimation of the highly maneuvering spawned targets in the presence of non-stationary process noise with unknown parameters. It is shown that this method deals with the sample impoverishment problem which is serious for tracking of the highly maneuvering targets by particle filters. Two different multi-target particle filter based TBD algorithms are developed. These algorithms are confirmed by simulations. Their performances are

analyzed on the basis of the probability of target existences and Root-Mean-Square (RMS) estimation accuracies.

Keywords: Track Before Detect, particle filter, spawning targets, process noise identification

ÖZ

DOĞURAN HEDEFLERİN ÇOKLU HEDEF PARÇACIK FİLTRE TABANLI İZLE BUL ALGORİTMASI İLE TAKİP EDİLMESİ

EYİLİ, Mehmet

Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Mübeccel DEMİREKLER

Haziran 2014, 153 sayfa

Bu çalışmada, ham radar ölçümleri eşliğinde, doğuran hedeflerin izlenmesi ve bulunması için bir İzle Bul yaklaşımı önerilmiştir. Bu yaklaşımın prensibi, başlıca çoklu model parçacık filtresi yöntemi kullanılarak oluşturulmuştur. Literatürdeki çalışmaların aksine, parçacık filtresini daha verimli kullanabilmek için düşük dereceli yeni bir dinamik model sunulmuş olup, bu modelde radar ölçümlerinden elde edilen yanca açı bilgisi kullanılmamıştır. Bununla birlikte, klasik hedef izleme yöntemleri için önerilmiş işlem gürültüsü tanımlama yöntemi [1], İzle Bul yaklaşımına adapte edilmiştir. İşlem gürültüsü tanımlama yöntemi, bilinmeyen parametreli sabit olmayan işlem gürültüleri eşliğinde, yüksek manevralı doğurulan hedeflerin durum vektörünün kestiriminde kullanılmaktadır. Bu yöntemin, yüksek manevralı hedeflerin izlenmesinde parçacık filtreler için ciddi bir sorun olan örnek fakirleşmesi sorunuyla başa çıkabildiği gösterilmiştir. İki farklı çoklu hedef parçacık filtre tabanlı İzle Bul algoritması önerilmiş ve geliştirilen algoritmalar simülasyon sonuçlarıyla doğrulanmıştır. Algoritmaların performansları, çok düşük sinyal gürültü oranı (S/N) değerleri için hedef varlık/yokluk olasılıkları ve kestirim hataları karekök ortalamaları göz önünde bulundurularak analiz edilmiştir.

Anahtar Kelimeler: İzle Bul, parçacık filtresi, doğuran hedefler, işlem gürültüsü tanımlama

To all my friends and my family,

ACKNOWLEDGEMENTS

The most enjoyable part of preparing this thesis is surely extending appreciation to those who helped in its completion. I wish to express my deepest gratitude to my supervisor Prof. Dr. Mübeccel Demirekler for her valuable guidance and support throughout all the thesis preparation period.

I would like to thank Prof. Dr. Seyit Sencer Koç for his help to modelling and data generation used in this thesis.

Many special thanks go to Kenan Ahıska for his valuable, continuous support and encouragements throughout all of the phases of my career.

My friends Serkan Erki and Anıl Koyuncu deserve thanks for their moral support in many ways and making the time enjoyable.

I am also happy to work for this thesis which gives the possibility to meet with beautiful people like Yalçın Demirekler who makes the time enjoyable.

Most importantly, words can't express my appreciation to a wonderful girl, Merve Bilgiç. The completion of this thesis would be impossible without her continuous support and love.

Finally, special thanks go to my family, my father (Aydın Eyili), my mother (Hülya Eyili), my brother (Mert Eyili) and my little sister (Ceren Eyili) for their support and patience over the years.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Target tracking can be defined as estimating the dynamics of the target as a function of time on the basis of sensor measurements like radar or camera. The first step of finding a solution to this problem is to find a model which fits the target dynamics. The model makes it possible to use statistical methods to estimate the states, i.e., dynamics of the target. Filtering which is the main block of these methods is used to extract maximum information from the noisy measurements and the model. The states of the target are estimated according to extracted information from noisy measurements in the filtering part of target tracking.

1.1 Radar

Radar is an acronym for RAdio Detection And Ranging. The main purpose of using radar is to obtain object properties like range, Doppler, bearing and elevation angles by using electromagnetic waves. Radar transmits energy and the transmitted energy reflects back from the object. It uses the energy which is reflected back from the object to obtain object properties.

The usages of radars are highly diverse. They include air-traffic control, navigation, air-defense systems, antimissile systems and weather forecasting applications etc., [2]. Radars for air-defense systems which are used for providing measurements about kinematic properties of objects to target tracking applications are mentioned in this thesis. They will be explained in more detail in Chapter 2.

1.2 Track Before Detect

In classical target tracking methods, raw data is initially thresholded in the detection phase to obtain the plots which usually contain range, Doppler, bearing and elevation angle measurements. Before tracking, the instantaneous threshold based decisions are made with respect to the reflected power measurements obtained from the raw data. The information from the near past is not used in these decisions. This means that the first step of the classical target tracking is a hard decision step. Based on the plots, estimating the state vector of the target that possibly contains the position, velocity and acceleration of the target is the main purpose of the tracking consisting data association and filtering stages as seen in Figure 1.1, see [2].

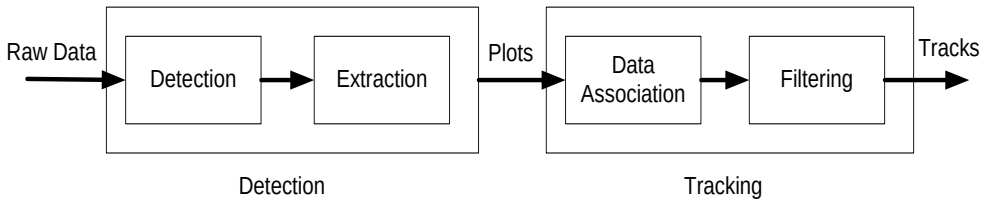
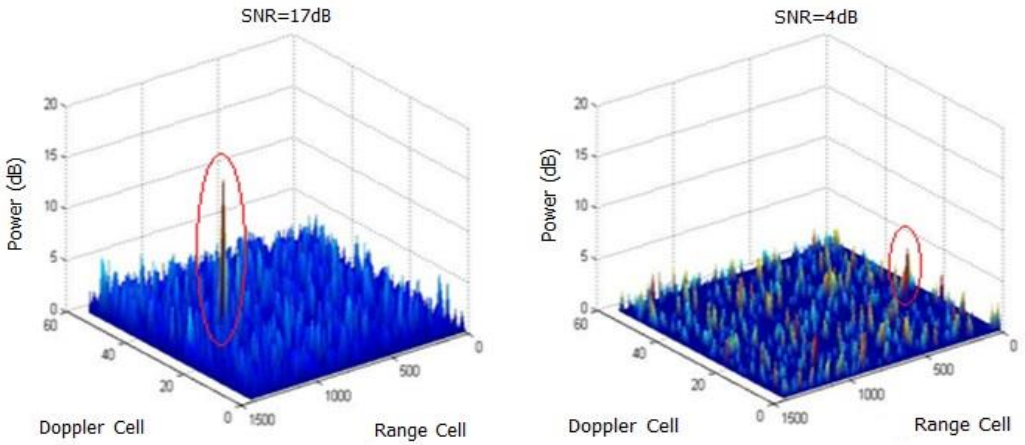


Figure 1.1. Classical target tracking stages (adopted from [2])

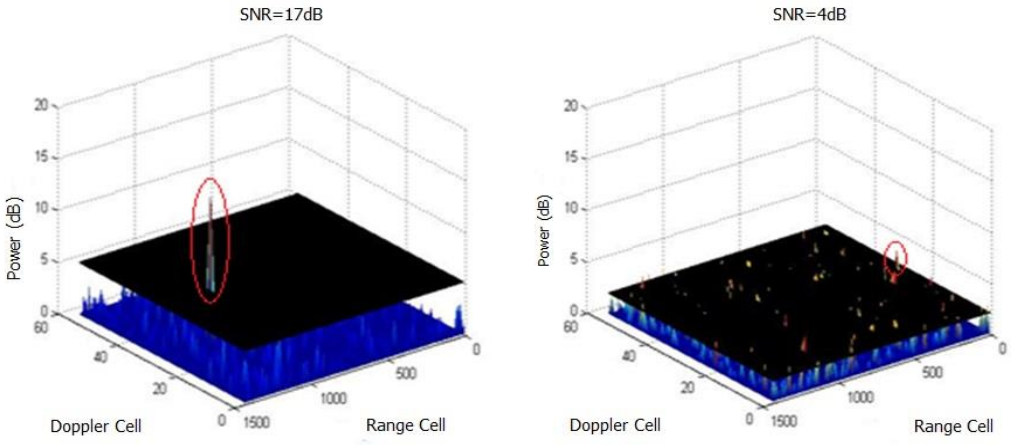
It is not possible to track weak targets with classical target tracking methods because of thresholding made in the detection stage. An example of thresholding is given in Figure 1.3. Thresholding may cause losing useful information gathered from measurements, i.e., returns from weak targets remain below threshold.

TBD algorithms which use the raw non-thresholded data like reflected power measurements are developed for the purpose of detecting and tracking weak targets. An example of raw radar data is given in Figure 1.2. Detection, extraction and data association stages don't take place in TBD; therefore, all information gathered from the raw radar data and integrated over time is used as seen in Figure 1.4. Decisions are made at the end of the process chain.



(a) (b)

Figure 1.2. Power received by the radar. In (a), it is easy to distinguish target from noise since target SNR is high; whereas, in (b), it is not possible to distinguish target from noise since target SNR is low. Note that the reflected power from targets is encircled by red ellipses.



(a) (b)

Figure 1.3. Examples of thresholding which takes place in detection stage in classical target tracking. It is not possible to detect the target when the target SNR is low as in (b). Note that the reflected power from targets is encircled by red ellipses.

Figure 1.4. Data and signal processing in TBD

There are various approaches used to deal with TBD problem. The earliest works [3], [5], [6] on TBD use Hough Transformation. Hough transform assumes linear motion for the target and accumulates the signal for the points on the possible paths of this motion. In other words, its purpose is to detect lines from multiple scans of data in a noisy plane. One of the recent work [4] proposed Multi-Dimensional Hough Transform to deal with TBD problem. It extended Hough Transform, which uses two dimensional data to detect lines, to multi-dimensional data. It is shown that this improves the efficiency of the Hough Transform based TBD.

In [7], [8] and [9], a dynamic programming algorithm is used to deal with TBD problem. Its purpose is to find the most likely state sequence by determining locally optimal state sequences. There is a need for data storage or multiple scans of data for dynamic programming based TBD.

Recent approaches on TBD use grid-based methods as in [10], [11] and [12]. Baum-Welch algorithm which is a fixed-grid method computes the posterior probability mass over fixed grid and propagates it over time. However, fixed-grid methods cause high computational burden since propagating the posterior probability mass in the places where there is negligible probability wastes much of the time. This problem is solved by using dynamic grid-based methods like using particle filters as in [11] and [12]. Particle filters use random samples to construct the posterior pdf.

In this thesis, a particle filter approach is used to perform TBD. Particle filters represent the posterior density function by a set of random samples and their associated weights. The target existence information is estimated on the basis of

these samples and weights. Compared to algorithms mentioned above, there is no need for data storage of past scans. Algorithms based on Hough transform or dynamic programming algorithm use discrete-valued state space; whereas, particle filter uses continuous-valued state space. It is also not restricted to straight-line trajectories like the Hough Transform. A good overview of different TBD algorithms is given in [8] in which several TBD algorithms mentioned above are compared with respect to their detection performances.

Related to work here, on the basis of radar measurements, a particle filter based TBD algorithm is used for a single target case, see [11]. Boers and Driessen extended the single target particle filter based TBD to the multi-target particle filter based TBD in [12]. They proposed an algorithm for the two target case where one of the targets is weak and spawned from the other one which is a strong target, e.g. a missile fired from a helicopter.

1.3 Historical Review of TBD

There are different TBD approaches in literature. One of them is that TBD problem can be solved by binary integration which applies a threshold to each frame and accumulates the binary results for detection of target/s. Hough Transform which is already mentioned in Section 1.2 is applied to TBD for the first time by Smith and Winter in the late 1970s, [5], [6]. Their purpose was constructing two dimensional image on the basis of multiple scans of data obtained from a one-dimensional sensor. Then Carlson published his works about the Hough Transform based TBD on the basis of radar measurements in 1994, [3]. Multi-Dimensional Hough Transform is applied as a TBD processing in 2011, [4].

It can be said that the first nonlinear filtering approach to TBD is developed by Mercier and Maybeck in 1978, [13], [14]. The method they used is based on constructing a nonlinear measurement function between the target state and stacked measurement vector which contains the image pixels. They used Extended Kalman Filter (EKF) by assuming that the noises are Gaussian.

The usage of sampling-based approximation takes place instead of analytic approximation in nonlinear filtering used for TBD. Barniv applied the dynamic programming with discrete state space to TBD in the mid-1980s, [15], [16], [17].

Streit used Histogram Probabilistic Multi-Hypothesis Tracking (H-PMHT) for tackling with TBD problem in 2000 and 2002, [18], [19]. In this method, measurements of underlying density are quantized and the quantized raw data is called as histogram. H-PMHT estimates the states by linking the states of components in histogram over time via the dynamical model of the target and using Expectation-Maximization (EM).

Bruno [10], [20] used the Baum-Welch algorithm which is already mentioned in Section 1.2 to deal with TBD problem in 2001 and 2004, respectively.

Fixed-grid methods like Baum-Welch algorithm cause high computational burden. Therefore, particle filtering which uses dynamic grid is proposed to tackle with this problem. Salmond [21] and Boers [22] introduced particle filter based TBD algorithm in 2001. Boers and Driessen [12] extended the single target particle filter based TBD to multi-target particle filter based TBD in 2004.

Most of the TBD algorithms mentioned above are batch processors; in other words, there is a need for data storage or multiple scans of data. However, it can be seen that the most recent algorithms are recursive algorithms. In this thesis, the focus is a particle filter based TBD algorithm which is a recursive algorithm.

1.4 Thesis Motivation and Objective

TBD is one of the efficient ways of tracking and detecting stealthy targets because of using raw data without thresholding unlike classical tracking techniques. TBD is more important now than at the past because of the huge development in the stealthy target technology as well as a huge increase of small missiles. It is a nonlinear and non-Gaussian tracking problem which is too difficult to be solved even for a single target setting. In this thesis we concentrate on a particle filter

based TBD for a two target setting: one of them is small compared to the other and is spawned from the big one. This setting is an example of moving launch platforms, e.g. a fighter or helicopter with the capability of firing a missile. This scenario is so common in practical view of defense systems. The aim of the multi-target particle based TBD algorithm will be detecting and tracking of both of these targets.

1.5 Thesis Outline

In Chapter 2, radar theory is introduced with the classical target tracking concepts.

Chapter 3 concentrates on particle filtering concept which is used in TBD approach in this thesis.

In Chapter 4, the main problem is stated and the proposed algorithms to solve this problem are presented by explaining each sub-algorithm.

Chapter 5 concentrates on the results and the simulation results of the proposed algorithms. The simulation results for the different scenarios are represented for each algorithm.

Thesis is summarized in Chapter 6 and some future works are suggested in this chapter.

CHAPTER 2

RADAR THEORY

2.1 Radars

2.1.1 Basics of Radars

As it is mentioned in Section 1.1, Radar is an acronym for RAdio Detection And Ranging. Radars are used for detection of targets and obtaining detected targets' kinematic properties like range, Doppler, bearing and elevation angles. In Figure 2.1, positions of radar and target can be seen in the spherical coordinates. We assume a coordinate system that the radar is at the origin, the plane tangent to the Earth is the x-y plane and the axis orthogonal to this plane is the z axis. Range which is the distance between the target and the radar is defined by r; θ is the elevation angle and Φ is the bearing angle of the target.

The transformation from the radar measurements to Cartesian coordinates is given by

$$x = r\cos\theta\cos\Phi, \quad y = r\cos\theta\sin\Phi, \quad z = r\sin\theta \tag{2.1}$$

In this work, a pulse radar simulator is used to generate raw data generation. Pulse radars transmit electromagnetic energy (consecutive pulses) and receive the energy (reflected pulses) reflected back from the target or clutters which are undesired objects, see [33]. The time difference between the time at which the pulses reflected back from the target are received by radar is used for range measurement.

This information is enough to calculate range between the radar and the target since electromagnetic waves' speed is the speed of light, $c = 3\,x\,10^8 m/s$ in free space, and the distance that the electromagnetic waves travel is two times of the range between the radar and the target. Therefore, the range is,

$$r = \frac{c\Delta t}{2} \tag{2.2}$$

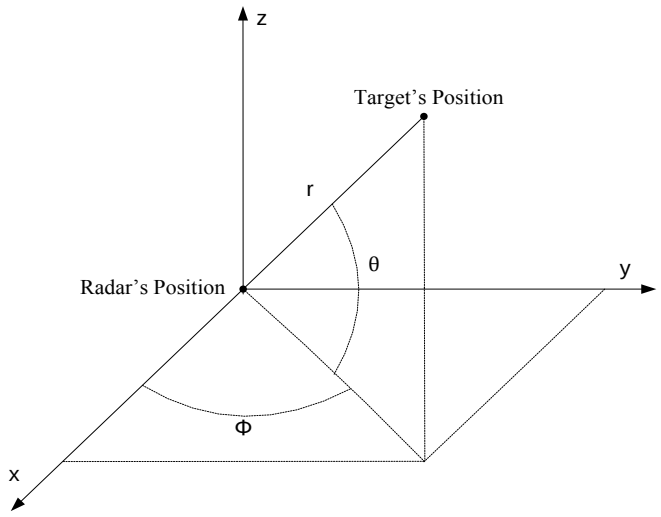


Figure 2.1. Range, elevation and bearing angles in spherical coordinate system

Target's motion causes the Doppler shift on the frequency of the reflected energy according to the transmitted energy. Pulse radars use the Doppler shift (f_d) to measure Doppler of the target. Therefore, Doppler is defined by,

$$\dot{r} = -\frac{f_d \lambda}{2} \tag{2.3}$$

where λ is the wavelength of the transmitted wave. Doppler shift is positive for the objects getting closer to the radar which is the reason for the negative sign in (2.3).

At the receiver part of pulse radars, first match filtering is applied to the received signal, see [23]. It discretizes ranges by filtering the received signal with different

delayed versions of transmitted pulse. Discretized ranges are called as '*range bins*'. Then, Doppler processing performs discrete Fourier transform to the filtered received signal. At the end of Doppler processing, '*Doppler bins*' are constructed. Range and Doppler bins constitute the '*Range-Doppler Matrix*' which gives the information about the received energy for each range and Doppler cell. An illustration of Range-Doppler Matrix can be seen in Figure 2.2.

The resolution of each cell is defined by

$$\Delta r = \frac{c\tau}{2}, \quad \Delta \dot{r} = \frac{\lambda}{2T} \tag{2.4}$$

where Δr is the range resolution (m), $\Delta \dot{r}$ is the Doppler resolution (m/sec), τ is the pulse width (sec), λ is the radar wavelength (m) and T is the pulse integration time (sec). As it can be seen in (2.4), the resolutions depend on some radar properties like τ, λ and T.

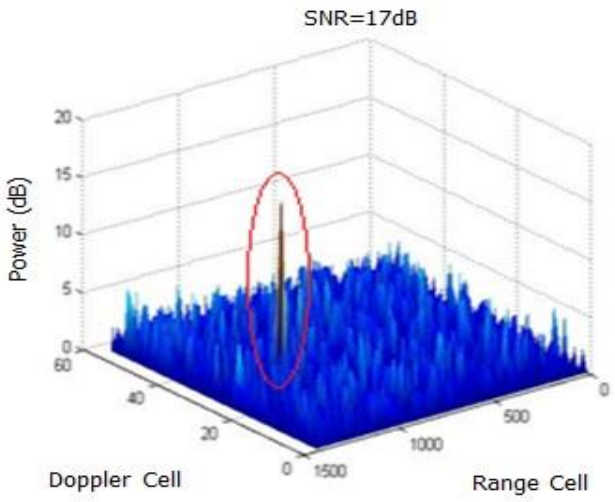


Figure 2.2. An illustration of range-Doppler matrix

2.1.2 The Radar Range Equation

The radar range equation describes the relationship between the received energy P_r and the transmitted energy P_t in terms of radar and target properties and environmental effects.

The transmitted energy P_t is related with the amplitude of the pulse envelope (denoted by $A(t)$) of a transmitted pulse whose form is given by

$$s_t(t) = A(t)e^{j[2\pi(f_t)t+\psi(t)]} \tag{2.5}$$

where f_t is the carrier frequency and $\psi(t)$ is the phase modulation of the transmitted pulse. The amplitude of the pulse envelope $A(t)$ is an ideal rectangle with amplitude of A and the transmitted energy P_t is proportional to amplitude of the pulse envelope A.

The received energy P_r is related with the received echo amplitude denoted by $R(t)$. Received echo can be written in the form of

$$s_r(t) = R(t - t_0)e^{j[2\pi(f_t+f_d)(t-t_0)+\varphi(t)]} + \eta(t) \tag{2.6}$$

where f_d is the Doppler shift, $t_0 = 2r/c$ is the time passed between the start time of transmission of the pulse and the time at which radar receives the pulse, $\varphi(t)$ is the phase modulation of the received echo and $\eta(t)$ is the noise of the receiver. The received echo amplitude $R(t - t_0)$ is also an ideal rectangle with amplitude of R since the amplitude of the pulse envelope $A(t)$ is an ideal rectangle. The received echo is demodulated to be used for match filtering [23] and Doppler processing, respectively. At the end of the demodulation process, the demodulated received echo is in the form of

$$s_r(t) = Re^{j[\varphi(t)]} + \eta_d(t) \tag{2.7}$$

The received energy P_r is proportional to the received echo amplitude R. The power density at range r for an isotropic antenna which transmits spherical waves is given by

$$Power\ density\ \ at\ range\ r\ for\ an\ isotropic\ antenna \propto \frac{P_t}{4\pi r^2} \qquad (2.8)$$

where r is the range between the radar and the target. A directive antenna is used in radars for strengthening the received power density by concentrating the transmitted energy at a specific direction. Therefore, there is a gain called the gain of transmitter antenna G_t in the power density for a directive antenna over for an isotropic antenna. Therefore, the power density at range r for a directive antenna is given by

$$Power\ density\ \ at\ range\ r\ for\ a\ directive\ antenna \propto \frac{P_t G_t}{4\pi r^2} \qquad (2.9)$$

The scattering properties like shape, size and material of the target also affect the received power density. A parameter called *'Radar Cross Section (RCS)'* is used to characterize these properties; in other words, RCS, denoted by σ, is a measure of energy which is scattered from target. Therefore, the reflected power density ($P_{reflected}$) near the target is given by

$$P_{reflected}\ near\ the\ target \propto \frac{P_t G_t \sigma}{4\pi r^2} \qquad (2.10)$$

However, as going far from the target, the reflected power density decreases by $1/(4\pi r^2)$. Therefore, the received power density at radar is given by

$$P_r \propto \frac{P_t G_t}{4\pi r^2} \frac{\sigma}{4\pi r^2} A_e \qquad (2.11)$$

where A_e is the effective area of the receiving antenna of the radar. It is a measure of reflected power density captured by radar antenna. The effective area of the receiving antenna of the radar is defined by

$$A_e = \frac{\lambda^2 G_r}{4\pi} \tag{2.12}$$

where G_r is the receiver antenna gain. (2.11) is given for ideal radar operating in free space conditions. However, there are losses in real conditions. Total loss, denoted by L, is the combination of the system and the atmospheric losses. Therefore, the received power density can be rewritten as

$$P_r = \frac{P_t G_t}{4\pi r^2} \frac{\sigma}{4\pi r^2} \frac{\lambda^2 G_r}{4\pi} \frac{1}{L} \tag{2.13}$$

The received power density in (2.13) is target originated. Radars also receive some noise power originated from many sources like ground, sun, atmosphere and the receiver part of radar itself. The total received noise power P_n can be written as

$$P_n = kT_0 B_n F_n \tag{2.14}$$

where k is Boltzmann's constant, $T_0 = 290\,K$ is the standard temperature, B_n is noise bandwidth of the receiver and F_n is the noise figure of the receiver.

Signal to Noise Ratio (SNR) is an important parameter for the detection part of tracking algorithms. It is calculated as follows

$$SNR = \frac{P_r}{P_n} = \frac{P_t G_t G_r \lambda^2 \sigma}{(4\pi)^3 r^4 k T_0 B_n F_n L} \tag{2.15}$$

The SNR value is very important for detection of targets in target tracking. As mentioned earlier, threshold based decisions are made with respect to the SNR value of targets in classical target tracking.

2.2 Target Tracking

Target tracking will be described in this section by covering both classical tracking and TBD.

2.2.1 Classical Target Tracking

Classical tracking methods consist of detection, extraction, data association and filtering stages as mentioned earlier. In this section, these stages will be described in more details.

2.2.1.1 Detection

In classical target detection, the received power in each range-Doppler cell is compared to a predefined threshold. If the received power exceeds the predefined threshold, it is declared that there is '*detection*' in that cell. Therefore, determining the target's presence is the main purpose of this stage. However, there are two kinds of error encountered in this stage. One of them is '*false alarm*'. It occurs when the received power exceeds threshold and there is no target in the surveillance region. The other one is '*miss detection*'. It occurs when the received power does not exceed threshold and there is at least one target in the surveillance region. The probability of false alarm and the probability of miss detection are denoted by P_{FA} and P_{MD}, respectively.

$$P_{MD} = 1 - P_D \tag{2.16}$$

where P_D is the probability of detection.

Determining the threshold is a difficult problem. The reason is that the probability of false alarm decreases and the probability of miss detection increases as the value of threshold increases; whereas, the probability of false alarm increases and the probability of miss detection decreases as the value of threshold decreases.

2.2.1.2 Extraction

After detection, '*extraction*' takes place. It is also called 'clustering'. As it can be understood from the name of clustering, this stage makes clusters of detections which are close to each other since close detections most likely belong to the same target. The center of each cluster is assumed to be the position of the target. At the end of extraction, plots which are the inputs of the tracking part of the classical target tracking are generated. As mentioned in Section 1.2, plots include range, Doppler, bearing and elevation angles of the targets.

2.2.1.3 Data Association

As the measurements are generated in the extraction stage, they are either associated to the existing tracks or are used to generate new tracks so measurements are mainly used for track initiation or track maintenance in target tracking. The aim of data association stage is to determine whether a measurement is used for track initiation or for track maintenance and determining the track which it belongs to if it is used for track maintenance.

2.2.1.4 Filtering

The aim of the '*filtering*' stage is to extract maximum information about the target's state vector from the system models and the associated measurements. In filtering, dynamic states like position, velocity and acceleration of the target are estimated by constructing the posterior probability density function $p(s_k|Z_k)$ where s_k is the target state at time k and Z_k is the sequence of measurements up to time k. Filtering is explained in Chapter 3 in more details.

2.2.2 Track Before Detect

The classical target tracking methods use thresholded raw data. The probability of target detection and the probability of false alarm are affected by the choice of threshold used in the detection part. It is not a big problem to threshold raw radar

data for high SNR targets; whereas, it is not the same for low SNR targets because a low value of threshold that is necessary to detect low SNR targets causes high density of false alarms. The problem of high density of false alarms can be solved by using more complex data association algorithms. Therefore, using un-thresholded raw data is more advantageous for low SNR targets.

Track Before Detect (TBD) is a target tracking algorithm using un-thresholded raw data for simultaneous detection and tracking. It has better performance on detecting and tracking low SNR targets compared to the classical approach.

CHAPTER 3

FILTERING

As mentioned earlier, the main purpose of filtering is to estimate the state vector of the target/s based on noisy measurements which most likely contain position and velocity of the target/s. There are different ways of implementing filtering in target tracking. In this section, one of the recursive filtering method called '*particle filter*' is explained in detail.

3.1 Bayesian Estimation

As mentioned earlier, the aim of filtering is to extract maximum information about the target's state vector from the system models and noisy measurements. System model consists of system dynamic model and measurement model.

System dynamic model describes the motion of the target in the state space. It is basically shown in discrete state space as follows.

$$s_{k+1} = f_k(s_k, v_k), \qquad k \in \mathbb{N} \tag{3.1}$$

where s_k is the state vector of the target, f_k is the function of system dynamics, and v_k is the process noise at time step k. The reason to add process noise term in target dynamics is to model the unexpected motion of the target.

The other model called the measurement model, describes the behavior of the sensor measurements. The discrete time measurement model is basically given by

$$z_k = h_k(s_k, w_k), \qquad k \in \mathbb{N} \tag{3.2}$$

where h_k is the measurement function and w_k is the measurement noise at time step k. Measurement noise term models sensor errors in the measurements. Note that the process noise v_k and measurement noise w_k are assumed to be white Gaussian in this work.

Bayesian estimation mainly contains two stages: prediction and update. At the end of these stages, it is expected to obtain the posterior probability density function $p(s_{k+1}|Z_{k+1})$ where Z_{k+1} is the set of measurements up to time step $k+1$.

The prediction part computes the prediction probability density function $p(s_{k+1}|Z_k)$; in other words, the prediction of the state vector at time step $k+1$ is made by using the measurements up to time step k. The prediction probability density function $p(s_{k+1}|Z_k)$ is represented as in (3.3).

$$p(s_{k+1}|Z_k) = \int p(s_{k+1}|s_k, Z_k)p(s_k|Z_k)ds_k \tag{3.3}$$

It can be rewritten as follows.

$$p(s_{k+1}|Z_k) = \int p(s_{k+1}|s_k)p(s_k|Z_k)ds_k \tag{3.4}$$

since $p(s_{k+1}|s_k, Z_k) = p(s_{k+1}|s_k)$ as seen in (3.1). The posterior pdf $p(s_k|Z_k)$ at time step k is assumed to be available and $p(s_{k+1}|s_k)$ is defined by system dynamics model defined in (3.1).

The update part of the Bayesian estimation involves the update of the prediction pdf by using the latest measurement. The prediction pdf which is broadened by process noise is tightened in the update part by using the knowledge extracted from the latest measurement. The posterior pdf $p(s_{k+1}|Z_{k+1})$ is calculated as follows in the update stage.

$$p(s_{k+1}|Z_{k+1}) = p(s_{k+1}|z_{k+1}, Z_k)$$

$$= \frac{p(z_{k+1}|s_{k+1}, Z_k)p(s_{k+1}|Z_k)}{p(z_{k+1}|Z_k)} \tag{3.5}$$

$$= \frac{p(z_{k+1}|s_{k+1})p(s_{k+1}|Z_k)}{p(z_{k+1}|Z_k)}$$

where

$$p(z_{k+1}|Z_k) = \int p(z_{k+1}|\, s_{k+1})\, p(s_{k+1}|Z_k) ds_{k+1} \tag{3.6}$$

The prediction pdf $p(s_{k+1}|Z_k)$ is defined in the prediction stage and $p(z_{k+1}|s_{k+1})$ can be calculated by using sensor measurement model defined in (3.2). Therefore, the posterior probability density function at time step $k+1$, $p(s_{k+1}|Z_{k+1})$ can be obtained at the end of update part in Bayesian estimation.

There are different methods to implement recursive Bayesian estimation; namely, Kalman filter, extended Kalman filter [24], unscented Kalman filter [25], particle filter etc. The preference among these methods is done according to the properties of the system. For instance, Kalman filter can be applied to linear and Gaussian systems. However, the system, in most of the applications, is nonlinear and non-Gaussian. Extended Kalman filter, unscented Kalman filter and particle filter are used for these type of systems. Extended Kalman filter applies linearization to nonlinear functions f_k and h_k defined in (3.1) and (3.2) to make the system be appropriate for applying Kalman filter; whereas, unscented Kalman filter uses approximation of the posterior pdf $p(s_{k+1}|Z_{k+1})$ by a Gaussian density which is defined by a set of sample points chosen deterministically. In this thesis, particle filtering which represents the posterior density function by a set of random samples and their associated weights is used.

3.1.1 Particle Filter

Particle filtering is developed for the state vector estimation in nonlinear and non-Gaussian systems. As mentioned earlier, it represents the posterior pdf by a set of random samples and their associated weights. In the remaining parts of this chapter we will give a basic information on particle filtering which explains the tools used during filtering, filtering steps, some discussion on the performance of the algorithms and the precautions taken to avoid some potential problems.

3.1.1.1 Monte Carlo Integration

It is not easy to take integrals in (3.4) and (4.22) for nonlinear and non-Gaussian systems. Therefore, the particle filter uses '*Monte Carlo Integration*' technique to approximate these integrals. Monte Carlo integration [26] can be applied to integrals in the form of

$$I = \int f(s)\pi(s)ds, \qquad s \in \mathbb{R}^n \tag{3.7}$$

where

$$\int \pi(s)ds = 1\,, \qquad \pi(s) > 0 \qquad \forall s \tag{3.8}$$

Monte Carlo integration is based on an assumption that the integral in (3.7) can be rewritten as follows by drawing N samples $\{s^i; i = 1, \ldots, N\}$ from $\pi(s)$, see [27].

$$I = \frac{1}{N}\sum_{i=1}^{N} f(s^i) \tag{3.9}$$

$\pi(s)$ refers to the posterior pdf $p(s_k|Z_k)$ in Bayesian solution. In general, it may not be possible to draw samples from this distribution since it is not available. Therefore, '*importance sampling*' is introduced for the solution of this problem.

3.1.1.2 Importance Sampling

As mentioned earlier, Monte Carlo integration cannot be applied effectively since the posterior pdf is generally not available. If an '*importance function*' from which it is possible to draw samples effectively is introduced and these samples are weighted correctly, it is possible to apply Monte Carlo integration.

The integral in (3.7) can be rewritten as follows

$$I = \int f(s)\frac{\pi(s)}{q(s)}q(s)ds \tag{3.10}$$

where $q(s)$ is the importance function. By drawing the samples from the importance function $q(s)$, the integral in (4.22) can be rewritten as follows.

$$I = \frac{1}{N}\sum_{i=1}^{N} f(s^i)\tilde{w}(s^i) \tag{3.11}$$

where

$$\tilde{w}(s^i) = \frac{\pi(s^i)}{q(s^i)} \tag{3.12}$$

$\tilde{w}(s^i)$ is the weight of the ith drawn sample. The weights of the samples are normalized as follows.

$$I = \frac{\frac{1}{N}\sum_{i=1}^{N} f(s^i)\tilde{w}(s^i)}{\frac{1}{N}\sum_{j=1}^{N}\tilde{w}(s^j)} = \sum_{i=1}^{N} f(s^i)w(s^i) \tag{3.13}$$

where

$$w(s^i) = \frac{\tilde{w}(s^i)}{\sum_{j=1}^{N}\tilde{w}(s^j)} \tag{3.14}$$

3.1.1.3 Sequential Importance Sampling

Importance sampling is a conceptual solution to the target tracking problem in Bayesian framework. Recursive importance sampling and correct weighting of the samples are the most important steps in this type of recursive state estimation.

Note that we denote all states up to time step k $\{s_j, j = 0, \ldots, k\}$ as S_k. Assume that the joint posterior density at time step k, $p(S_k|Z_k)$, is approximated as follows, see [28].

$$p(S_k|Z_k) = \sum_{i=1}^{N} w_k^i \delta(S_k - S_k^i) \tag{3.15}$$

where w_k^i is the normalized weight of the state sequence S_k^i. Let us denote the importance function from which the samples S_k^i are drawn as $q(S_k|Z_k)$. Then, the importance weights can be defined as follows.

$$\widetilde{w}_k^i \propto \frac{p(S_k^i|Z_k)}{q(S_k^i|Z_k)} \tag{3.16}$$

If the importance function can be represented as in (4.22),

$$q(S_k|Z_k) = q(s_k|S_{k-1}, Z_k)q(S_{k-1}|Z_{k-1}) \tag{3.17}$$

then the samples of states up to time step k, $S_k^i \sim q(S_k|Z_k)$, can be obtained by adding the new sample $s_k^i \sim q(s_k|S_{k-1}, Z_k)$ to the samples of the set of states up to time step $k-1$, $S_{k-1}^i \sim q(S_{k-1}|Z_{k-1})$.

The joint posterior density $p(S_k|Z_k)$ can be obtained as follows.

$$
\begin{aligned}
p(S_k|Z_k) &= \frac{p(z_k|S_k, Z_{k-1})p(S_k|Z_{k-1})}{p(z_k|Z_{k-1})} \\
&= \frac{p(z_k|S_k, Z_{k-1})p(s_k|S_{k-1}, Z_{k-1})p(S_{k-1}|Z_{k-1})}{p(z_k|Z_{k-1})} \\
&= \frac{p(z_k|s_k)p(s_k|s_{k-1})}{p(z_k|Z_{k-1})} p(S_{k-1}|Z_{k-1}) \\
&\propto p(z_k|s_k)p(s_k|s_{k-1})p(S_{k-1}|Z_{k-1})
\end{aligned}
\tag{3.18}
$$

The importance weights can be obtained recursively by substituting (3.17) and (4.22) into (3.16) as follows.

$$
\begin{aligned}
\widetilde{w}_k^i &\propto \frac{p\left(z_k|s_k^i\right)p\left(s_k^i|s_{k-1}^i\right)p(S_{k-1}^i|Z_{k-1})}{q\left(s_k^i|S_{k-1}^i, Z_k\right)q(S_{k-1}^i|Z_{k-1})} \\
&= w_{k-1}^i \frac{p\left(z_k|s_k^i\right)p(s_k^i|s_{k-1}^i)}{q(s_k^i|S_{k-1}^i, Z_k)}
\end{aligned}
\tag{3.19}
$$

If the importance function is selected in the form of

$$
q(s_k|S_{k-1}, Z_k) = q(s_k|s_{k-1,}z_k) \tag{3.20}
$$

then the posterior pdf $p(s_k|Z_k)$ can be calculated with no need to store the samples of sets of states up to time step $k-1$, $\{S_{k-1}^i\}_{i=1}^N$, and the measurements up to time $k-1$, Z_{k-1}. In this case, the importance weights can be recalculated as in (4.22).

$$
\widetilde{w}_k^i = w_{k-1}^i \frac{p\left(z_k|s_k^i\right)p(s_k^i|s_{k-1}^i)}{q(s_k^i|s_{k-1}^i, z_k)} \tag{3.21}
$$

Therefore, the posterior pdf $p(s_k|Z_k)$ can be expressed as follows.

$$p(s_k|Z_k) \approx \sum_{i=1}^{N} w_k^i \delta(s_k - s_k^i) \tag{3.22}$$

This computation shows that the posterior pdf can be represented by a set of random samples and their associated weights. Recursive estimations can be made on the basis of these samples and weights as recursive measurements are received. Moreover, as $N \to \infty$, the approximation in (4.22) approaches to the true posterior pdf, see [29].

The particle filtering is developed based on Sequential Importance Sampling (SIS) whose pseudocode is given in Pseudocode 1. The choice of the importance function is very important for the performance of a particle filter. The ideal importance function is the posterior pdf $p(s_k|Z_k)$. However, it is not possible to draw samples from the posterior pdf in most of the time. Therefore, it is aimed to select an importance function which is similar to the posterior density $p(s_k|Z_k)$.

Pseudocode 1 Sequential Importance Sampling

$\left[\{s_k^i, w_k^i\}_{i=1}^N\right] = \text{SIS}\left[\{s_{k-1}^i, w_{k-1}^i\}_{i=1}^N, z_k\right]$

- FOR $i = 1 : N$
  - Draw samples $s_k^i \sim q(s_k^i|s_{k-1}^i, z_k)$
  - Assign weights to the samples as follows

$$\widetilde{w}_k^i = w_{k-1}^i \frac{p(z_k|s_k^i)p(s_k^i|s_{k-1}^i)}{q(s_k^i|s_{k-1}^i, z_k)}$$

- END FOR
- FOR $i = 1 : N$
  - Normalize the importance weights as follows

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

- END FOR

3.1.1.4 Degeneracy and Resampling

The weights of particles are obtained by recursive application of the importance weight update in SIS particle filter. That causes a dramatic increase in the variance of the importance weights, [28]. After a few number of recursive time steps, there remain a few particles whose weights are not negligible. This is an unavoidable problem, called '*degeneracy*' in SIS particle filter, [27].

The severity of the degeneracy problem can be measured by effective number of particles which is denoted as N_{eff}. Effective number of particles can be approximately calculated as follows, see [30].

$$N_{eff} \cong \frac{1}{\sum_{i=1}^{N} (w_k^i)^2} \tag{3.23}$$

The less effective number of particles, the more severe the degeneracy problem is. The solution of this problem is resampling.

The purpose of the resampling method is eliminating the particles with low weights and generating copies of particles with high weights. At the end of resampling, all particles have the same weight which is equal to $1/N$. In other words, a new set of particles $\{s_k^{i*}\}_{i=1}^{N}$ with uniform weights are obtained from the set of particles $\{s_k^i\}_{i=1}^{N}$ with weights w_k^i by resampling N times from the posterior pdf $p(s_k|Z_k)$ which is represented in (4.22).

$$p(s_k|Z_k) \approx \sum_{i=1}^{N} w_k^i \delta(s_k - s_k^i) \tag{3.24}$$

There are different methods available for resampling like stratified and residual resampling, [31]. The pseudocode of the systematic resampling algorithm used in this thesis is given in Pseudocode 2.

Pseudocode 2 Resampling

$\left[\{s_k^{j*}, w_k^{j*}, i^j\}_{j=1}^N\right] = \text{Resample}\left[\{s_k^i, w_k^i\}_{i=1}^N\right]$

- Initialize the cumulative sum of importance weights: $csw_1 = w_k^1$
- FOR $i = 2 : N$
    - Calculate the cumulative sum of importance weights:

$$csw_i = csw_{i-1} + w_k^i$$

- END FOR
- Start at the initial cumulative sum of importance weights: $i = 1$
- Determine the starting point: $a_1 \sim \mathcal{U}[0, N^{-1}]$
- FOR $j = 1 : N$
    - Determine the place of the random variable a_j on the cumulative sum of importance weights: $a_j = a_1 + N^{-1}(j-1)$
    - WHILE $a_j > csw_i$
        - ❖ $i = i + 1$
    - END WHILE
    - Determine the new sample: $s_k^{j*} = s_k^i$
    - Assign the weight of the new sample: $w_k^{j*} = N^{-1}$
    - Store the parent sample of the new sample: $i^j = i$
- END FOR

3.1.1.5 Sequential Importance Resampling

SIS algorithm forms the basis of the Sequential Importance Resampling (SIR) which applies resampling algorithm at every time step. In this thesis, the transitional prior, $p(s_k|s_{k-1}^i)$, is used as the importance density. The advantage of selecting the transitional prior as importance density is that it is easy to draw samples from it and compute the weights of these samples by substituting the transitional prior in (3.21). The weights of the samples can be computed as follows.

$$\widetilde{w}_k^i \propto w_{k-1}^i p(z_k|s_k^i) \tag{3.25}$$

The normalized weights, w_{k-1}^i, in (4.22) can be omitted since SIR algorithm applies resampling at every time step and all samples have the same weight which equals to $1/N$. Therefore, the weights of the samples can be rewritten as follows.

$$\widetilde{w}_k^i \propto p(z_k|s_k^i) \tag{3.26}$$

The transitional prior pdf does not contain any knowledge of the measurement z_k. Therefore, SIR algorithm which uses the transitional prior as importance density may be inefficient when the target maneuvers, [27].

The pseudocode of the SIR algorithm is given in Pseudocode 3. SIR algorithm forms the basis of the TBD algorithm used in this thesis.

Pseudocode 3 SIR Algorithm

$\left[\{s_k^i\}_{i=1}^N\right] = \text{SIR}\left[\{s_{k-1}^i\}_{i=1}^N, z_k\right]$

- FOR $i = 1 : N$
    - Draw the particle: $s_k^i \sim p(s_k|s_{k-1}^i)$
    - Compute the weight of the particle: $\widetilde{w}_k^i = p(z_k|s_k^i)$
- END FOR
- FOR $i = 1 : N$
    - Normalize the weights:

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

- END FOR
- Resample by using Pseudocode 2

CHAPTER 4

DETECTION AND TRACKING OF SPAWNING TARGETS BY USING PARTICLE FILTER BASED TBD APPROACH

4.1 Problem Statement

In this work, a particle filter based TBD algorithm is proposed for the two target case where one of the targets is weak and spawned from the main platform which is a strong target, e.g. a missile fired from a fighter airplane. This is an important scenario for defense systems. Moreover, it is a difficult problem since the reflected power from the main platform is so strong that makes the detection of the weak target so complicated.

TBD applications given in the literature use the constant velocity model, see [11], [12]. The measurement used in this study and the relevant work in the literature is the range-Doppler matrix. Without elevation and bearing angles, this model becomes unobservable. Since the aim is to get the existence of the spawned target, we have introduced a novel reduced order model for a nearly constant velocity motion of the target to improve the efficiency of the particle filter. Furthermore, in contrast to the related works in literature like [2] and [12], bearing angle between the radar and the target is also not included in the measurement space for the same purpose.

The spawned targets are possibly highly maneuvering, i.e. missiles. The sample impoverishment problem becomes serious in the case of highly maneuvering

targets. In this thesis, a new process noise identification method [1] proposed for the classical target tracking methods is adapted to the TBD framework to deal with the sample impoverishment problem. Furthermore, in some studies, target SNR is assumed to be constant and known as in [12], [22] and [32]. Our study is not restricted to the case of constant and known SNR as in [11]. In addition to this, SNR values of targets are also estimated in this study.

In this thesis, two different particle filter based TBD algorithms are developed. The algorithms mentioned above are explained in details in the following sections.

4.2 System Setup

A general nonlinear discrete time dynamic system can be represented as in (3.1) and (3.2). For our problem, although this general representation is valid, the knowledge of the target existence should be included. The target existences are modelled as a 'mode' variable which is denoted by m_k.

The effects of the modes in system behavior are added to (3.1) and (3.2) as follows.

$$s_{k+1} = f_k(s_k, m_k, v_k) \tag{4.1}$$

$$z_k = h_k(s_k, m_k, w_k) \tag{4.2}$$

The transitions between the modes are modelled by the Markov transition matrix, Π, which is composed of the transitional probabilities, π_{ij}, defined as in (4.3).

$$\pi_{ij} \triangleq P\{m_k = j | m_{k-1} = i\} \tag{4.3}$$

where

$$\pi_{ij} \geq 0 \quad \text{and} \quad \sum_j \pi_{ij} = 1 \tag{4.4}$$

The systems represented by (4.1), (4.2) and (4.3) are called as jump Markov systems and this type of systems are used in this thesis.

4.3 Algorithm 1

As mentioned earlier, the aim of both of the algorithms proposed in this study is to track and detect the main target and the spawned one. The main difference between the two proposed algorithms is the system dynamic model and the measurement model they use. The main principle of Algorithm 1 is to track the main platform and the spawned target separately by using a single target system model. It achieves this by modifying the raw radar data according to the information gathered from the near past. This algorithm is explained in detail in the following sections.

4.3.1 Conceptual Solution of the TBD Problem in the Bayesian Framework

Algorithm 1 is for tracking of a single target. For this algorithm, the mode variable, m_k, represents the two hypotheses:

- $m_k = 0$: There is no target present.
- $m_k = 1$: There is one target present.

The transition from the absence of the target to the presence of the target is called '*birth of the target*' and its probability is denoted by P_b. The transition from the presence of the target to the absence of the target is called '*death of the target*' and its probability is denoted as P_d. Then, the Markov transition matrix can be written as follows.

$$\Pi = \begin{pmatrix} 1 - P_b & P_b \\ P_d & 1 - P_d \end{pmatrix} \tag{4.5}$$

TBD problem can be solved in the Bayesian framework, see [27]. As mentioned before, Bayesian approach mainly contains two stages: prediction and update. The

aim is to construct the joint posterior pdf at time step k, $p(s_k, m_k|Z_k)$ by using the joint posterior pdf at time step $k-1$, $p(s_{k-1}, m_{k-1}|Z_{k-1})$, when the measurement z_k is available.

Given that the measurements up to time step k, Z_k, the probability of the target existence denoted by P_k can be represented as follows.

$$P_k = \mathrm{P}\{m_k = 1|Z_k\} \tag{4.6}$$

It can be derived from the joint posterior $p(s_k, m_k = 1|Z_k)$ by marginalization.

$$P_k = \int p(s_k, m_k = 1|Z_k) ds_k \tag{4.7}$$

The joint posterior pdf, $p(s_k, m_k = 1|Z_k)$, is obtained in the update part of the Bayesian solution as in (4.22).

$$p(s_k, m_k = 1|Z_k) = \frac{p(z_k|s_k, m_k = 1)p(s_k, m_k = 1|Z_{k-1})}{p(z_k|Z_{k-1})} \tag{4.8}$$

where $p(z_k|Z_{k-1})$ is a normalizing factor and $p(z_k|s_k, m_k = 1)$ is the likelihood function which can be obtained from the measurement model. The prediction density, $p(s_k, m_k = 1|Z_{k-1})$, can be obtained as follows.

$$p(s_k, m_k = 1|Z_{k-1})$$

$$= \int p(s_k, m_k = 1|s_{k-1}, m_{k-1} = 1, Z_{k-1}) p(s_{k-1}, m_{k-1} = 1|Z_{k-1}) ds_{k-1} +$$

$$\int p(s_k, m_k = 1|s_{k-1}, m_{k-1} = 0, Z_{k-1}) p(\, s_{k-1}, m_{k-1} = 0|Z_{k-1}) ds_{k-1} \tag{4.9}$$

where

$$p(s_k, m_k = 1 | s_{k-1}, m_{k-1} = 1, Z_{k-1})$$

$$= p(s_k | s_{k-1}, m_k = 1, m_{k-1} = 1)\mathrm{P}\{m_k = 1 | m_{k-1} = 1\}$$

$$= p(s_k | s_{k-1}, m_k = 1, m_{k-1} = 1)(1 - P_d) \quad (4.10)$$

and

$$p(s_k, m_k = 1 | s_{k-1}, m_{k-1} = 0, Z_{k-1})$$

$$= p(s_k | s_{k-1}, m_k = 1, m_{k-1} = 0)\mathrm{P}\{m_k = 1 | m_{k-1} = 0\} = p_b(s_k)P_b \quad (4.11)$$

The probability density $p(s_k | s_{k-1}, m_k = 1, m_{k-1} = 1)$ which is given in (4.10) can be obtained from the system dynamic model given in (4.1). $p_b(s_k)$ is the target birth density which is assumed to be known and explained in the following sections.

This conceptual solution given above is implemented in Algorithm 1.

4.3.2 System Dynamic Model

As mentioned earlier, the system dynamic model is in the form of

$$s_{k+1} = f_k(s_k, m_k, v_k) \quad (4.12)$$

where $m_k \in \{0, 1\}$. '0' denotes that there is no target present and '1' denotes that there is one target present. Therefore, (4.12) can rewritten as in (4.13).

$$s_{k+1} = \begin{cases} f_k(s_k, v_k), & if\ there\ is\ one\ target\ present\ (m_k = 1) \\ undefined, & if\ there\ is\ no\ target\ present\ (m_k = 0) \end{cases} \quad (4.13)$$

The targets are assumed to move with a constant velocity in many applications given in the literature. The state space used for the constant velocity model contains the positions and velocities in Cartesian coordinates in literature as in (4.14).

$$s_k = [x_k \;\; vel_k^x \;\; y_k \;\; vel_k^y \;\; z_k \;\; vel_k^z \;\; I_k]^T \tag{4.14}$$

where x_k, y_k, z_k are the positions, vel_k^x, vel_k^y, vel_k^z are the velocities and I_k is the SNR value of the target. The state of target SNR, I_k, is augmented to the state vector since the SNR values of the targets are intended to be estimated.

As mentioned earlier, the raw radar data contains the power measurements for each range-Doppler cell. The constant velocity model together with constant target SNR is a linear state space model but the measurements are nonlinear. Due to the nonlinear nature of the measurements, most of the applications given in the literature, use particle filtering. Particle filtering with a state space of dimension 7 usually gives unsatisfactory results. This problem is not stated in the dim target tracking literature that uses particle filtering. We have observed that this is due to the unobservability of the states, i.e., the measurements and the state model defined above is not an observable pair. So we decided to use a novel model with less number of states which are observable except the state of u_k which is defined in (4.16). The state vector in this model is shown in (4.15).

$$s_k = [r_k \;\; d_k \;\; u_k \;\; I_k]^T \tag{4.15}$$

where r_k and d_k is the range and the Doppler of the target, respectively. u_k is defined as follows.

$$u_k = (vel_k^x)^2 + (vel_k^y)^2 + (vel_k^z)^2 \tag{4.16}$$

Note that the range and the Doppler of the target are defined in terms of the positions and the velocities in Cartesian coordinates as in (4.17) and (4.18), respectively.

$$r_k = \sqrt{x_k^2 + y_k^2 + z_k^2} \tag{4.17}$$

$$d_k = \frac{1}{\sqrt{x_k^2 + y_k^2 + z_k^2}} (x_k vel_k^x + y_k vel_k^y + z_k vel_k^z) \tag{4.18}$$

The time update of the range is done according to the (4.19).

$$\begin{aligned} r_{k+1} &= \sqrt{x_{k+1}^2 + y_{k+1}^2 + z_{k+1}^2} \\ &= \sqrt{(x_k + \mathrm{T} vel_k^x)^2 + \left(y_k + \mathrm{T} vel_k^y\right)^2 + (z_k + \mathrm{T} vel_k^z)^2} \\ &= \sqrt{x_k^2 + y_k^2 + z_k^2 + 2\mathrm{T}\left(x_k vel_k^x + y_k vel_k^y + z_k vel_k^z\right) + \mathrm{T}^2 u_k} \\ &= \sqrt{r_k^2 + 2\mathrm{T} r_k d_k + \mathrm{T}^2 u_k} \end{aligned} \tag{4.19}$$

where T is the update time. The time update of the Doppler is done according to (4.20).

$$\begin{aligned} d_{k+1} &= \frac{1}{r_{k+1}} \left(x_{k+1} vel_{k+1}^x + y_{k+1} vel_{k+1}^y + z_{k+1} vel_{k+1}^z\right) \\ &= \frac{1}{r_{k+1}} ((x_k + \mathrm{T} vel_k^x) vel_k^x + (y_k + \mathrm{T} vel_k^y) vel_k^y + (z_k + \mathrm{T} vel_k^z) vel_k^z) \\ &= \frac{1}{r_{k+1}} (r_k d_k + \mathrm{T} u_k) \end{aligned} \tag{4.20}$$

where $r_{k+1} = \sqrt{r_k^2 + 2\mathrm{T} r_k d_k + \mathrm{T}^2 u_k}$. The time update of u_k is done according to (4.21).

$$\begin{aligned} u_{k+1} &= (vel_{k+1}^x)^2 + (vel_{k+1}^y)^2 + (vel_{k+1}^z)^2 \\ &= (vel_k^x)^2 + (vel_k^y)^2 + (vel_k^z)^2 = u_k \end{aligned} \tag{4.21}$$

Furthermore, it is assumed that $I_{k+1} = I_k$ since the target SNR can change in negligible amount in small time intervals.

In conclusion, the system dynamic model is represented as follows.

$$s_{k+1} = \begin{bmatrix} r_{k+1} \\ d_{k+1} \\ u_{k+1} \\ I_{k+1} \end{bmatrix} = \begin{bmatrix} \sqrt{r_k^2 + 2\mathrm{T} r_k d_k + \mathrm{T}^2 u_k} \\ (r_k d_k + \mathrm{T} u_k)/(\sqrt{r_k^2 + 2\mathrm{T} r_k d_k + \mathrm{T}^2 u_k}) \\ u_k \\ I_k \end{bmatrix} + \begin{bmatrix} v_k^r \\ v_k^d \\ v_k^u \\ v_k^I \end{bmatrix} \tag{4.22}$$

where v_k^r, v_k^d, v_k^u and v_k^I are process noises which are assumed to be white Gaussian.

4.3.3 Measurement Model

As mentioned earlier, radar provides a measurement, z_k, which contains $N_r N_d$ power measurements z_k^{aj}, where N_r is the number of range cells, N_d is the number of Doppler cells and z_k^{aj} is the power measurement at the $(a, j)^{th}$ range-Doppler cell at time step k. The power measurements z_k^{aj} is defined as follows, see [23].

$$z_k^{aj} = |z_{A,k}^{aj}|^2 \tag{4.23}$$

where $z_{A,k}^{aj}$ is the complex amplitude data of the received target signal at the $(a, j)^{th}$ range-Doppler cell and can be written as follows.

$$z_{A,k} = \tilde{A}_k h_{A,k}(s_k) + n_k \tag{4.24}$$

where

$$\tilde{A}_k = A_k e^{i\varphi_k}, \qquad \varphi_k \in (0, 2\pi) \tag{4.25}$$

$\tilde{A}_k$ is the complex amplitude of the signal that the target reflects, n_k is the complex Gaussian noise and $h_{A,k}(s_k)$ is the reflection form defined for each range-Doppler cell as follows.

$$h_{A,k}^{aj}(s_k) = e^{-\frac{(r_a - r_k)^2}{2R}L_r - \frac{(d_j - d_k)^2}{2D}L_d} \tag{4.26}$$

In this representation, R and D are constants which are related to the size of a range-Doppler cell and L_r, L_d are the constants of losses. r_a and d_j are the range and Doppler values of the $(a,j)^{th}$ cell, respectively.

As mentioned above, n_k is complex Gaussian noise and it can be written as follows.

$$n_k = n_{I,k} + in_{Q,k} \tag{4.27}$$

where n_{Ik} and n_{Qk} are in-phase and quadrature-phase components of n_k. They are zero mean white Gaussian noises with variance σ_n^2.

(4.23) can be rewritten by substituting (4.24) and (4.27) as follows.

$$z_k^{aj} = |\tilde{A}_k h_{A,k}^{aj}(s_k) + n_{I,k} + in_{Q,k}|^2 \tag{4.28}$$

It is assumed that these power measurements conditioned on the state vector, s_k, are exponentially distributed, see [33].

$$p(z_k^{aj}|s_k, m_k) \cong \begin{cases} p_{S+N}(z_k^{aj}|s_k) & , \ m_k = 1 \\ p_N(z_k^{aj}) & , \ m_k = 0 \end{cases} \tag{4.29}$$

where

$$p_{S+N}(z_k^{aj}|s_k) = \frac{1}{\mu_t^{aj}} \, e^{-\frac{1}{\mu_t^{aj}} z_k^{aj}} \tag{4.30}$$

and

$$p_N(z_k^{aj}) = \frac{1}{\mu_v^{aj}}\, e^{-\frac{1}{\mu_v^{aj}} z_k^{aj}} \tag{4.31}$$

$p_{S+N}(z_k^{aj}|s_k)$ is the pdf of target plus noise power conditioned on the state vector; whereas, $p_N(z_k^{aj})$ is the pdf of the power measurement when there is only measurement noise in the $(a,j)^{th}$ range-Doppler cell. Note that μ_t^{aj} is the expected power of $(a,j)^{th}$ range-Doppler cell when there is a target; whereas, μ_v^{aj} is the expected power when there is no target present. They can be represented as in (4.32) and (4.33).

$$\begin{aligned}
\mu_t^{aj} &= E\left[\left(z_k^{aj}\middle|s_k, m_k = 1\right)\right] \\
&= E\left[\left|A_k e^{i\varphi_k} h_{A,k}^{aj}(s_k) + n_{I,k} + i n_{Q,k}\right|^2\right] \\
&= E\left[\left(A_k h_{A,k}^{aj}(s_k)\, cos(\varphi_k) + n_{I,k}\right)^2 + \left(A_k h_{A,k}^{aj}(s_k)\, sin(\varphi_k) + n_{Q,k}\right)^2\right] \\
&= A_k^2\left(h_{A,k}^{aj}(s_k)\right)^2 + 2\sigma_n^2 \\
&= P_k h_{P,k}^{aj}(s_k) + 2\sigma_n^2
\end{aligned} \tag{4.32}$$

$$\begin{aligned}
\mu_v^{aj} &= E\left[\left(z_k^{aj}\middle|s_k, m_k = 0\right)\right] \\
&= E\left[\left|n_{I,k} + i n_{Q,k}\right|^2\right] = E\left[\left(n_{I,k}\right)^2 + \left(n_{Q,k}\right)^2\right] = 2\sigma_n^2
\end{aligned} \tag{4.33}$$

P_k refers to the target power. Target power generally affects more than one range-Doppler cell in the range-Doppler matrix. The power contribution of the target in each range-Doppler cell is taken into account by $h_{Pk}^{aj}(s_k)$ which can be written as follows.

$$h_{P,k}^{aj}(s_k) = \left(h_{A,k}^{aj}(s_k)\right)^2 = e^{-\frac{(r_a - r_k)^2}{R} L_r - \frac{(d_j - d_k)^2}{D} L_d} \tag{4.34}$$

As seen in (4.35), the likelihood function $p(z_k|s_k, m_k)$ can be written as the product of the density functions of each range-Doppler cell, $p(z_k^{aj}|s_k, m_k)$, since the received power in each range-Doppler cell is conditionally independent from each other.

$$p(z_k|s_k, m_k) = \prod_{aj} p(z_k^{aj}|s_k, m_k) = \begin{cases} \prod_{aj} p_{S+N}(z_k^{aj}|s_k) & , \ m_k = 1 \\ \prod_{aj} p_N(z_k^{aj}) & , \ m_k = 0 \end{cases} \tag{4.35}$$

A target only contributes to the cells in its vicinity. Let us denote the group of cells which are affected by target as $\mathbb{P}$. (4.35) can be rewritten as follows.

$$p(z_k|s_k, m_k) \cong \begin{cases} \prod_{aj \in \mathbb{P}} p_{S+N}(z_k^{aj}|s_k) \prod_{aj \notin \mathbb{P}} p_N(z_k^{aj}) & , \ m_k = 1 \\ \prod_{aj} p_N(z_k^{aj}) & , \ m_k = 0 \end{cases} \tag{4.36}$$

4.3.4 Steps of Algorithm 1

4.3.4.1 Initialization of the Particles

The particle filter is initialized by drawing the set of samples $\{s_0^i, m_0^i\}$ according to initial state and mode variable distributions as follows.

$$s_0^i \sim p(s_0), \qquad m_0^i \sim p(m_0) \tag{4.37}$$

Different initial state and mode variable distributions are used in the initialization part of the particle filters, [11], [27]. If there isn't any information about the initial distributions, then the initial mode variables, m_0^i, which are binary variables can be drawn from the uniform density. For each particle whose mode variable is $m_0^i = 1$, the state vector of the particle s_0^i can be drawn from the uniform distribution over the surveillance region. For each particle whose mode variable is $m_0^i = 0$, the state vector is undefined as mentioned earlier. For our problem, Algorithm 1 initializes two particle filters for the two targets: main platform and the spawned one.

4.3.4.2 Predictions of the Particles

After the initialization of the particles, the first step is to determine the transitions of the mode variables according to the Markov transition matrix by using Pseudocode 4.

Pseudocode 4 Mode Variable Transitions

$\left[\{m_k^i\}_{i=1}^N\right] = \text{Transitions of Mode Variables}\left[\{m_{k-1}^i\}_{i=1}^N, \Pi\right]$

- FOR $i = 1 : 2$
  - $a_i(0) = 0$
  - FOR $j = 1 : 2$
    - ❖ $a_i(j) = a_i(j-1) + \pi_{ij}$
  - END FOR
- END FOR
- FOR $d = 1 : N$
  - Draw: $u_d \sim U[0,1]$
  - Assign: $i = m_{k-1}^d$ and $v = 1$

- WHILE ($a_i(v) < u_d$)
  - ❖ $v = v + 1$
- END WHILE
- Assign: $m_k^d = v$

- END FOR

Predictions of the particles are done according to the mode variable transitions. There are two different groups of particles which differ from each other according to the way of prediction done: existing particles and newborn particles.

a) **<u>Newborn Particles:</u>** The particles whose transitions of modes are from $m_{k-1}^i = 0$ to $m_k^i = 1$ are called the '*newborn particles*'. In this thesis, the range, r_k^i, and the Doppler, d_k^i, are drawn from the uniform density over those regions of the range-Doppler matrix for which $z_k^{aj} > \gamma$, where γ is a predefined threshold. The state of target SNR, I_k^i, is chosen as the power (SNR value) of the cell which corresponds to the selected range and Doppler values. The value of the threshold is very important since the choice of the threshold value affects the performance of the particle filter. It should not be very low since the low value causes most of the particles to be in the places where there isn't any target and it should not be very high since this causes an increase in the probability of losing the meaningful information. The state of u_k^i is also drawn from the uniform density, $U[u_{min}, u_{max}]$, where u_{min}, u_{max} are the suitably chosen minimum and maximum values of the target state u_k, respectively.

Any a priori information about the states of the target can also be used for drawing the newborn particles.

b) Existing Particles: The particles whose mode variable transitions are from $m_{k-1}^i = 1$ to $m_k^i = 1$ are called the '*existing particles*'. The transitional prior, $p(s_k^i|s_{k-1}^i)$, is used for the existing particles although it is not the optimal one.

$$s_k^i \sim p(s_k|s_{k-1}^i) \tag{4.38}$$

The transitional prior can be obtained from the system dynamic model represented in (4.22).

Note that the target state vector is undefined for the particles whose mode variable is $m_k^i = 0$.

4.3.4.3 Measurement Update of the Particles

In the measurement update part of particle filters, the importance weights of the particles are updated. As mentioned earlier, if the transitional prior density, $p(s_k|s_{k-1}^i)$, is selected as the importance density, then the unnormalized importance weights of the particles are computed as follows.

$$\tilde{w}_k^i \propto w_{k-1}^i p\left(z_k|s_k^i, m_k^i\right) \tag{4.39}$$

The normalized weights, w_{k-1}^i, can be omitted since resampling is applied at the end of each time step. As mentioned earlier, all normalized weights of the particles have the same weight which is equal to $1/N$ at the end of resampling. Therefore, (4.39) can be rewritten as follows.

$$\tilde{w}_k^i \propto p\left(z_k|s_k^i, m_k^i\right) \tag{4.40}$$

$p\left(z_k|s_k^i, m_k^i\right)$ is derived in (4.36) which can be divided by $\prod_{aj} p_N(z_k^{aj})$ since the proportionality of the importance weights is actually important in the measurement update. Therefore, (4.36) can be rewritten as follows.

$$p\left(z_k \middle| s_k^i, m_k^i\right) \cong \begin{cases} \prod_{aj \in \mathbb{P}} l\left(z_k^{aj} \middle| s_k^i\right) & , \ m_k^i = 1 \\ 1 & , \ m_k^i = 0 \end{cases} \tag{4.41}$$

where

$$l\left(z_k^{aj} \middle| s_k^i\right) = \frac{p_{S+N}\left(z_k^{aj} \middle| s_k^i\right)}{p_N\left(z_k^{aj}\right)} \tag{4.42}$$

In conclusion, the unnormalized weights of the particles are calculated according to (4.41).

4.3.4.4 Extraction of the Main Platform's Power Contribution from the Range-Doppler Matrix

In order to prevent gathering of the weak target's particles in the vicinity of the main platform, the main platform's power contribution is extracted from the range-Doppler matrix. After that, this modified measurement is used for the measurement update of the particles which belong to the weak target. The expected mean of the state vector estimation at time step k given the state vector estimation at time step $k-1$, denoted as $E[s_k^{est}|s_{k-1}^{est}]$, is firstly calculated in order to determine the expected place of the main platform in the range-Doppler matrix. However, this may not be the exact place of the main platform due to the possible maneuvers or bad estimations at time step $k-1$, s_{k-1}^{est}. Therefore, the SNR values close to the value of $E[I_k^{est}|I_{k-1}^{est}]$ are searched in the vicinity of the values of $E[r_k^{est}|r_{k-1}^{est}]$ and $E[d_k^{est}|d_{k-1}^{est}]$ in the range-Doppler matrix to find the exact place of the main platform. After it is found, the powers of the group of cells which are probably affected from the main platform are replaced by noise power as shown in Figure 4.1.

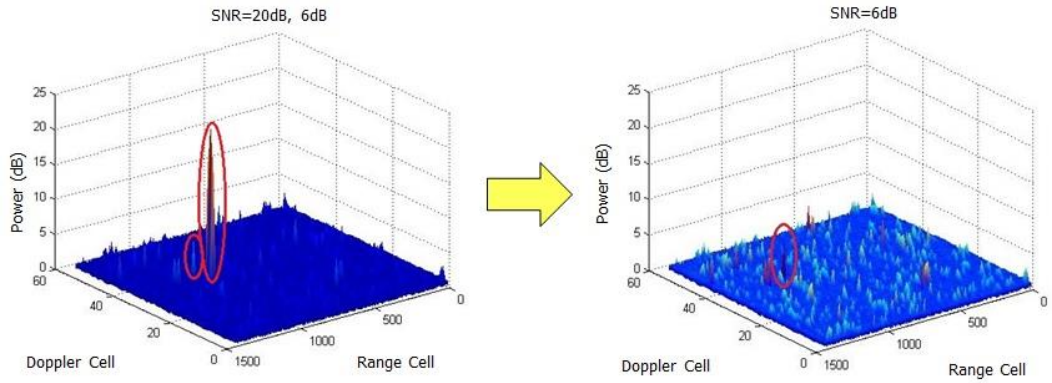


Figure 4.1. Extraction of the main platform's power contribution from the range-Doppler matrix

4.3.4.5 Normalization

The weights are normalized to approximate the true posterior pdf as follows.

$$w_k^i = \frac{\widetilde{w}_k^i}{\sum_{i=1}^{N} \widetilde{w}_k^i} \tag{4.43}$$

Therefore, the posterior pdf, $p(s_k|Z_k)$, can be expressed as follows.

$$p(s_k|Z_k) \approx \sum_{i=1}^{N} w_k^i \delta(s_k - s_k^i) \tag{4.44}$$

Note that the normalization of the weights is also done separately for each target.

4.3.4.6 Outputs

The probabilities of the targets' existences and the estimations of the state vectors are outputs of Algorithm 1. Outputs can be obtained in two ways: before resampling and after resampling. In this thesis, it is preferred to compute them before resampling since resampling causes information loss and this leads to decrease in accuracy, [25].

Target detection is performed by using the probability of target existence, P_k, which is represented in (4.45).

$$P_k = \sum_{i=1}^{N} m_k^i w_k^i \tag{4.45}$$

The declaration of the target existence is made if $P_k > \lambda_{TE}$, where λ_{TE} is a predefined threshold value.

The Minimum Mean Square Error (MMSE) estimate of a state vector, denoted as $s_{k|k}^{MMSE}$, is represented as follows.

$$s_{k|k}^{MMSE} \triangleq E[s_k|Z_k] = \int s_k \, p(s_k|Z_k) ds_k \tag{4.46}$$

In this work, (4.46) can be rewritten as follows.

$$s_{k|k}^{MMSE} = \frac{\sum_{i=1}^{N} s_k^i m_k^i \, w_k^i}{\sum_{i=1}^{N} m_k^i \, w_k^i} \tag{4.47}$$

Note that the probability of the target existence and the MMSE estimate of the state vector are computed separately for each target.

4.3.4.7 Resampling

Resampling is explained in details and the pseudocode of the resampling algorithm is given in Chapter 3.

Note that resampling is performed separately for each target's particles.

The pseudocode of the Algorithm 1 is given in Pseudocode 5. Note that (1) is used for the main platform and (2) is used for the weak target in Pseudocode 5.

Pseudocode 5 Algorithm 1

$$\left[\left\{s_k^{i(1)}\right\}_{i=1}^{N}, \left\{s_k^{i(2)}\right\}_{i=1}^{N}, s_{k|k}^{MMSE(1)}, s_{k|k}^{MMSE(2)}, P_k^{(1)}, P_k^{(2)}\right] = \text{TBD}\left[\left\{s_{k-1}^{i(1)}\right\}_{i=1}^{N}, \left\{s_{k-1}^{i(2)}\right\}_{i=1}^{N}, z_k\right]$$

- Mode variable transitions for each target using Pseudocode 4.

$$\left[\left\{m_k^{i(1)}\right\}_{i=1}^{N}\right] = \text{Transitions of Mode Variables}\left[\left\{m_{k-1}^{i(1)}\right\}_{i=1}^{N}, \Pi\right]$$

$$\left[\left\{m_k^{i(2)}\right\}_{i=1}^{N}\right] = \text{Transitions of Mode Variables}\left[\left\{m_{k-1}^{i(2)}\right\}_{i=1}^{N}, \Pi\right]$$

- FOR $i = 1 : N$
  - IF newborn particle
    - ❖ Draw $s_k^{i(1)}$ as mentioned in Section 4.3.4.2
  - ELSE IF existing particle
    - ❖ Draw $s_k^{i(1)} \sim p(s_k^{(1)} | s_{k-1}^{i(1)})$
  - END IF
  - Compute the importance weights $\widetilde{w}_k^{i(1)}$ using (4.41)
- END FOR
- Extract the main platform's power contribution from the range-Doppler matrix
- FOR $i = 1 : N$
  - IF newborn particle
    - ❖ Draw $s_k^{i(2)}$ as mentioned in Section 4.3.4.2
  - ELSE IF existing particle

❖ Draw $s_k^{i(2)} \sim p(s_k^{(2)} | s_{k-1}^{i(2)})$

- END IF

- Compute the importance weights $\widetilde{w}_k^{i(2)}$ using (4.41)

- END FOR

- FOR $i = 1 : N$

  - Normalize the weights

$$w_k^{i(1)} = \frac{\widetilde{w}_k^{i(1)}}{\sum_{i=1}^{N} \widetilde{w}_k^{i(1)}}$$

$$w_k^{i(2)} = \frac{\widetilde{w}_k^{i(2)}}{\sum_{i=1}^{N} \widetilde{w}_k^{i(2)}}$$

- END FOR

- Compute the outputs:

$$P_k^{(1)} = \sum_{i=1}^{N} m_k^{i(1)} w_k^{i(1)}$$

$$s_{k|k}^{MMSE(1)} = \frac{\sum_{i=1}^{N} s_k^{i(1)} m_k^{i(1)} w_k^{i(1)}}{\sum_{i=1}^{N} m_k^{i(1)} w_k^{i(1)}}$$

$$P_k^{(2)} = \sum_{i=1}^{N} m_k^{i(2)} w_k^{i(2)}$$

$$s_{k|k}^{MMSE(2)} = \frac{\sum_{i=1}^{N} s_k^{i(2)} m_k^{i(2)} w_k^{i(2)}}{\sum_{i=1}^{N} m_k^{i(2)} w_k^{i(2)}}$$

- Resample the particles separately for each target by using Pseudocode 2

- Declare the targets' existences if:

$$P_k{}^{(1)} > \lambda_{TE}$$

$$P_k{}^{(2)} > \lambda_{TE}$$

4.3.5 Algorithm 1 with Process Noise Identification

In this thesis, we focused on the case of the spawning targets, e.g. a missile fired from a fighter airplane. In real world, the spawned targets like missiles can highly maneuver throughout all periods of their motions, see [40]. In the particle filter framework, multiple dynamic model and the probability matrix constructed according to probabilities of the transitions between these models are able to handle this situation. However, the proposed process noise identification based method adopts one simple dynamic model during whole tracking process and does not need to know the possible multiple models and transition probabilities. Furthermore, for tracking of highly maneuvering targets like missiles by particle filters, the sample impoverishment is a very serious problem. In order to solve this problem, a new process noise identification method [1] is proposed for the classical target tracking. This method is adapted to the TBD framework in this study. The process noise identification is used for the state estimation of highly maneuvering spawned targets in the presence of non-stationary process noise with unknown parameters. In literature, there are different noise parameter estimation methods for particle filters. Most of them rely on the state augmentation technique, see [34] and [35]. Since the particle filters suffer from the increase in the state dimension, such approaches are difficult to apply to the systems with high dimensional state vector. In some methods, marginalization idea is performed with the assumption of a suitable prior distributions for the unknown noise parameters in particle filters, see [36] and [37]. However, such approaches are focused on the systems with slowly varying noise parameters which is not the case in this work. Therefore, the process noise identification approach proposed in [1] is used in this work.

4.3.5.1 Conceptual Solution of the Process Noise Identification in Particle Filter Framework

In the proposed method, a dynamic system is considered to be adapted to model the process noise. In this system, the noise vector, v_{k-1}, is the state vector of this

system. The main purpose is to construct the posterior pdf of the process noise, $p(v_{k-1}|Z_k)$, which is represented in (4.48).

$$p(v_{k-1}|Z_k) = \Upsilon p(z_k|v_{k-1})p(v_{k-1}|Z_{k-1}) \tag{4.48}$$

where

$$\Upsilon = \frac{1}{p(z_k|Z_{k-1})} \tag{4.49}$$

$p(z_k|Z_{k-1})$ is a normalizing constant. v_{k-1} is independent from the previous measurements, Z_{k-1}, since there is no information about it. Therefore, (4.48) can be rewritten as follows.

$$p(v_{k-1}|Z_k) = \Upsilon p(z_k|v_{k-1})p(v_{k-1}) \tag{4.50}$$

The process noise, v_{k-1}, is assumed to be uniformly distributed in the interval of $[-d, d]$ where d is the known process noise bound. Therefore, as seen in (4.51), the distribution of the process noise, $p(v_{k-1})$, can be represented by H particles which are drawn from $U[-d, d]$ where U denotes the uniform distribution.

$$p(v_{k-1}) \cong \frac{1}{H}\sum_{j=1}^{H} \delta(v_{k-1} - v_{k-1}^{j}) \tag{4.51}$$

(4.50) can be rewritten by substituting (4.51) as follows.

$$p(v_{k-1}|Z_k) = \frac{\Upsilon}{H}\sum_{j=1}^{H} p(z_k|v_{k-1}^{j})\delta(v_{k-1} - v_{k-1}^{j}) \tag{4.52}$$

Let's denote $p(z_k|v_{k-1}^{j})$ as ε_k^j for simplification. It corresponds to the weight of the jth process noise particle, v_{k-1}^{j}. The process noise particles are then resampled to eliminate the noise particles with low weights and generating copies of noise

particles with high weights. At the end of resampling, all noise particles have the same weight which is equal to $1/H$.

The weights of the particles, ε_k^j, can be obtained by expanding the likelihood function, $p(z_k|v_{k-1}^j)$, based on the resampled state particles at time step $k-1$, s_{k-1}^i.

$$p\left(z_k\middle|v_{k-1}^j\right) = \sum_{i=1}^{H} p(z_k|s_{k-1}^i, v_{k-1}^j)p(s_{k-1}^i|v_{k-1}^j) \tag{4.53}$$

where $p\left(s_{k-1}^i\middle|v_{k-1}^j\right) = p(s_{k-1}^i)$ since s_{k-1}^i and v_{k-1}^j are independent. Furthermore, $p\left(s_{k-1}^i\right) = 1/H$ because of the fact that all resampled particles have the same weight. (4.53) can be rewritten as follows.

$$p\left(z_k\middle|v_{k-1}^j\right) = \sum_{i=1}^{H} p(z_k|s_{k-1}^i, v_{k-1}^j)\frac{1}{H} \tag{4.54}$$

Let's define the intermediate particles denoted by $\mu_k^{i,j}$ in order to calculate $p(z_k|s_{k-1}^i, v_{k-1}^j)$.

$$\mu_k^{i,j} = f\left(s_{k-1}^i, v_{k-1}^j\right) \tag{4.55}$$

Based on the intermediate particles, $\mu_k^{i,j}$, $p(z_k|s_{k-1}^i, v_{k-1}^j)$ can be represented as in (4.56).

$$p\left(z_k\middle|s_{k-1}^i, v_{k-1}^j\right) = \sum_{p=1}^{H}\sum_{q=1}^{H}[p(z_k|\mu_k^{p,q}, s_{k-1}^i, v_{k-1}^j)p(\mu_k^{p,q}|s_{k-1}^i, v_{k-1}^j))] \tag{4.56}$$

Since s_{k-1}^i and v_{k-1}^j are known, $\mu_k^{p,q}$ is obtained as follows.

$$p\left(\mu_k^{p,q}|s_{k-1}^i, v_{k-1}^j\right) = \begin{cases} 1, & for\ p = i\ and\ q = j \\ 0, & for\ p \neq i\ or\ q \neq j \end{cases} \tag{4.57}$$

Therefore, (4.54) can be rewritten by substituting (4.56) and (4.57) as follows.

$$p(z_k|v_{k-1}^j) = \sum_{i=1}^{H} p(z_k|\mu_k^{i,j}) \frac{1}{H} \tag{4.58}$$

where $p(z_k|s_{k-1}^i, v_{k-1}^j) = p(z_k|\mu_k^{i,j})$.

After assigning the weights of the process noise particles and resampling these particles based on their weights, the procedures of the standard particle filter without process noise identification are then followed by obtaining the predicted particles $\{s_k^i, i = 1, \dots, H\}$ based on the resampled process noise samples $\{v_{k-1}^i, i = 1, \dots, H\}$ using the dynamic model represented in (4.1).

4.3.5.2 Simplification of the Proposed Method in TBD framework

In the proposed method, there is much more computational burden compared to a standard particle filter with H particles since $H\ x\ H$ intermediate particles are evaluated via (4.55) at each time step. It is observed that the particles belonged to the spawned target, $\{s_{k-1}^{i(2)}, i = 1, \dots, H\}$, gather around the state estimates of the spawned target when the existence of the spawned target is declared, e.g. $P_{k-1}^{(2)} > \lambda_{TE}$. They focus in a very small area. Therefore, the particles $\{s_{k-1}^i, i = 1, \dots, H\}$ represented in (4.55) are replaced by the state estimate of the spawned target at time step $k-1$, $s_{k-1}^{est(2)}$, when $P_{k-1}^{(2)} > \lambda_{TE}$. Therefore, the intermediate particles are then evaluated as follows.

$$\mu_k^q = f\left(s_{k-1}^{est(2)}, v_{k-1}^q\right) \quad if \ \ P_{k-1}^{(2)} > \lambda_{TE} \tag{4.59}$$

The likelihood function represented in (4.58) can be rewritten as follows.

$$p(z_k|v_{k-1}^j) = \sum_{q=1}^{H} p(z_k|\mu_k^q) p(\mu_k^q|v_{k-1}^j) \tag{4.60}$$

Therefore, when $P_{k-1}^{(2)} > \lambda_{TE}$**,** the simplification method mentioned above gives the possibility of getting rid of the computational burden generated from the intermediate particles represented in (4.55).

Note that when $P_{k-1}^{(2)} < \lambda_{TE}$, the procedures of the standard particle filter without process noise identification are then followed since the process noise identification without simplification increases the computational burden too much. Furthermore, the process noise identification method uses only the existing particles whose mode transitions of the mode variables are from $m_{k-1}^{i(2)} = 1$ to $m_k^{i(2)} = 1$. The newborn particles can't be used with the process noise identification since the dynamic model is not used when drawing these particles.

The pseudocode of the Algorithm 1 with process noise identification is given in Pseudocode 6.

Pseudocode 6 Algorithm 1 with Process Noise Identification

$$\left[\left\{s_k^{i(1)}\right\}_{i=1}^{N}, \left\{s_k^{i(2)}\right\}_{i=1}^{N}, s_{k|k}^{MMSE(1)}, s_{k|k}^{MMSE(2)}, P_k^{(1)}, P_k^{(2)}\right]$$
$$= \text{TBD}\left[\left\{s_{k-1}^{i(1)}\right\}_{i=1}^{N}, \left\{s_{k-1}^{i(2)}\right\}_{i=1}^{N}, z_k\right]$$

- Mode variable transitions for each target using Pseudocode 4

$$\left[\left\{m_k^{i(1)}\right\}_{i=1}^{N}\right] = \text{Transitions of Mode Variables}\left[\left\{m_{k-1}^{i(1)}\right\}_{i=1}^{N}, \Pi\right]$$
$$\left[\left\{m_k^{i(2)}\right\}_{i=1}^{N}\right] = \text{Transitions of Mode Variables}\left[\left\{m_{k-1}^{i(2)}\right\}_{i=1}^{N}, \Pi\right]$$

- IF $P_{k-1}^{(2)} > \lambda_{TE}$
  - Determine H which is the number of the particles whose mode transitions are from $m_{k-1}^{i(2)} = 1$ to $m_k^{i(2)} = 1$.
  - Draw the process noise particles $\{v_{k-1}^{j}, j = 1, \dots, H\}$ from the

uniform distribution, $U[-d, d]$.

- Determine the intermediate particles, μ_k^q, by using (4.59).
- Compute the weights of the process noise particles, $\{\varepsilon_k^j, j = 1, \dots, H\}$, by using (4.62).
- Resample the process noise particles by using Pseudocode 2.

- END IF
- FOR $i = 1 : N$
  - IF newborn particle
    - ❖ Draw $s_k^{i(1)}$ as mentioned in Section 4.3.4.2
  - ELSE IF existing particle
    - ❖ Draw $s_k^{i(1)} \sim p(s_k^{(1)} | s_{k-1}^{i(1)})$
  - END IF
  - Compute the importance weights $\widetilde{w}_k^{i(1)}$ using (4.41)
- END FOR
- Extract the main platform's power contribution from the range-Doppler matrix
- FOR $i = 1 : N$
  - IF newborn particle
    - ❖ Draw $s_k^{i(2)}$ as mentioned in Section 4.3.4.2
  - ELSE IF existing particle
    - ❖ IF $P_{k-1}^{(2)} > \lambda_{TE}$
      - ➢ Draw $s_k^{i(2)} \sim p(s_k^{(2)} | s_{k-1}^{i(2)})$, but use the resampled process noises $\{v_{k-1}^i, \ i = 1, \dots, H\}$ rather than

$[v_k^r\ v_k^d\ v_k^u\ v_k^I]^{\mathrm{T}}$ when drawing $s_k^{i(2)}$.

- ❖ ELSE IF $P_{k-1}^{(2)} < \lambda_{TE}$
  - ➢ Draw $s_k^{i(2)} \sim p(s_k^{(2)}|s_{k-1}^{i(2)})$
- ❖ END IF

- END IF
- Compute the importance weights $\widetilde{w}_k^{i(2)}$ using (4.41)

- END FOR
- FOR $i = 1 : N$
  - Normalize the weights

$$w_k^{i(1)} = \frac{\widetilde{w}_k^{i(1)}}{\sum_{i=1}^{N} \widetilde{w}_k^{i(1)}}$$

$$w_k^{i(2)} = \frac{\widetilde{w}_k^{i(2)}}{\sum_{i=1}^{N} \widetilde{w}_k^{i(2)}}$$

- END FOR
- Compute the outputs:

$$P_k^{(1)} = \sum_{i=1}^{N} m_k^{i(1)} w_k^{i(1)}$$

$$s_{k|k}^{MMSE(1)} = \frac{\sum_{i=1}^{N} s_k^{i(1)} m_k^{i(1)} w_k^{i(1)}}{\sum_{i=1}^{N} m_k^{i(1)} w_k^{i(1)}}$$

$$P_k^{(2)} = \sum_{i=1}^{N} m_k^{i(2)} w_k^{i(2)}$$

$$s_{k|k}^{MMSE(2)} = \frac{\sum_{i=1}^{N} s_k^{i(2)} m_k^{i(2)} w_k^{i(2)}}{\sum_{i=1}^{N} m_k^{i(2)} w_k^{i(2)}}$$

- Resample the particles separately for each target by using Pseudocode 2

- Declare the targets' existences if:

$$P_k{}^{(1)} > \lambda_{TE}$$
$$P_k{}^{(2)} > \lambda_{TE}$$

4.4 Algorithm 2

The main differences between Algorithm 1 and Algorithm 2 are the hypotheses which the mode variable contains, the system dynamic and the measurement models. The TBD approach of Algorithm 2 is based on the work in [12]. In contrast to that work, Algorithm 2 uses a novel reduced order system dynamic model. Furthermore, it is not restricted to the case of constant and known SNR as in [12], e.g. target SNR values are also estimated. In addition to these modifications, the process noise identification approach used in Algorithm 1 is also used in this algorithm.

This algorithm is explained in detail in the following sections.

4.4.1 System Dynamic Model

In this algorithm, the system dynamic model given in (4.1) is used with the state vector which is composed of the state vectors of both targets as represented in (4.62).

$$s_k = \begin{pmatrix} s_k^{(1)} \\ s_k^{(2)} \end{pmatrix} \tag{4.61}$$

The mode variable, m_k, represents the number of targets in the surveillance region:

- $m_k = 0$: There is no target present.
- $m_k = 1$: There is one target present.
- $m_k = 2$: There are two targets present.

The transition matrix, Π, becomes in the form of

$$\Pi = \begin{pmatrix} \pi_{00} & \pi_{01} & \pi_{02} \\ \pi_{10} & \pi_{11} & \pi_{12} \\ \pi_{20} & \pi_{21} & \pi_{22} \end{pmatrix} \tag{4.62}$$

Note that $s_k^{(1)} = s_k^{(2)}$ when the mode variable is $m_k = 1$, since in this way, the state vector of the weak target is initially the same as the state vector of the main platform when the weak target is spawned from the main platform. For the sake of completeness, the state vector is undefined when the mode variable is $m_k = 0$.

4.4.2 Measurement Model

A slightly modified version of the measurement model of Algorithm 1 is used in this algorithm.

The complex amplitude data, $z_{A,k}$, represented in (4.24) is modified for the case of two targets as follows.

$$z_{A,k} = \tilde{A}_k^{(1)} h_{A,k}^{(1)}(s_k) + \tilde{A}_k^{(2)} h_{A,k}^{(2)}(s_k) + n_k \tag{4.63}$$

The power measurements per cell, z_k^{ij}, can then be written as follows.

$$z_k^{aj} = |z_{A,k}^{aj}|^2 = |\tilde{A}_k^{(1)} h_{A,k}^{aj(1)}(s_k) + \tilde{A}_k^{(2)} h_{A,k}^{aj(2)}(s_k) + n_{I,k} + i n_{Q,k}|^2 \tag{4.64}$$

As mentioned earlier, it is assumed that these power measurements conditioned on the state vector are exponentially distributed.

$$p(z_k^{aj}|s_k, m_k) \cong \begin{cases} p_N(z_k^{aj}) & , \ m_k = 0 \\ p_{S+N}^{(1)}(z_k^{aj}|s_k) & , \ m_k = 1 \\ p_{S+N}^{(2)}(z_k^{aj}|s_k) & , \ m_k = 2 \end{cases} \tag{4.65}$$

where

$$p_{S+N}^{(2)}(z_k^{aj}|s_k) = \frac{1}{\mu_t^{aj*}}\, e^{-\frac{1}{\mu_t^{aj*}} z_k^{aj}} \tag{4.66}$$

and

$$\mu_t^{aj*} = E\left[z_k^{aj}|s_k, m_k = 2\right]$$

$$= E\left[\left|A_k^{(1)} e^{i\varphi_k} h_{A,k}^{aj(1)}(s_k) + A_k^{(2)} e^{i\varphi_k} h_{A,k}^{aj(2)}(s_k) + n_{I,k} + i n_{Q,k}\right|^2\right]$$

$$= E[\left(A_k^{(1)} h_{A,k}^{aj(1)}(s_k)\cos(\varphi_k) + A_k^{(2)} h_{A,k}^{aj(2)}(s_k)\cos(\varphi_k) + n_{I,k}\right)^2 +$$

$$\left(A_k^{(1)} h_{A,k}^{aj(1)}(s_k)\sin(\varphi_k) + A_k^{(2)} h_{A,k}^{aj(2)}(s_k)\sin(\varphi_k) + n_{Q,k}\right)^2]$$

$$= A_k^{(1)}\left(h_{A,k}^{aj(1)}(s_k)\right)^2 + A_k^{(2)}\left(h_{A,k}^{aj(2)}(s_k)\right)^2 + 2\sigma_n^2$$

$$= P_k^{(1)} h_{P,k}^{aj(1)}(s_k) + P_k^{(2)} h_{P,k}^{aj(2)}(s_k) + 2\sigma_n^2 \tag{4.67}$$

As mentioned earlier, $p_N(z_k^{aj})$ and $p_{S+N}^{(1)}(z_k^{aj}|s_k)$ are represented as in (4.68) and (4.69), respectively.

$$p_{S+N}^{(1)}(z_k^{aj}|s_k) = \frac{1}{\mu_t^{aj}}\, e^{-\frac{1}{\mu_t^{aj}} z_k^{aj}} \tag{4.68}$$

$$p_N(z_k^{aj}) = \frac{1}{\mu_v^{aj}}\, e^{-\frac{1}{\mu_v^{aj}} z_k^{aj}} \tag{4.69}$$

where

$$\mu_t^{aj} = P_k h_{P,k}^{aj}(s_k) + 2\sigma_n^2 \tag{4.70}$$

and

$$\mu_v^{aj} = 2\sigma_n^2 \tag{4.71}$$

As mentioned earlier, the likelihood function, $p(z_k|s_k, m_k)$, can be written as the product of the density functions of each range-Doppler cell, $p(z_k^{aj}|s_k, m_k)$, since the received power in each range-Doppler cell is independent from each other.

$$p(z_k|s_k, m_k) = \begin{cases} \prod_{aj} p_N(z_k^{aj}) & , \ m_k = 0 \\ \prod_{aj} p_{S+N}{}^{(1)}(z_k^{aj}|s_k) & , \ m_k = 1 \\ \prod_{aj} p_{S+N}{}^{(2)}(z_k^{aj}|s_k) & , \ m_k = 2 \end{cases} \tag{4.72}$$

Targets only contribute to the cells in their vicinity. Let us denote the group of cells which are affected by targets as $\mathbb{P}$. (4.72) can then be rewritten as follows.

$$p(z_k|s_k, m_k) = \begin{cases} \prod_{aj} p_N(z_k^{aj}) & , \ m_k = 0 \\ \prod_{aj\in\mathbb{P}} p_{S+N}{}^{(1)}(z_k^{aj}|s_k) \prod_{aj\notin\mathbb{P}} p_N(z_k^{aj}) & , \ m_k = 1 \\ \prod_{aj\in\mathbb{P}} p_{S+N}{}^{(2)}(z_k^{aj}|s_k) \prod_{aj\notin\mathbb{P}} p_N(z_k^{aj}) & , \ m_k = 2 \end{cases} \tag{4.73}$$

4.4.3 Steps of Algorithm 2

4.4.3.1 Initialization of the Particles

Initialization of the particles is made according to the same principals mentioned in the first algorithm. At first, initial mode variables, m_0^i, where $m_k \in \{0, 1, 2\}$, are drawn uniformly. After drawing the initial mode variables, initial state vectors of the particles are drawn according to the initial mode variables:

- For each particle whose mode variable is $m_0^i = 0$, the state vector is undefined.
- For each particle whose mode variable is $m_0^i = 1$, the state vector is $s_0^i = [s_0^{i(1)}\ s_0^{i(2)}]^{\mathrm{T}}$ such that $s_0^{i(1)}$ is drawn from the uniform distribution over the surveillance region and $s_0^{i(2)} = s_0^{i(1)}$.
- For each particle whose mode variable is $m_0^i = 2$, the state vector is $s_0^i = [s_0^{i(1)}\ s_0^{i(2)}]^{\mathrm{T}}$ such that $s_0^{i(1)}$ and $s_0^{i(2)}$ are drawn from the uniform distribution over the surveillance region.

4.4.3.2 Predictions of the Particles

After the initialization of the particles, the first step is to determine the transitions of the mode variables according to the Markov transition matrix by using Pseudocode 8. Predictions of the particles are made according to these mode variable transitions. There are five different cases:

- **The transition from $\boldsymbol{m_{k-1}^i = 0}$ to $\boldsymbol{m_k^i = 1}$:** In this case, $s_k^{i(1)}$ is drawn as a newborn particle and $s_k^{i(2)} = s_k^{i(1)}$.
- **The transition from $\boldsymbol{m_{k-1}^i = 1}$ to $\boldsymbol{m_k^i = 1}$:** In this case, $s_k^{i(1)}$ is drawn as an existing particle and $s_k^{i(2)} = s_k^{i(1)}$.
- **The transition from $\boldsymbol{m_{k-1}^i = 1}$ to $\boldsymbol{m_k^i = 2}$:** In this case, $s_k^{i(1)}$ is drawn as an existing particle and $s_k^{i(2)}$ is drawn as a newborn particle.
- **The transition from $\boldsymbol{m_{k-1}^i = 2}$ to $\boldsymbol{m_k^i = 1}$:** In this case, $s_k^{i(1)}$ is drawn as an existing particle and $s_k^{i(2)} = s_k^{i(1)}$.
- **The transition from $\boldsymbol{m_{k-1}^i = 2}$ to $\boldsymbol{m_k^i = 2}$:** In this case, $s_k^{i(1)}$ and $s_k^{i(2)}$ are drawn as existing particles.

Note that the transition from $m_{k-1}^i = 0$ to $m_k^i = 2$ is not included in the list above since there is no direct transition from the case of no target present to two target

present in the scenarios used in this thesis. However, this is not crucial for the proposed algorithms.

4.4.3.3 Measurement Update of the Particles

As mentioned earlier, the importance weights of the particles are updated in the measurement update stage of the particle filter as in (4.40).

$p\left(z_k \middle| s_k^i, m_k^i\right)$ is derived in (4.73) which can be divided by $\prod_{aj} p_N(z_k^{aj})$ since the proportionality of the importance weights is actually important in the measurement update. Therefore, (4.73) can be rewritten as follows.

$$p\left(z_k \middle| s_k^i, m_k^i\right) \cong \begin{cases} 1 & , \ m_k = 0 \\ \prod_{aj \in \mathbb{P}} l^{(1)}(z_k^{aj} \middle| s_k^i) & , \ m_k = 1 \\ \prod_{aj \in \mathbb{P}} l^{(2)}(z_k^{aj} \middle| s_k^i) & , \ m_k = 2 \end{cases} \tag{4.74}$$

where

$$l^{(1)}(z_k^{aj} \middle| s_k^i) = \frac{{p_{S+N}}^{(1)}(z_k^{aj} | s_k)}{p_N(z_k^{aj})} \tag{4.75}$$

and

$$l^{(2)}(z_k^{aj} \middle| s_k^i) = \frac{{p_{S+N}}^{(2)}(z_k^{aj} | s_k)}{p_N(z_k^{aj})} \tag{4.76}$$

The weights are then normalized to approximate the true posterior pdf as in (4.43).

4.4.3.4 Outputs

In this algorithm, it is also preferred to compute the outputs before resampling. The outputs of Algorithm 2 are the probabilities of the targets' existences and the estimations of the state vector.

Target detections are performed by using the probabilities of targets' existences, denoted by $P_k^{(1)}$ and $P_k^{(2)}$, respectively. $P_k^{(1)}$ is equal to the sum of the weights of the particles whose mode variable is $m_k^i = 1$.

$$P_k^{(1)} = \sum\nolimits_{i\ such\ that\ m_k^i=1} w_k^i \tag{4.77}$$

$P_k^{(2)}$ is equal to the sum of the weights of the particles whose mode variable is $m_k^i = 2$ as follows.

$$P_k^{(2)} = \sum\nolimits_{i\ such\ that\ m_k^i=2} w_k^i \tag{4.78}$$

The declarations of the target existences are made if the probabilities of the target existences are higher than a predefined threshold value, λ_{TE}.

The MMSE estimate of the state vector is represented as follows.

$$s_{k|k}^{MMSE} = [s_{k|k}^{MMSE(1)}\ s_{k|k}^{MMSE(2)}]^{\mathrm{T}} \tag{4.79}$$

where

$$s_{k|k}^{MMSE(1)} = \frac{\sum_{i\ such\ that\ m_k^i=1\ or\ 2} s_k^{i(1)} w_k^i}{\sum_{i\ such\ that\ m_k^i=1\ or\ 2} w_k^i} \tag{4.80}$$

and

$$s_{k|k}^{MMSE(2)} = \frac{\sum_{i\ such\ that\ m_k^i=2} s_k^{i(2)} w_k^i}{\sum_{i\ such\ that\ m_k^i=2} w_k^i} \tag{4.81}$$

After computing the outputs, particles are resampled as explained in Section 4.3.4.7.

The pseudocode of Algorithm 2 is given in Pseudocode 7.

Pseudocode 7 Algorithm 2

$\left[\{s_k^i\}_{i=1}^N, s_{k|k}^{MMSE(1)}, s_{k|k}^{MMSE(2)}, P_k^{(1)}, P_k^{(2)}\right] = \text{TBD}\left[\{s_{k-1}^i\}_{i=1}^N, z_k\right]$

- Mode variable transitions for each target using Pseudocode 4

  $\left[\{m_k^i\}_{i=1}^N\right] = \text{Transitions of Mode Variables}\left[\{m_{k-1}^i\}_{i=1}^N, \Pi\right]$

- FOR $i = 1 : N$
  - IF $m_{k-1}^i = 0$ to $m_k^i = 1$:
    - ❖ Draw $s_k^{i(1)}$ as a newborn particle and assign $s_k^{i(2)} = s_k^{i(1)}$
  - ELSE IF $m_{k-1}^i = 1$ to $m_k^i = 1$*:*
    - ❖ Draw $s_k^{i(1)}$ as an existing particle and assign $s_k^{i(2)} = s_k^{i(1)}$
  - ELSE IF $m_{k-1}^i = 1$ to $m_k^i = 2$*:*
    - ❖ Draw $s_k^{i(1)}$ as an existing particle and $s_k^{i(2)}$ as a newborn particle
  - ELSE IF $m_{k-1}^i = 2$ to $m_k^i = 1$*:*
    - ❖ Draw $s_k^{i(1)}$ as an existing particle and assign $s_k^{i(2)} = s_k^{i(1)}$
  - ELSE IF $m_{k-1}^i = 2$ to $m_k^i = 2$*:*

❖ Draw $s_k^{i(1)}$ and $s_k^{i(2)}$ as existing particles

- END IF
- Compute the importance weights $\widetilde{w}_k^i$ by using (4.74)

- END FOR
- FOR $i = 1 : N$
  - Normalize the weights

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

- END FOR
- Compute the outputs

$$P_k^{(1)} = \sum\nolimits_{i\ such\ that\ m_k^i=1\ or\ 2} w_k^i$$

$$s_{k|k}^{MMSE(1)} = \frac{\sum_{i\ such\ that\ m_k^i=1\ or\ 2} s_k^{i(1)} w_k^i}{\sum_{i\ such\ that\ m_k^i=1\ or\ 2} w_k^i}$$

$$P_k^{(2)} = \sum\nolimits_{i\ such\ that\ m_k^i=2} w_k^i$$

$$s_{k|k}^{MMSE(2)} = \frac{\sum_{i\ such\ that\ m_k^i=2} s_k^{i(2)} w_k^i}{\sum_{i\ such\ that\ m_k^i=2} w_k^i}$$

- Resample the particles using Pseudocode 2
- Declare targets' existences if:

$$P_k{}^{(1)} > \lambda_{TE}$$

$$P_k{}^{(2)} > \lambda_{TE}$$

4.4.4 Algorithm 2 with Process Noise Identification

The process noise identification approach used in Algorithm 2 is the same as the approach used in Algorithm 1. However, there are some slight differences in the implementation of the algorithms.

As mentioned earlier, the main purpose of the process noise identification is to track and detect the highly maneuvering spawned target. In Algorithm 1, each target has its own particles so that the process noise identification method can only be applied for the spawned target. However, it is not the case for the Algorithm 2 since the state vector used in the system dynamic model is composed of the state vectors of both targets, i.e. $s_k^i = [s_k^{i(1)}\ s_k^{i(2)}]^T$. Therefore, we have to apply the process noise identification approach for both targets so that the method uses only the existing particles whose transitions of the mode variables are from $m_{k-1}^i = 2$ to $m_k^i = 2$. Furthermore, in order to be able to apply the simplification mentioned in Section 4.3.5.2, the probability of the two targets' presence must be $P_{k-1}^{(2)} > \lambda_{TE}$.

The pseudocode of Algorithm 2 with process noise identification is given in Pseudocode 8.

Pseudocode 8 Algorithm 2 with Process Noise Identification

$$\left[\{s_k^i\}_{i=1}^N, s_{k|k}^{MMSE(1)}, s_{k|k}^{MMSE(2)}, P_k^{(1)}, P_k^{(2)}\right] = \text{TBD}\left[\{s_{k-1}^i\}_{i=1}^N, z_k\right]$$

- Mode variable transitions for each target using Pseudocode 4

$$\left[\{m_k^i\}_{i=1}^N\right] = \text{Transitions of Mode Variables}\left[\{m_{k-1}^i\}_{i=1}^N, \Pi\right]$$

- IF $P_{k-1}^{(2)} > \lambda_{TE}$
  - Determine H which corresponds to the number of the particles whose mode variable transitions are from $m_{k-1}^i = 2$ to $m_k^i = 2$.

- Draw the process noise particles $\{v_{k-1}^{j}, j = 1, \dots, H\}$ from the uniform distribution, $U[-d, d]$.
- Determine the intermediate particles: $\mu_k^q = f\left(s_{k-1}^{est}, v_{k-1}^q\right)$
- Compute the weights of the process noise particles, $\{\varepsilon_k^j, j = 1, \dots, H\}$, by using (4.62).
- Resample the process noise particles by using Pseudocode 2.

- END IF
- FOR $i = 1 : N$
  - IF $m_{k-1}^i = 0$ to $m_k^i = 1$:
    - ❖ Draw $s_k^{i(1)}$ as a newborn particle and assign $s_k^{i(2)} = s_k^{i(1)}$
  - ELSE IF $m_{k-1}^i = 1$ to $m_k^i = 1$:
    - ❖ Draw $s_k^{i(1)}$ as an existing particle and assign $s_k^{i(2)} = s_k^{i(1)}$
  - ELSE IF $m_{k-1}^i = 1$ to $m_k^i = 2$:
    - ❖ Draw $s_k^{i(1)}$ as an existing particle and $s_k^{i(2)}$ as a newborn particle
  - ELSE IF $m_{k-1}^i = 2$ to $m_k^i = 1$:
    - ❖ Draw $s_k^{i(1)}$ as an existing particle and assign $s_k^{i(2)} = s_k^{i(1)}$
  - ELSE IF $m_{k-1}^i = 2$ to $m_k^i = 2$:
    - ❖ IF $P_{k-1}^{(2)} > \lambda_{TE}$
      - ➢ Draw $s_k^{i(1)}$ and $s_k^{i(2)}$ as existing particles, but use the resampled process noises $\{v_{k-1}^i,\ i = 1, \dots, H\}$ rather than $[v_k^{r(1)}\ v_k^{d(1)}\ v_k^{u(1)}\ v_k^{I(1)}\ v_k^{r(2)}\ v_k^{d(2)}\ v_k^{u(2)}\ v_k^{I(2)}]^{\mathrm{T}}$ when drawing s_k^i.

- ❖ ELSE IF $P_{k-1}^{(2)} < \lambda_{TE}$
  - ➢ Draw $s_k^{i(1)}$ and $s_k^{i(2)}$ as existing particles.
- ❖ END IF

- END IF
- Compute the importance weights $\widetilde{w}_k^i$ by using (4.74)

- END FOR
- FOR $i = 1 : N$
  - Normalize the weights

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

- END FOR
- Compute the outputs:

$$P_k^{(1)} = \sum\nolimits_{i\ such\ that\ m_k^i=1\ or\ 2} w_k^i$$

$$s_{k|k}^{MMSE(1)} = \frac{\sum_{i\ such\ that\ m_k^i=1\ or\ 2} s_k^{i(1)} w_k^i}{\sum_{i\ such\ that\ m_k^i=1\ or\ 2} w_k^i}$$

$$P_k^{(2)} = \sum\nolimits_{i\ such\ that\ m_k^i=2} w_k^i$$

$$s_{k|k}^{MMSE(2)} = \frac{\sum_{i\ such\ that\ m_k^i=2} s_k^{i(2)} w_k^i}{\sum_{i\ such\ that\ m_k^i=2} w_k^i}$$

- Resample the particles by using Pseudocode 2
- Declare the targets' existences if:

$$P_k{}^{(1)} > \lambda_{TE}$$

$$P_k{}^{(2)} > \lambda_{TE}$$

CHAPTER 5

RESULTS AND SIMULATIONS

5.1 Radar Simulator

A MATLAB based radar simulator developed in [38] is used for raw data generation. It generates a range-Doppler matrix based on the radar, targets and the environment related parameters for each run. The simulator given in [38] is modified to generate range-Doppler matrices for given targets' trajectories in a given time period. Different scenarios such as different number of targets, different SNR values of targets and different clutter spaces can be generated by changing the parameters. Moreover, the radar simulator [38] has a capability of generating target reflections according to different Swerling models [39]. Swerling-0 and Swerling-1 types of targets are used in the radar simulator to evaluate the algorithms used in this thesis.

5.2 Scenario and Parameters

In this section, scenarios used to evaluate the performances of the proposed TBD algorithms are explained in detail. There are four different scenarios. First scenario is used for the confirmation of the algorithms. In the second scenario, the spawned target maneuvers at some time intervals. This scenario is used for the analysis of the algorithms' performances for maneuvering targets. In the third scenario, the spawned target maneuvers so highly that the limit of the algorithms' performances is reached. By using this scenario, it is shown that the process noise identification

method increases the performances of the algorithms. In all scenarios except the fourth scenario, Swerling-1 type of targets are used. In the fourth scenario, Swerling-0 type of targets which move very slowly are used in order to see the detection times independently from the changes in the SNR values of the targets. Moreover, weak clutters are used in all of these scenarios.

- **First Scenario:** The first scenario begins with no target in the surveillance region. The main platform appears at time $t = 6\,s$ with the position of $24\,km$ from the radar and goes towards the radar with the constant velocity of $200\,ms^{-1}$. At time $t = 21\,s$, the weak target is spawned from the main platform and goes towards the radar. Initially, it moves with a constant acceleration of $40\,ms^{-2}$ in the first three time steps. At time $t = 24\,s$, it starts to move with a constant velocity of $320\,ms^{-1}$. The targets continue their motions until the scenario ends at time $t = 40\,s$. Figure 5.1 shows the trajectories of the targets in the first scenario. Note that the radar is at the origin (0, 0, 0).
- **Second Scenario:** The only difference between the first scenario and the second scenario is that the spawned target maneuvers with $+3g$ between $t = 28\,s$ and $t = 32\,s$, $-3g$ between $t = 33\,s$ and $t = 37\,s$. After the spawned target maneuvers, it continues to move with constant velocity. Figure 5.2 shows the trajectories of the targets in the second scenario.

- **Third Scenario:** The main platform appears at time $t = 6\,s$ with the position of $24.0674\,km$ from the radar. At time $t = 16\,s$, the weak target is spawned from the main platform and goes towards the radar. Initially, it moves with a constant acceleration of $40\,ms^{-2}$ in the first three time steps. At time $t = 19\,s$, it starts to move with a constant velocity of $320\,ms^{-1}$. The spawned target maneuvers with $-7g$ between $t = 28\,s$ and $t = 32\,s$, $+7g$ between $t = 33\,s$ and $t = 37\,s$. The targets continue their motions until the scenario ends at time $t = 40\,s$. Figure 5.3 shows the trajectories of the targets in the third scenario.

- **Fourth Scenario:** The fourth scenario is almost same as the first scenario. One of the difference between them is that Swerling-0 type of targets are used in the fourth scenario in order to see the detection times independently from the fluctuations in the SNR values of targets. The other difference is that the targets move very slowly in the fourth scenario. The main platform goes towards the radar with the constant velocity of $10\ ms^{-1}$. At time $t = 21\ s$, the weak target is spawned from the main platform and goes towards the radar. Initially, it moves with a constant acceleration of $10\ ms^{-2}$ in the first three time steps. At time $t = 24\ s$, it starts to move with a constant velocity of $40\ ms^{-1}$. In all scenarios except the fourth scenario, since the targets go towards the radar with high velocities, the mean SNR values of targets increase significantly. As we want to see the detection times independently from the increase in the mean SNR values of the targets, the targets move with very low velocities; so that, the mean SNR values increase very slightly during the fourth scenario. Figure 5.4 shows the trajectories of the targets in the fourth scenario.

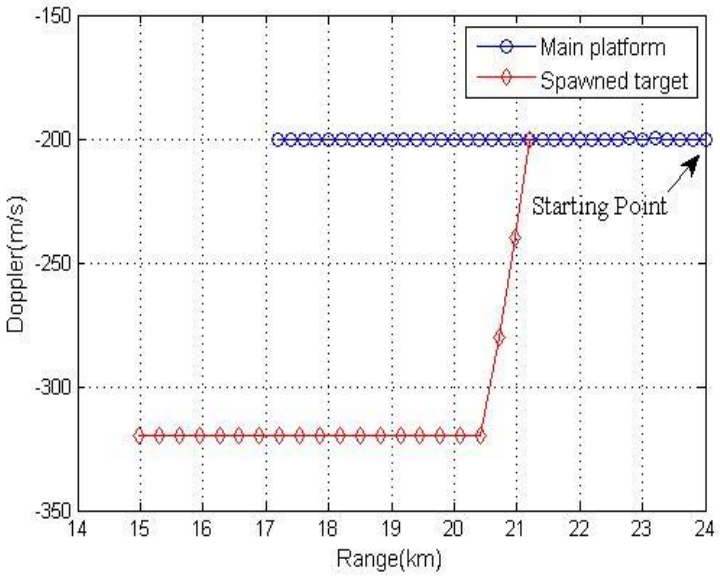


Figure 5.1. The first scenario

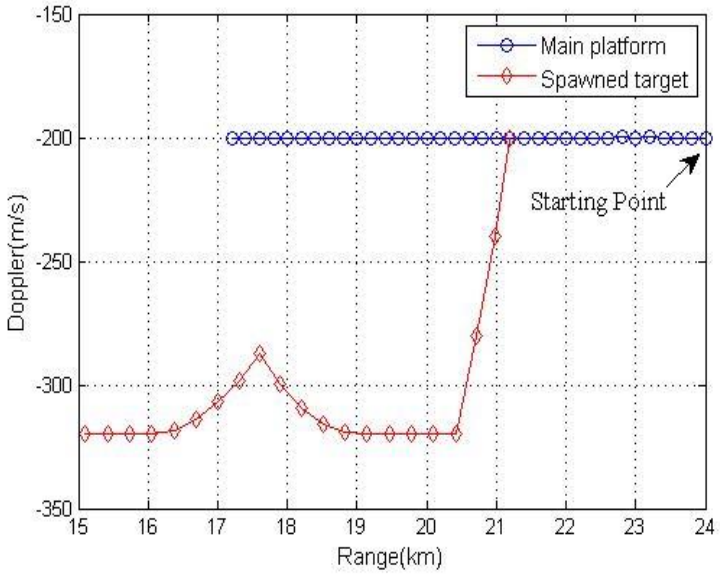


Figure 5.2. The second scenario

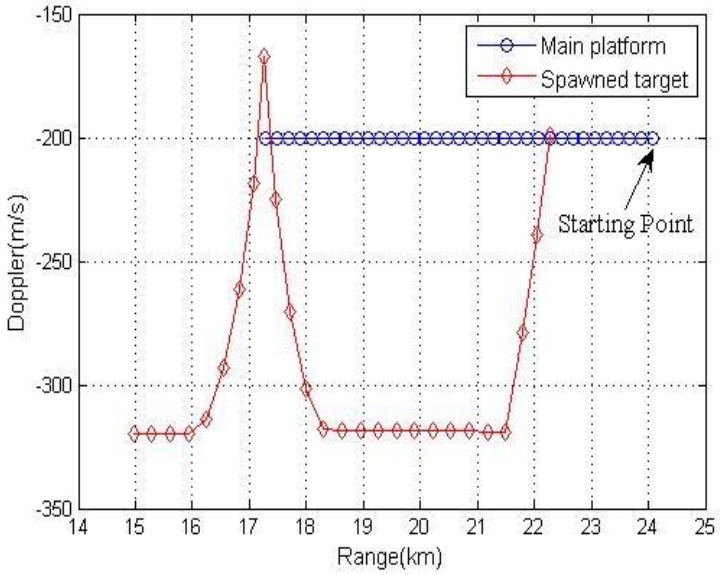


Figure 5.3. The third scenario

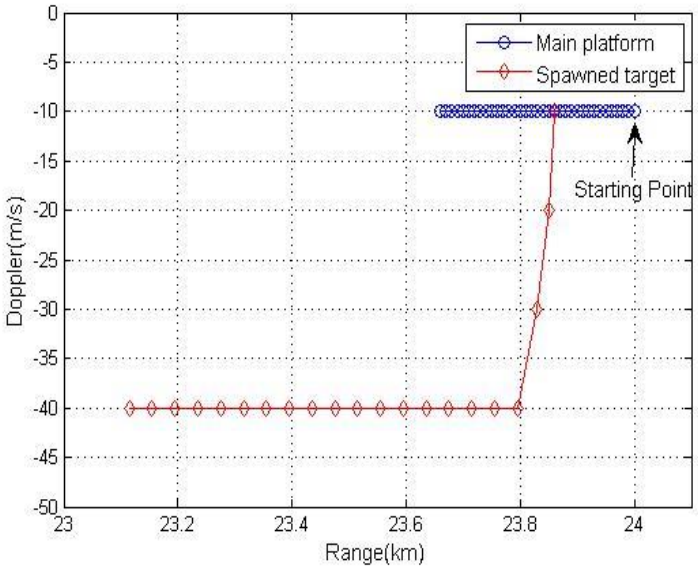


Figure 5.4. The fourth scenario

The measurement space is constructed by the range interval of $[0, 28.4]\ km$ and the Doppler interval of $[0, 570]\ ms^{-1}$. It is divided into N_rxN_d cells, where $N_r = 1420$ and $N_d = 57$. The constants related to the size of range and Doppler cells, R and D, are equal to $20\ m$ and $10\ ms^{-1}$, respectively. The constant of losses are $L_r = 0.23$ and $L_d = 0.2$.

As mentioned earlier, the expected intervals of the target states on the targets' appearance constitute the surveillance region of the targets as given in Table 5.1 and Table 5.2. Table 5.1 is given for all scenarios except the fourth scenario; whereas, Table 5.2 is given for the fourth scenario.

Note that $^{(1)}$ refers to the states which belong to the main platform; whereas, $^{(2)}$ refers to the states which belong to the spawned target. Initially, particles are uniformly drawn from the surveillance region given in Table 5.1.

Table 5.1. The surveillance region for all scenarios except the fourth scenario

Parameter	**Min Value**	**Max Value**	**Unit**
Range$^{(1)}$	14500	24500	m
Doppler$^{(1)}$	150	250	m/s
u$^{(1)}$	22500	62500	m^2/s^2
Intensity$^{(1)}$	12	30	dB
Range$^{(2)}$	14500	24500	m
Doppler$^{(2)}$	150	380	m/s
u$^{(2)}$	22500	144400	m^2/s^2
Intensity$^{(2)}$	3	15	dB

Table 5.2. The surveillance region for the fourth scenario

Parameter	**Min Value**	**Max Value**	**Unit**
Range$^{(1)}$	23000	24500	m
Doppler$^{(1)}$	0	20	m/s
u$^{(1)}$	0	400	m^2/s^2
Intensity$^{(1)}$	12	30	dB
Range$^{(2)}$	23000	24500	m
Doppler$^{(2)}$	0	50	m/s
u$^{(2)}$	0	2500	m^2/s^2
Intensity$^{(2)}$	3	15	dB

The transition probability matrix is given as follows for Algorithm 1.

$$\Pi = \begin{pmatrix} 1 - P_b & P_b \\ P_d & 1 - P_d \end{pmatrix} \tag{5.1}$$

where $P_b = 0.05$ and $P_d = 0.05$. It is same for both targets. Since there are three hypotheses that mode variable contains, the transition probability matrix is a $3x3$ matrix for Algorithm 2 as follows.

$$\Pi = \begin{pmatrix} 0.9 & 0.1 & 0 \\ 0.1 & 0.8 & 0.1 \\ 0 & 0.1 & 0.9 \end{pmatrix} \tag{5.2}$$

As it can be seen in (5.2), the spawned target can't appear before the main platform appears in the surveillance region. In addition to this, the main platform can't disappear before the weak target disappears in the surveillance region. However, this condition is not necessary for Algorithm 2.

The predefined threshold for the target existences is $\lambda_{TE} = 0.6$ and the sampling time is $\mathrm{T} = 1\,s$ for both algorithms. As mentioned earlier, the process noise represented in (4.22) is assumed to be white Gaussian. In all scenarios except the fourth scenario, the process noise for the main platform is selected as follows.

$$v_k^{(1)} \sim N\left(\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} (5/3)^2 & 0 & 0 & 0 \\ 0 & (10/3)^2 & 0 & 0 \\ 0 & 0 & (3264/3)^2 & 0 \\ 0 & 0 & 0 & (5/3)^2 \end{pmatrix}\right) \tag{5.3}$$

In the fourth scenario, the process noise for the main platform is selected as in (5.4).

$$v_k^{(1)} \sim N\left(\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} (1/3)^2 & 0 & 0 & 0 \\ 0 & (2/3)^2 & 0 & 0 \\ 0 & 0 & (44/3)^2 & 0 \\ 0 & 0 & 0 & (1.5/3)^2 \end{pmatrix}\right) \tag{5.4}$$

In all scenarios except the fourth scenario, the process noise for the spawned target is selected as follows.

$$v_k^{(2)} \sim N\left(\begin{pmatrix}0\\0\\0\\0\end{pmatrix}, \begin{pmatrix}(22/3)^2 & 0 & 0 & 0\\ 0 & (44/3)^2 & 0 & 0\\ 0 & 0 & (25000/3)^2 & 0\\ 0 & 0 & 0 & (4/3)^2\end{pmatrix}\right) \quad (5.5)$$

In the fourth scenario, the process noise for the spawned target is selected as in (5.6).

$$v_k^{(2)} \sim N\left(\begin{pmatrix}0\\0\\0\\0\end{pmatrix}, \begin{pmatrix}(6/3)^2 & 0 & 0 & 0\\ 0 & (12/3)^2 & 0 & 0\\ 0 & 0 & (700/3)^2 & 0\\ 0 & 0 & 0 & (1/3)^2\end{pmatrix}\right) \quad (5.6)$$

As mentioned earlier, the process noise particles are drawn from the uniform distribution of $U[-d, d]$ when using the process noise identification method. Table 5.3 shows the intervals which the process noises are drawn from.

Table 5.3. The intervals from which the process noises are drawn when using the process noise identification method

Parameter	**Min Value**	**Max Value**	**Unit**
$v_k^{r(1)}$	-20	20	m
$v_k^{d(1)}$	-10	10	m/s
$v_k^{u(1)}$	-5000	5000	m^2/s^2
$v_k^{I(1)}$	-1.5	1.5	dB
$v_k^{r(2)}$	-90	90	m
$v_k^{d(2)}$	-65	65	m/s
$v_k^{u(2)}$	-20000	20000	m^2/s^2
$v_k^{I(2)}$	-1.5	1.5	dB

5.3 Simulation Results

The scenarios mentioned in Section 5.2 are used to confirm the algorithms and analyze their performances. In these scenarios, SNR values of each target fluctuate around the mean SNR according to the Swerling-1 model. Note that the mean SNR increases in time since targets get closer to the radar regardless of SNR fluctuations originated from the Swerling-1 model.

5.3.1 Simulation Results of Algorithm 1

5.3.1.1 Simulation Results for the First Scenario

The performance of Algorithm 1 is analyzed for different initial SNR values of the spawned target as 10 dB, 8 dB, 6 dB, 4 dB. In all simulations, the initial SNR of the main platform is 18 dB.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

5.3.1.1.1 Initial SNR of the Spawned Target: 10 dB

The analysis of the performance of Algorithm 1 is started with the 10dB initial SNR of the spawned target.

Figure 5.5 shows the positions of the particles at time $t = 4, 16, 22, 23, 25$ and $32\ s$.

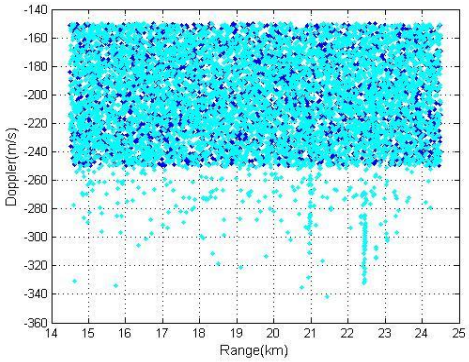
The particles of the main platform are indicated by blue dots and yellow ‘o’ indicates the actual position of the main platform when the main platform is present. The particles of the weak target are indicated by cyan dots. Blue ‘*’ indicates the actual position of the weak target when the weak target is present. As it can be seen in Figure 5.5, the particles gather around the targets after a finite number of time steps following the target appearance in the surveillance region.

This is what we expected for the successful detection and tracking in the particle filter based algorithms.

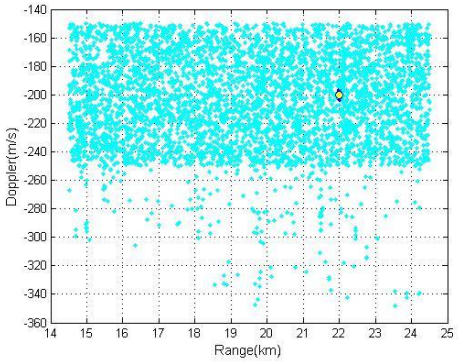
For the sake of clarity, we want to mention again that the main platform appears at time $t = 6\,s$; whereas, the spawned target appears at time $t = 21\,s$ in the surveillance region.

The probability of existence of the main platform is shown in Figure 5.6. The left axis of Figure 5.6 shows the probability of existence of the main platform for different number of particles; whereas, the right axis is for the SNR of the main platform. The SNR of the main platform is represented by black '*'.

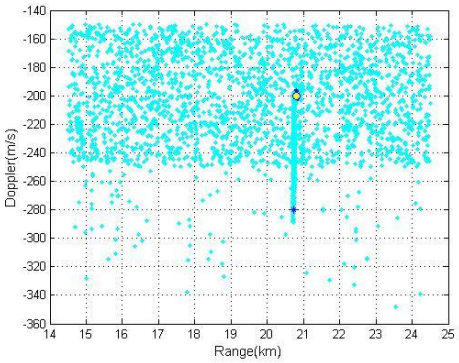
As it can be seen in Figure 5.6, the main platform is detected almost immediately as it appears in the surveillance region, e.g. there is a rapid increase of the probability of existence after the appearance of the main platform because of its high SNR value (18 dB). Furthermore, all selected numbers of particles give the same result that indicates that the number of particles can be reduced further below 500.



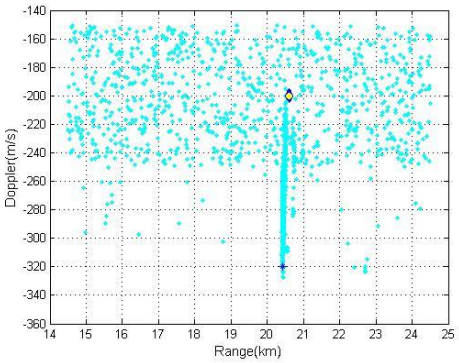
a) $t = 4\ s$



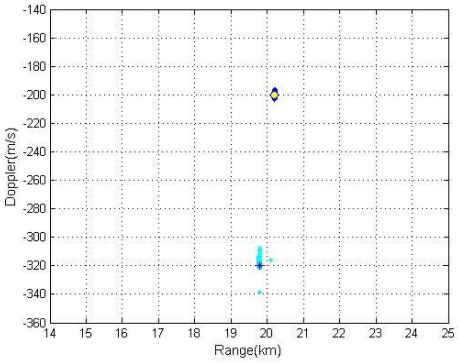
b) $t = 16\ s$



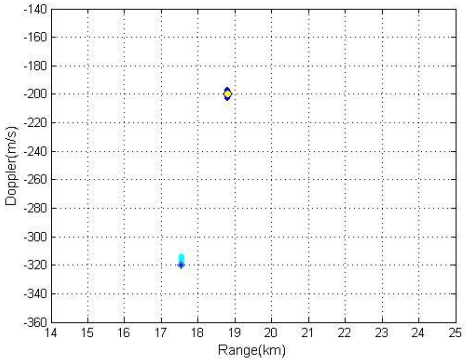
c) $t = 22\ s$



d) $t = 23\ s$



c) $t = 25\ s$



d) $t = 32\ s$

Figure 5.5. Positions of the particles at different time steps

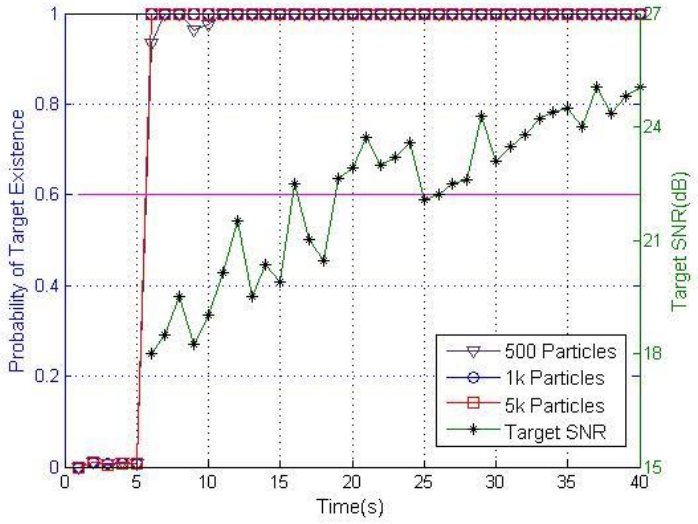


Figure 5.6. The probability of existence of the main platform and its SNR value (First scenario, 10 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

Figure 5.7 plays the same role as Figure 5.6 for the spawned target. On the left axis, it shows the probability of existence of the spawned target for different number of particles; whereas, on the right axis, the SNR of the spawned target is shown. The SNR value is represented by black '*'. As it can be seen in Figure 5.7, the existence of the weak target is declared in 2 seconds for 5k particles, 3 seconds for 1k particles and 4 seconds for 500 particles after its appearance. Therefore, it can be concluded that in the transient part about the declaration of the target existence, the detection performance of Algorithm 1 increases with the increase in the number of particles.

Figure 5.8 and Figure 5.9 show the range versus Doppler estimates together with the actual positions of the main platform and the spawned target, respectively. As it can be seen in Figure 5.7 and Figure 5.9 especially for 500 particles, the more accurate estimates are made after the detection of the spawned target. Furthermore, as it can be seen in Figure 5.8 and Figure 5.9, there is an estimation error which behaves like an offset after the target is detected. This is actually a resolution based

estimation error. The algorithms can't realize any difference between the true range-Doppler value and the estimated range-Doppler value due to resolution of the range-Doppler cells.

The RMS range errors for the main platform and the spawned target are given in Figure 5.10 and Figure 5.12, respectively.

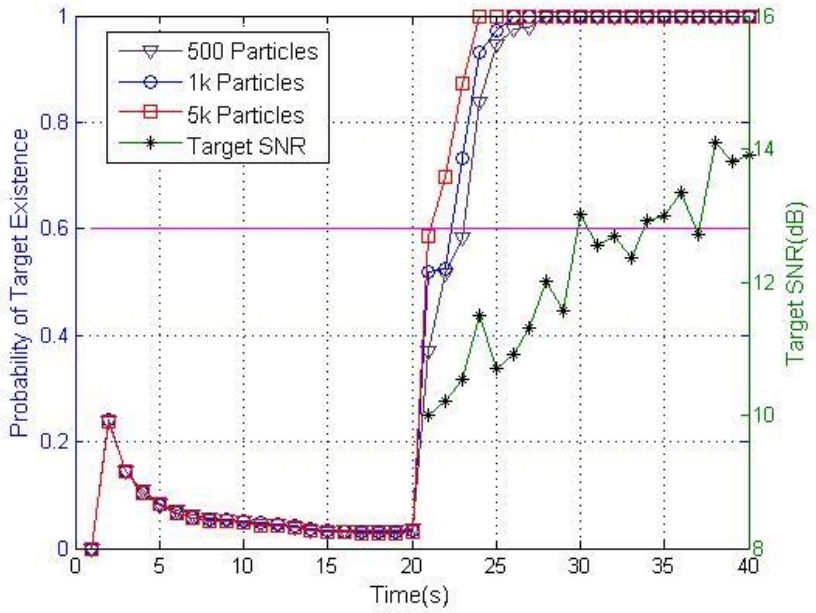


Figure 5.7. The probability of existence of the spawned target and its SNR value (First scenario, 10 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

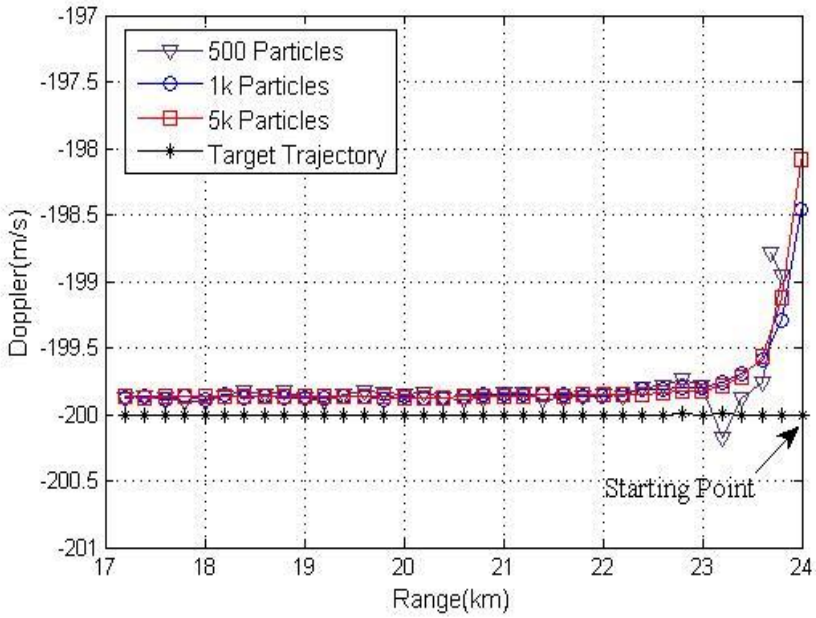


Figure 5.8. Range versus Doppler estimates of the main platform and its trajectory (First scenario, 10 dB initial SNR of the spawned target)

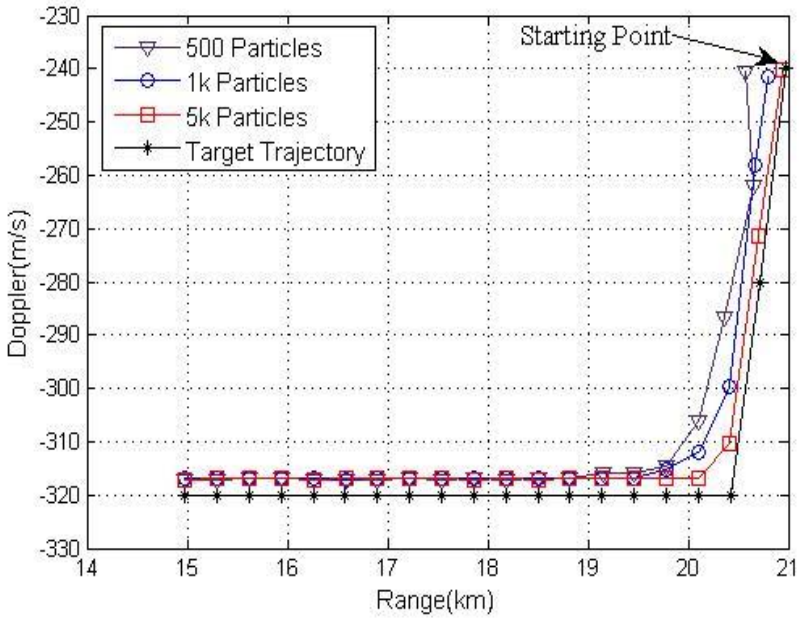


Figure 5.9. Range versus Doppler estimates for the spawned target and its trajectory (First scenario, 10 dB initial SNR of the spawned target)

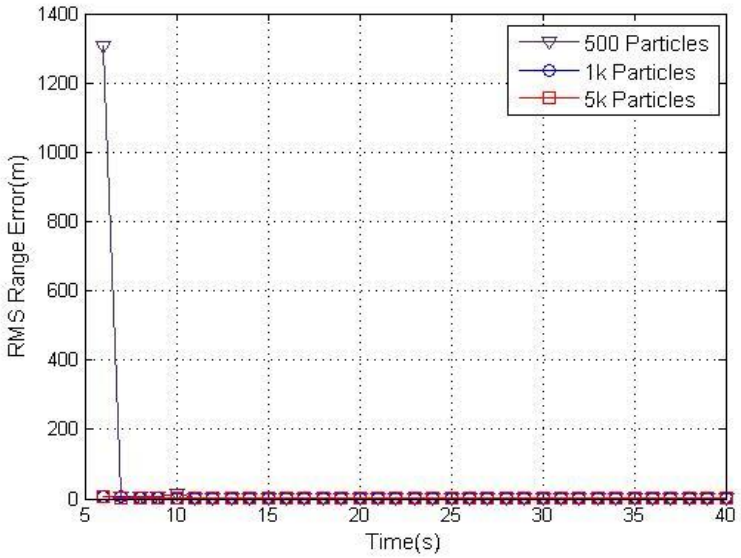


Figure 5.10. The RMS range error for the main platform (First scenario, 10 dB initial SNR of the spawned target)

In figure below, Figure 5.10 is zoomed in to display the small errors in steady state.

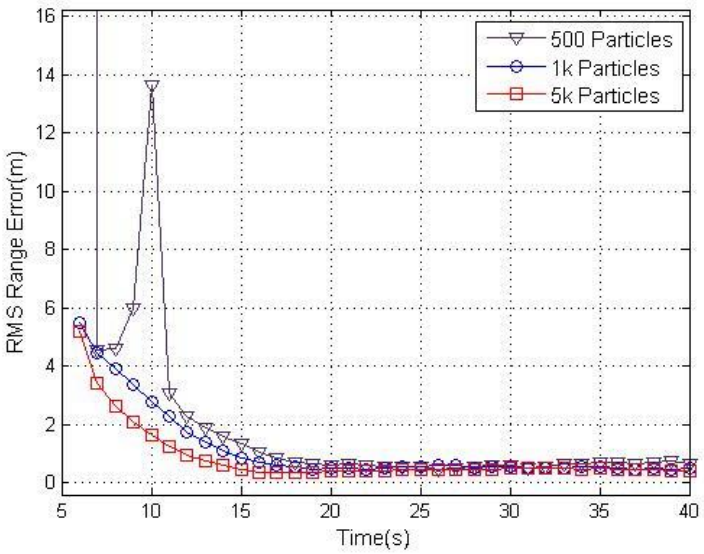


Figure 5.11. The RMS range error for the main platform (First scenario, 10 dB initial SNR of the spawned target)

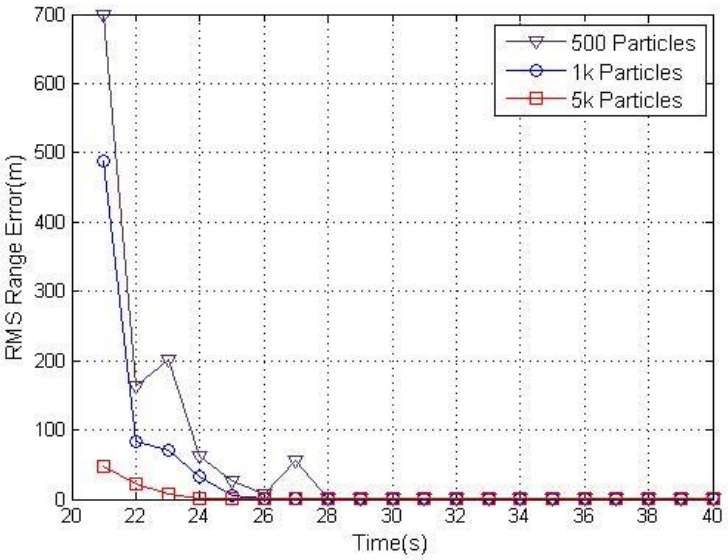


Figure 5.12. The RMS range error for the spawned target (First scenario, 10 dB initial SNR of the spawned target)

The RMS Doppler errors for the main platform and the spawned target are given in Figure 5.13 and Figure 5.14, respectively.

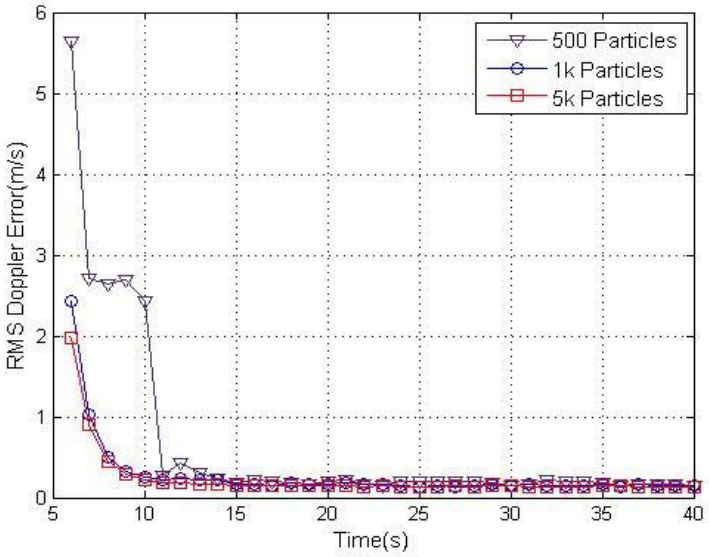


Figure 5.13. The RMS Doppler error for the main platform (First scenario, 10 dB initial SNR of the spawned target)

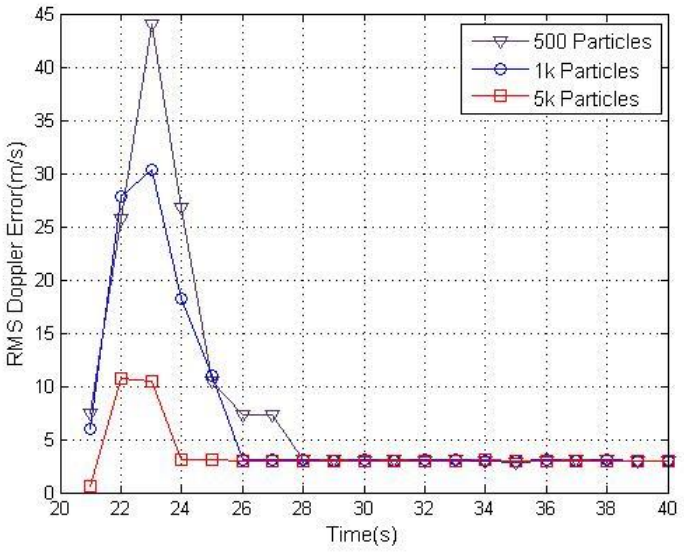


Figure 5.14. The RMS Doppler error for the spawned target (First scenario, 10 dB initial SNR of the spawned target)

As it can be seen in Figure 5.14, the RMS Doppler error for the spawned target initially increases since the spawned target starts its motion with constant acceleration. However, it decreases after the spawned target begins to move with constant velocity which is consistent with the dynamic model used in this thesis.

The effective number of particles gives an idea about the performance of the particle filters. The performance of the particle filters increase with the increase in the effective number of particles. The ratio N_{eff}/N for the main platform and the spawned target can be seen in Figure 5.15 and Figure 5.16, respectively. Note that N_{eff} is the effective number of particles and N is the number of particles.

As it can be seen in Figure 5.15 and Figure 5.16, the ratio N_{eff}/N is relatively high before the targets appear in the surveillance region since the mode variables of almost all of the particles equal to 0 and the unnormalized weights of these particles equal to 1 according to (4.41). Furthermore, there is a rapid decrease in the effective number of particles in a finite number of time steps after the appearance of the targets in the surveillance region. This is because as the targets

appear in the surveillance region, some newborn particles appear around the targets and there is a big difference between the weights of these newborn particles and the particles whose mode variable is $m_k^i = 0$.

After the existence of the spawned target is declared, the ratio of the effective number of particles for the spawned target becomes around 12% which is low. The process noise for the spawned target, $q_k^{r(2)}$, $q_k^{d(2)}$ and $q_k^{u(2)}$, have high variances since the spawned target moves with constant acceleration at first three time steps although the constant velocity model is used as system dynamic model. Therefore, the existing particles for the spawned target spread so much that most of the particles situate in the positions where the spawned target doesn't exist. This increases the differences in the weights of the existing particles so that the effective number of particles decreases.

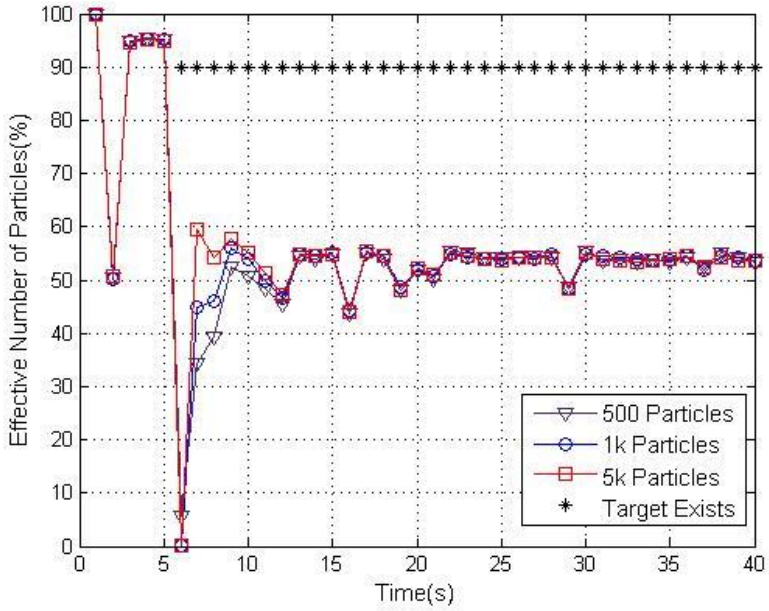


Figure 5.15. N_{eff}/N for the main platform (First scenario, 10 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

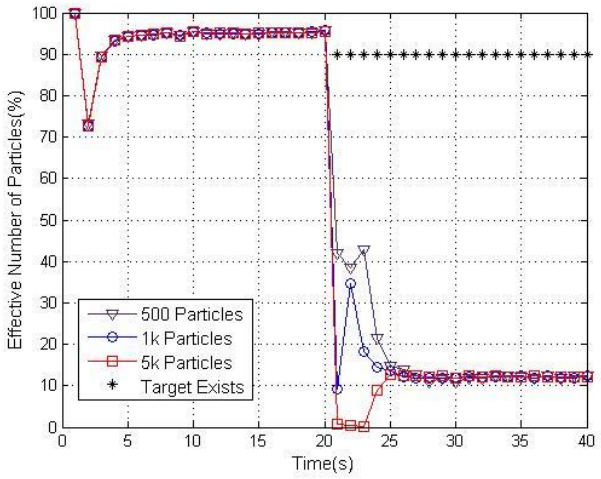


Figure 5.16. N_{eff}/N for the spawned target (First scenario, 10 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

As mentioned earlier, the algorithms proposed in this thesis are able to estimate the SNR values of the targets. The SNR estimates for the main platform and the spawned target with the actual SNR values of the targets can be seen in Figure 5.17 and Figure 5.18, respectively.

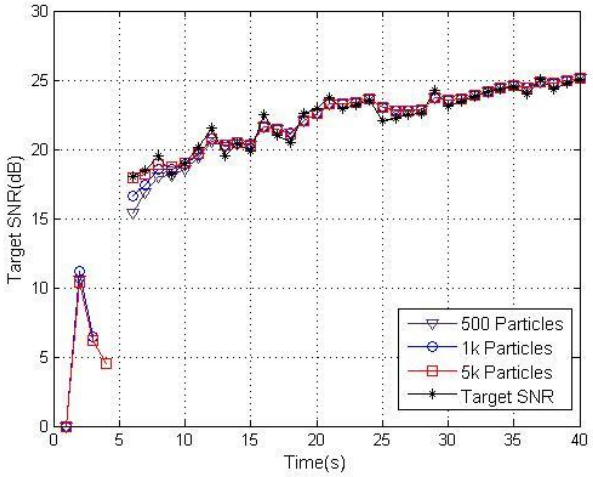


Figure 5.17. The SNR estimates for the main platform (First scenario, 10 dB initial SNR of the spawned target)

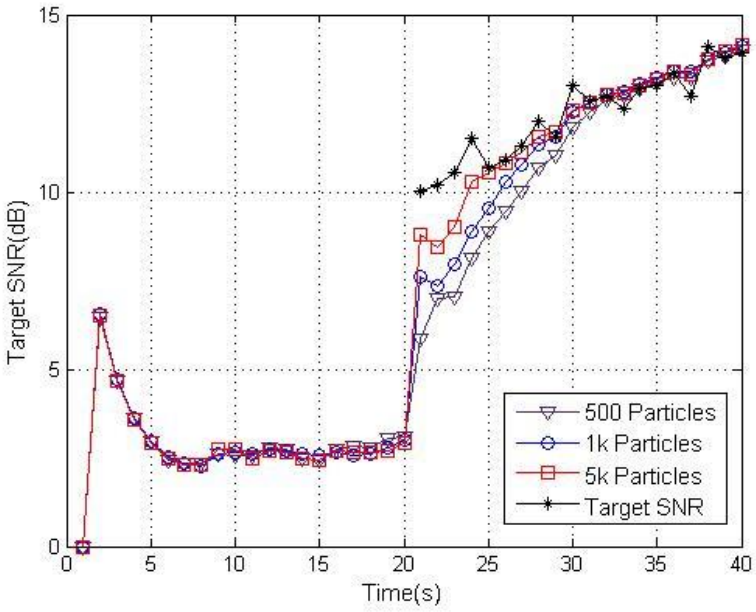


Figure 5.18. The SNR estimates for the spawned target (First scenario, 10 dB initial SNR of the spawned target)

5.3.1.1.2 Initial SNR of the Spawned Target: 8 dB

The initial SNR of the spawned target is decreased to the value of 8dB to determine the minimum SNR value of the weak target at which Algorithm 1 is successful at the detection and the tracking of both targets. The results about the main platform aren't shown in this section again since the decreasing SNR of the spawned target doesn't affect the performance of tracking and detection of the main platform. Note that 35 Monte Carlo simulations are performed to obtain each result given below.

On the left axis, Figure 5.19 shows the probability of the spawned target's existence for different number of particles; whereas, on the right axis, the SNR of the spawned target is represented by black '*'. The delay in detection of the spawned target increases compared to the case of 10 dB initial target SNR as it is expected. As it can be seen in Figure 5.19, the existence of the spawned target is

declared in 5 seconds for 500, 1k and 5k particles. At this time, the target SNR is at most slightly above 9 dB which is lower compared to the first experiment.

Figure 5.20 shows the range versus Doppler estimates and the actual positions of the spawned target.

The RMS range and Doppler errors for the spawned target are given in Figure 5.21 and Figure 5.22, respectively.

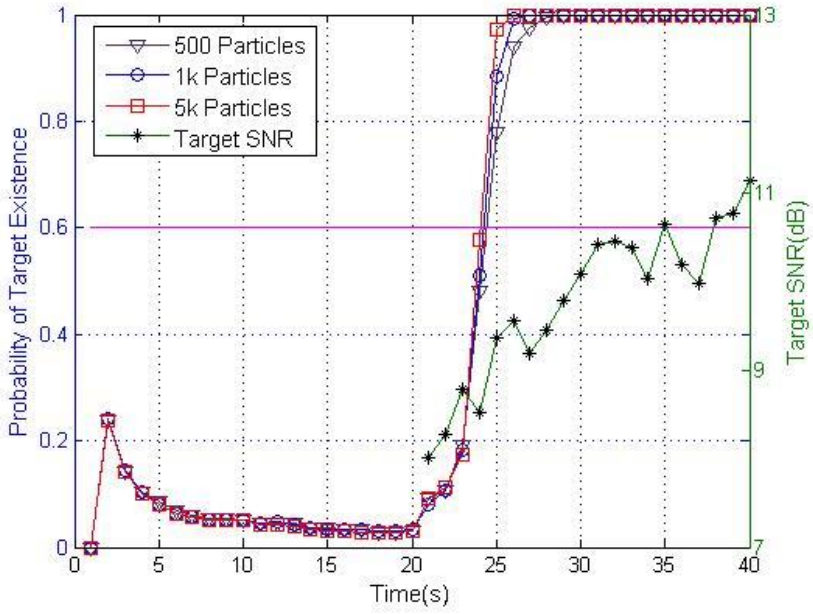


Figure 5.19. The probability of existence of the spawned target and its SNR value (First scenario, 8 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

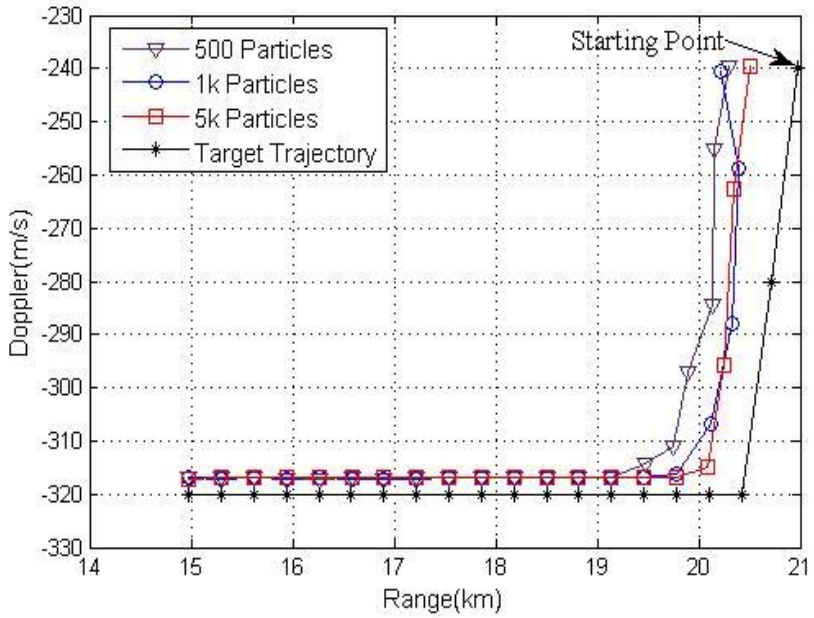


Figure 5.20. Range versus Doppler estimates for the spawned target and its trajectory (First scenario, 8 dB initial SNR of the spawned target)

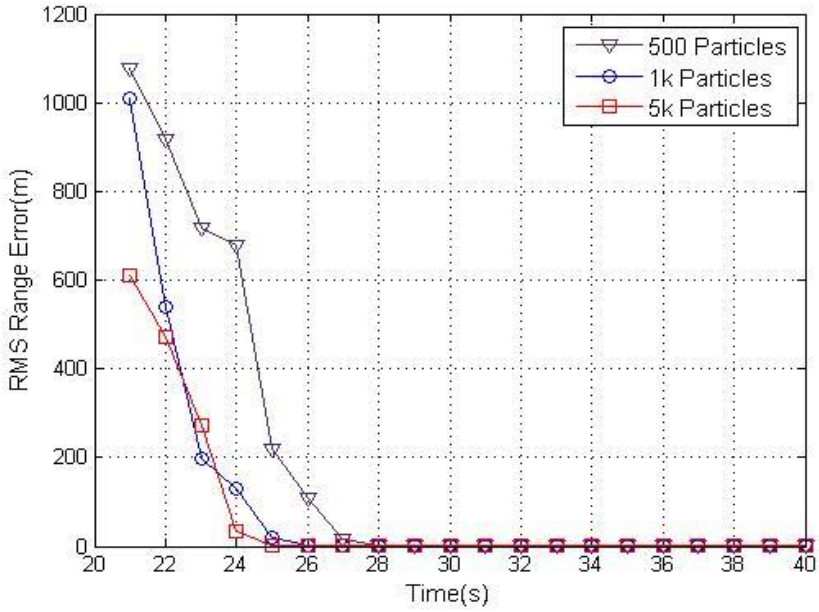


Figure 5.21. The RMS range error for the spawned target (First scenario, 8 dB initial SNR of the spawned target)

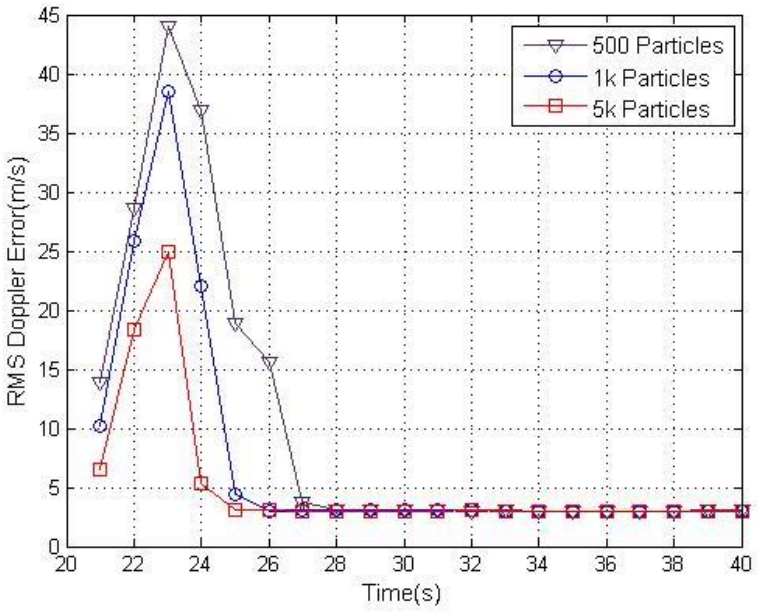


Figure 5.22. The RMS Doppler error for the spawned target (First scenario, 8 dB initial SNR of the spawned target)

As it can be seen in figures above, in the transient part, the accuracies of the range and Doppler estimates for the spawned target decrease; whereas, the RMS errors for the spawned target increase compared to the case of initial 10 dB SNR of the spawned target.

The ratio N_{eff}/N for the spawned target can be seen in Figure 5.23.

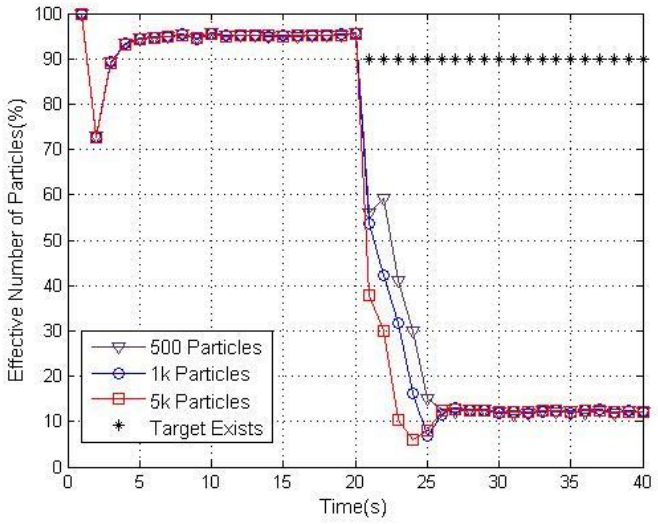


Figure 5.23. N_{eff}/N for the spawned target (First scenario, 8 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

The SNR estimates for the spawned target and its actual SNR values can be seen in Figure 5.24.

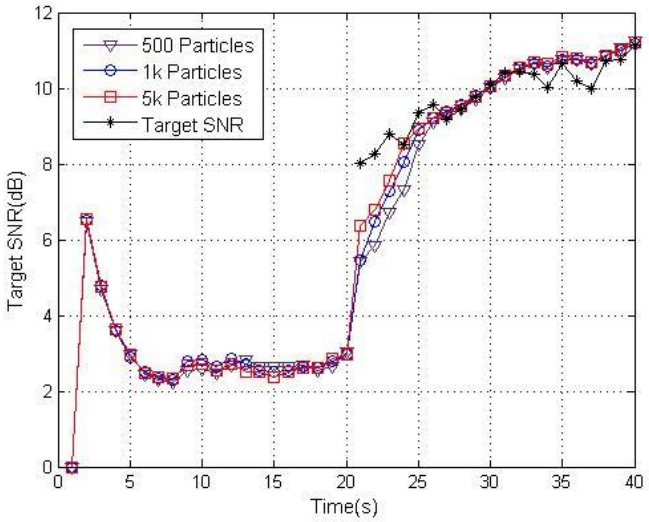


Figure 5.24. The SNR estimates for the spawned target (First scenario, 8 dB initial SNR of the spawned target)

5.3.1.1.3 Initial SNR of the Spawned Target: 6 dB

The initial SNR of the spawned target is decreased to the value of 6dB to determine the minimum SNR value of the weak target at which Algorithm 1 is successful at detection and tracking of both targets. Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.25 shows the probability of the spawned target's existence for different number of particles. As it can be seen in Figure 5.25, the existence of the spawned target is declared in 10 seconds for 500 particles, 9 seconds for 1k and 5k particles. The maximum SNR value is around 8 dB.

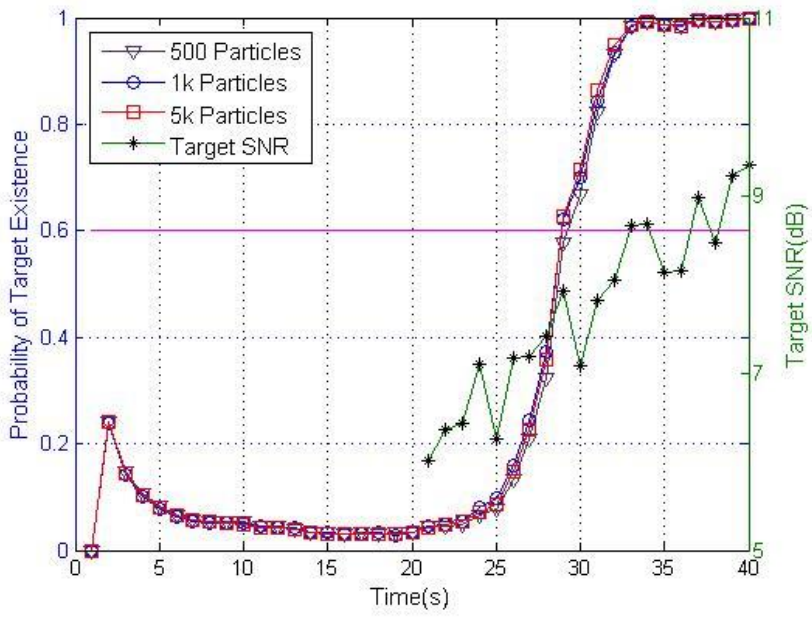


Figure 5.25. The probability of existence of the spawned target and its SNR value (First scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

Figure 5.26 shows the range versus Doppler estimates and the actual positions of the spawned target.

The RMS range and Doppler errors for the spawned target are given in Figure 5.27 and Figure 5.28, respectively.

The ratio N_{eff}/N for the spawned target can be seen in Figure 5.29. Furthermore, the SNR estimates for the spawned target and its actual SNR values can be seen in Figure 5.30.

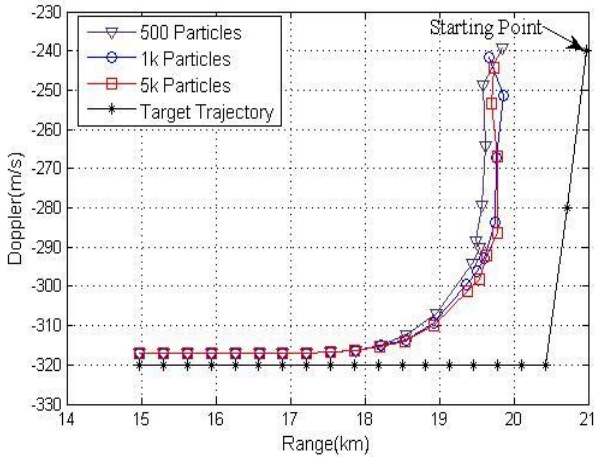


Figure 5.26. Range versus Doppler estimates for the spawned target and its trajectory (First scenario, 6 dB initial SNR of the spawned target)

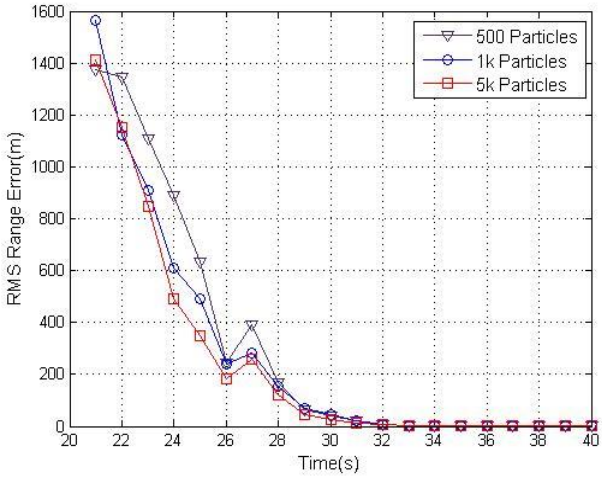


Figure 5.27. The RMS range error for the spawned target (First scenario, 6 dB initial SNR of the spawned target)

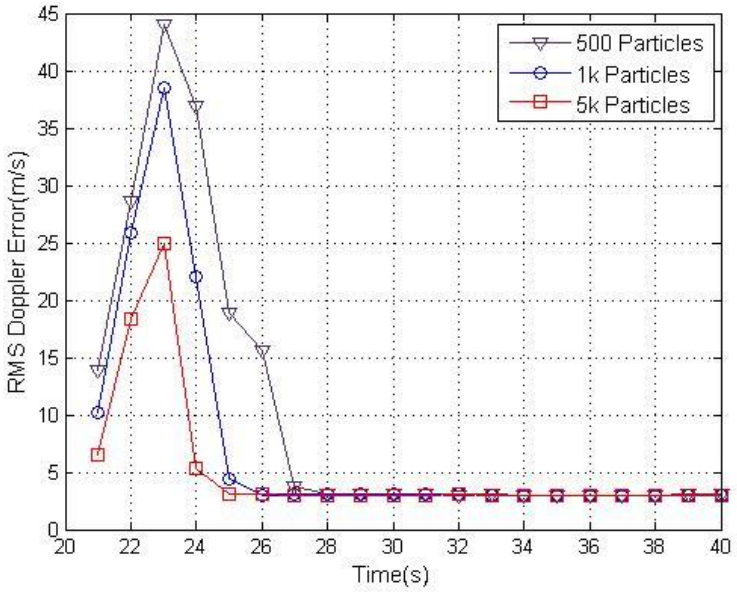


Figure 5.28. The RMS Doppler error for the spawned target (First scenario, 6 dB initial SNR of the spawned target)

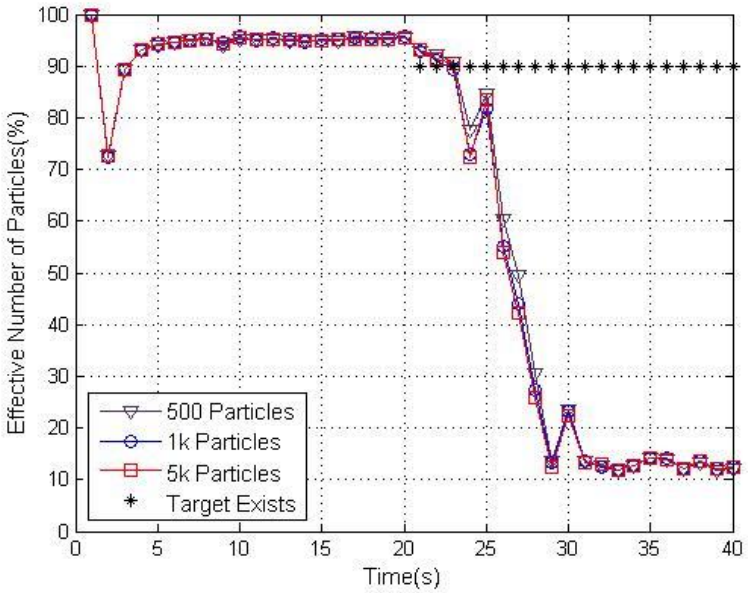


Figure 5.29. N_{eff}/N for the spawned target (First scenario, 6 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

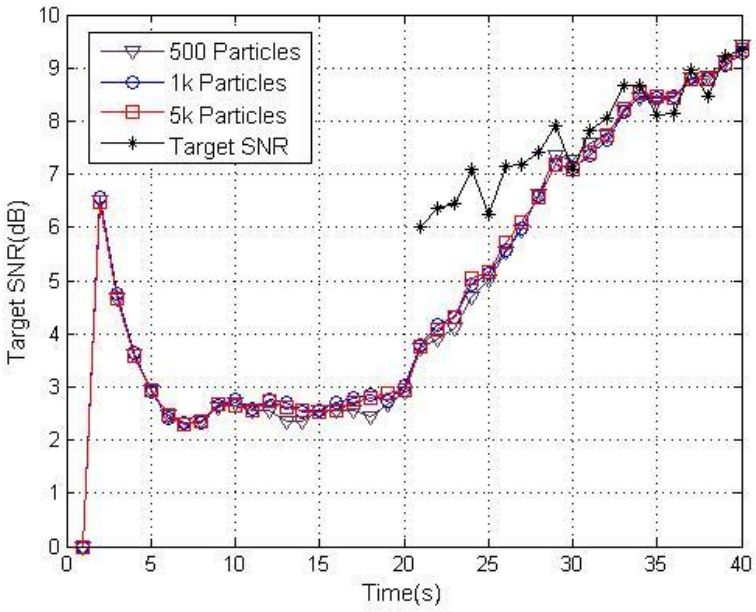


Figure 5.30. The SNR estimates for the spawned target (First scenario, 6 dB initial SNR of the spawned target)

5.3.1.1.4 Initial SNR of the Spawned Target: 4 dB

The initial SNR of the spawned target is again decreased to the value of 4dB to determine the minimum SNR value of the spawned target at which Algorithm 1 is successful at detection and tracking of both targets. As it can be seen in Figure 5.31, Algorithm 1 is not successful at detecting the spawned target with 4dB initial SNR. Note that the SNR value never exceeds 7.5 dB during the scenario.

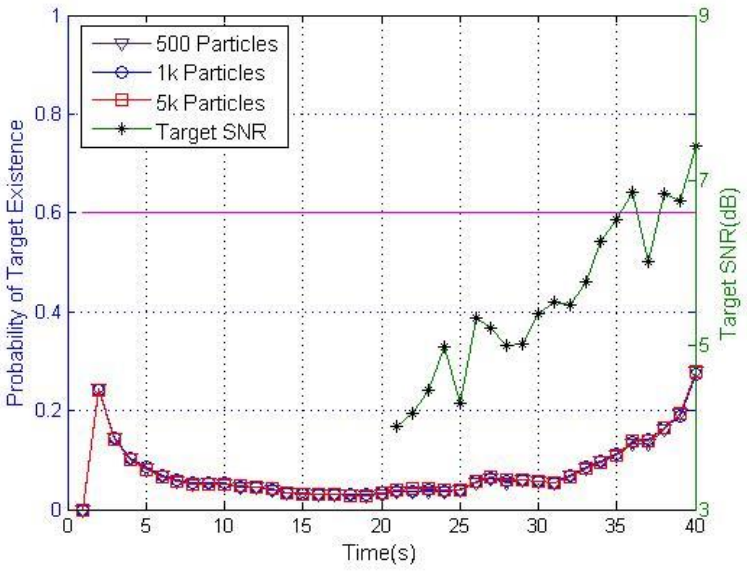


Figure 5.31. The probability of existence of the spawned target and its SNR value (First scenario, 4 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

5.3.1.2 Simulation Results for the Second Scenario

The second scenario is for the spawned target that makes some maneuvers. The spawned target maneuvers with $+3g$ between $t = 28\ s$ and $t = 32\ s$, $-3g$ between $t = 33\ s$ and $t = 37\ s$ in the second scenario. The trajectories of the targets are shown in Figure 5.2. Furthermore, the initial SNR of the spawned target is 10 dB and 35 Monte Carlo simulations are performed to obtain all of the results given below.

Figure 5.32 shows the probability of the spawned target's existence for different number of particles. As seen in Figure 5.32 , there is no change in the probability of existence during the maneuvers of the spawned target. This means that Algorithm 1 can deal with the maneuvers of the spawned target.

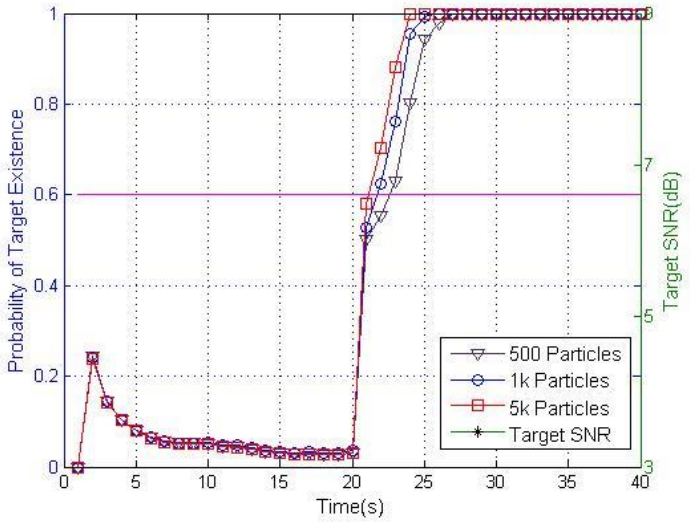


Figure 5.32. The probability of existence of the spawned target and its SNR value (Second scenario, 10 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

Figure 5.33 shows the range versus Doppler estimates and the actual positions of the spawned target.

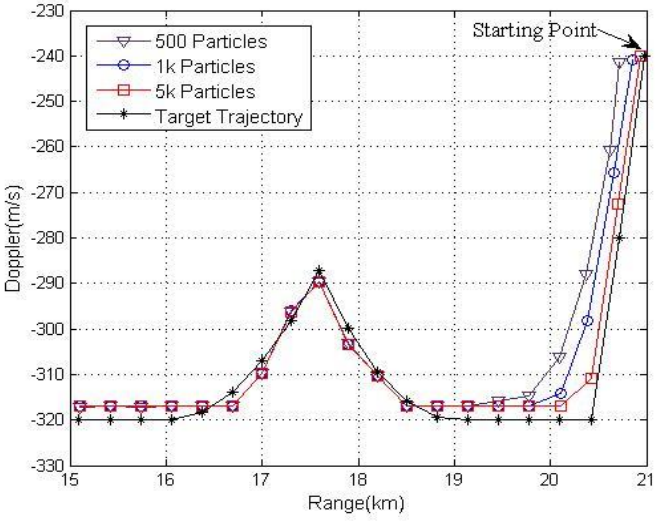


Figure 5.33 Range versus Doppler estimates for the spawned target and its trajectory (Second scenario, 10 dB initial SNR of the spawned target)

The RMS range and Doppler errors for the spawned target are given in Figure 5.34 and Figure 5.35, respectively.

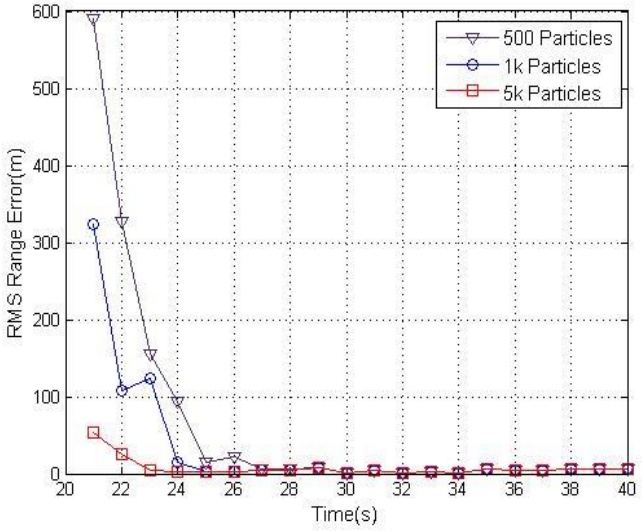


Figure 5.34. The RMS range error for the spawned target (Second scenario, 10 dB initial SNR of the spawned target)

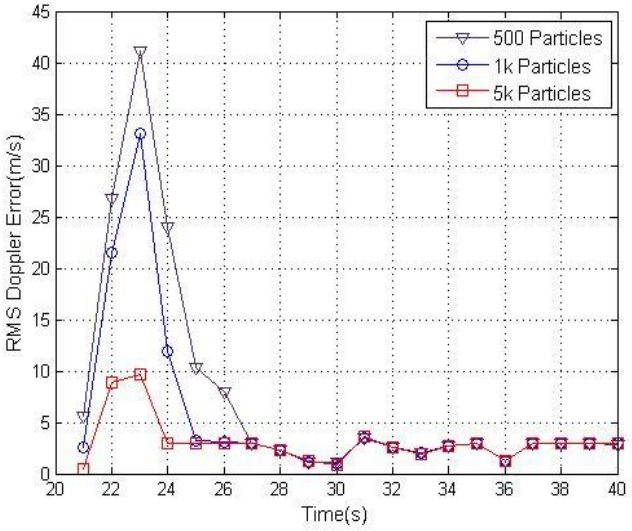


Figure 5.35. The RMS Doppler error for the spawned target (Second scenario, 10 dB initial SNR of the spawned target)

As seen in Figure 5.33, Figure 5.34 and Figure 5.35, the accuracies of the estimates don't decrease with the maneuvers of the spawned target.

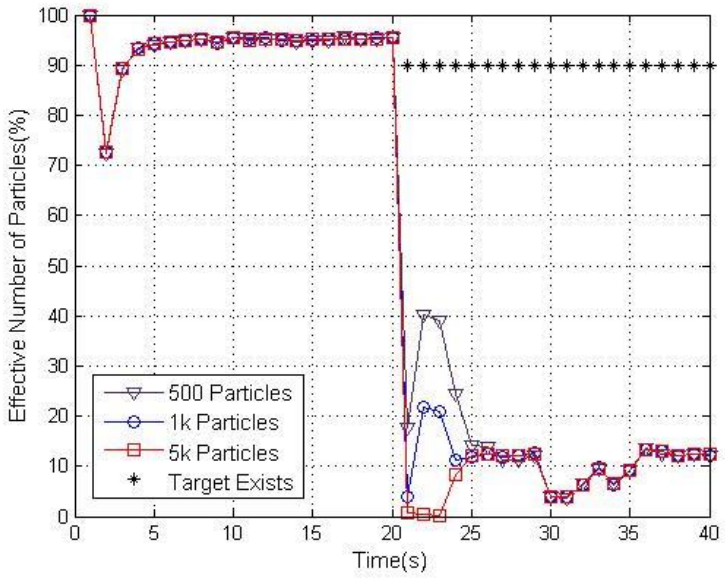


Figure 5.36. N_{eff}/N for the spawned target (Second scenario, 10 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

Figure 5.36 shows the ratio N_{eff}/N for the spawned target. As it can be seen in Figure 5.36, the effective number of particles decreases between $t = 30\ s$ and $t = 35\ s$. This means that the sample impoverishment problem becomes serious at time steps when the spawned target maneuvers although the detection performance and the accuracies of the estimates don't decrease.

Figure 5.37 shows that the SNR estimates for the spawned target and its actual SNR values. It can be seen that the accuracies of the SNR estimates also don't decrease with the maneuvers of the spawned target.

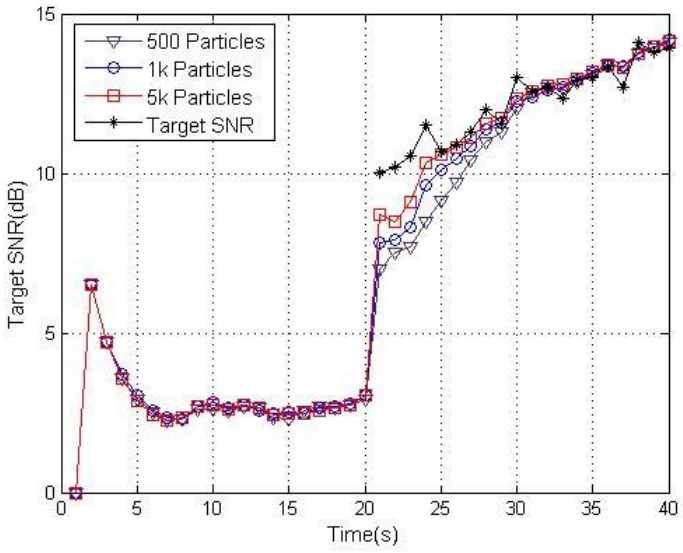


Figure 5.37. The SNR estimates for the spawned target (Second scenario, 10 dB initial SNR of the spawned target)

5.3.1.3 Simulation Results for the Third Scenario

In the third scenario, we want to simulate a highly maneuvering spawned target since the spawned targets are usually highly maneuvering targets in the real world. In this scenario, the spawned target maneuvers so highly that the limit of the algorithms' performances is reached. We want to show that the process noise identification method gives a chance to track and detect a highly maneuvering target by not suffering from the sample impoverishment problem by using this scenario.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.3 shows the trajectories of the targets in the third scenario. Note that the spawned target maneuvers with $-7g$ between $t = 28\,s$ and $t = 32\,s$, $+7g$ between $t = 33\,s$ and $t = 37\,s$ in this scenario.

Figure 5.38 shows the probability of the spawned target's existences obtained by using 10k particles in Algorithm 1. As seen in Figure 5.38, there is an obvious decrease in the probability of the spawned target's existence between $t = 29\ s$ and $t = 37\ s$ because of the high maneuvers. In fact, Algorithm 1 can't detect the spawned target between $t = 32\ s$ and $t = 36\ s$.

The same scenario is run with process noise identification added to Algorithm 1. The results obtained are shown in Figure 5.39. Figure indicates that no decrease in the probability of target existence during the maneuvers of the spawned target. Therefore, it can be said that the proposed process noise identification method can deal with the problem mentioned above.

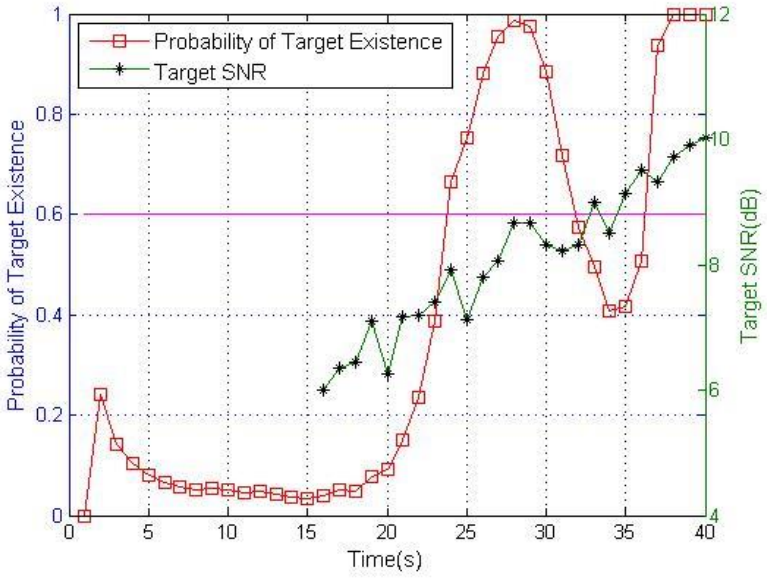


Figure 5.38. The probability of existence of the spawned target and its SNR value (Third scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

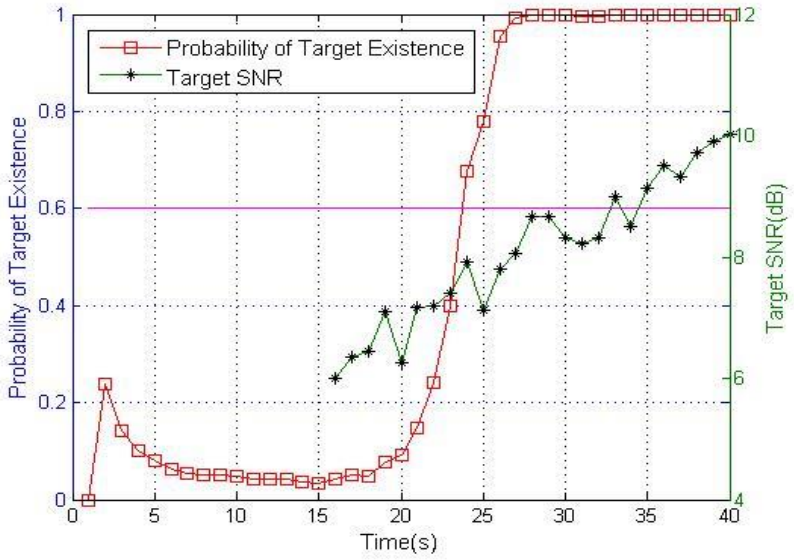


Figure 5.39. Actual SNR values and the probability of existence of the spawned target obtained by using Algorithm 1 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

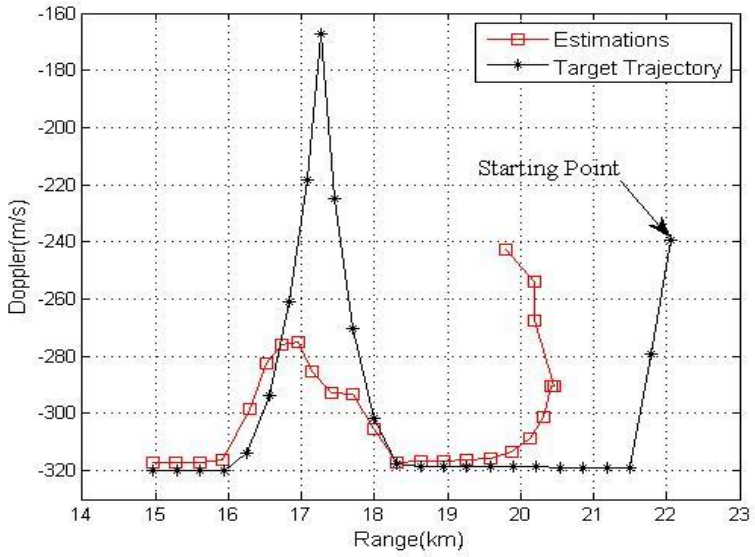


Figure 5.40. Range versus Doppler estimates for the spawned target and its trajectory (Third scenario, 6 dB initial SNR of the spawned target)

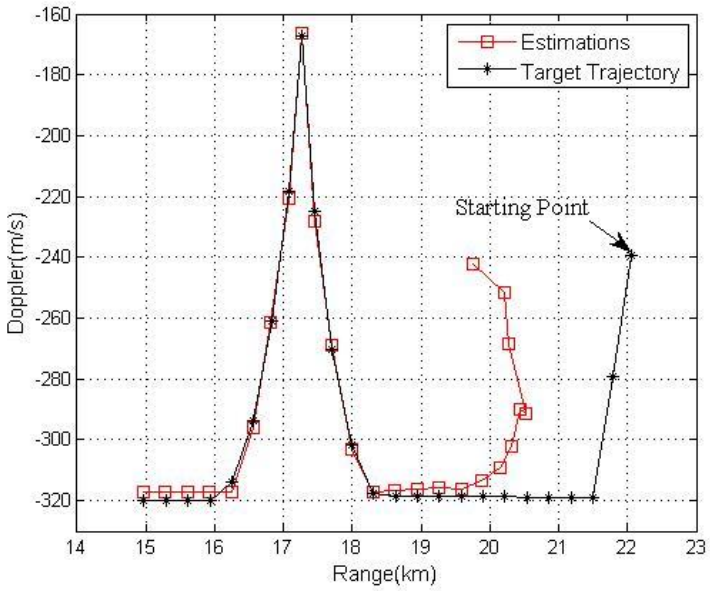


Figure 5.41. Actual trajectory and range versus Doppler estimates for the spawned target obtained by using Algorithm 1 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target)

Figure 5.40 and Figure 5.41 show the range versus Doppler estimates for the spawned target obtained by using Algorithm 1 and Algorithm 1 with process noise identification, respectively. It is obvious that the proposed process noise identification method increases the tracking performance of Algorithm 1.

Figure 5.42 and Figure 5.43 show the RMS range errors for the spawned target obtained by using Algorithm 1 and Algorithm 1 with process noise identification, respectively.

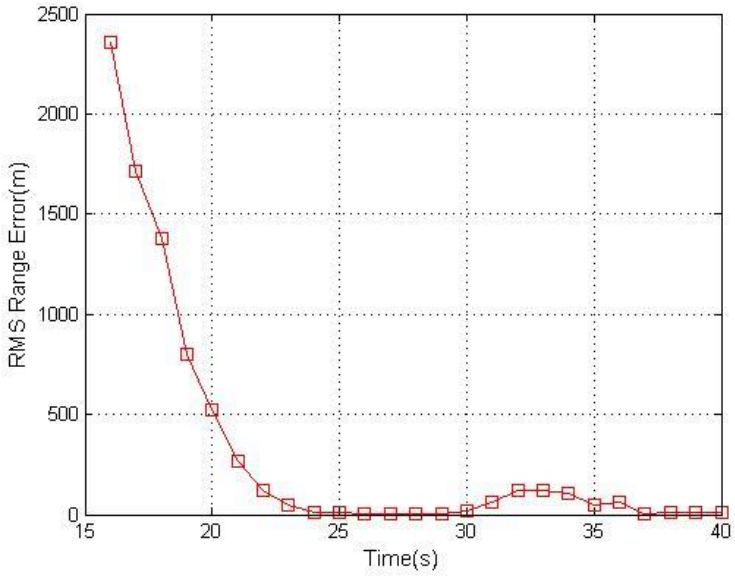


Figure 5.42. The RMS range error for the spawned target (Third scenario, 6 dB initial SNR of the spawned target)

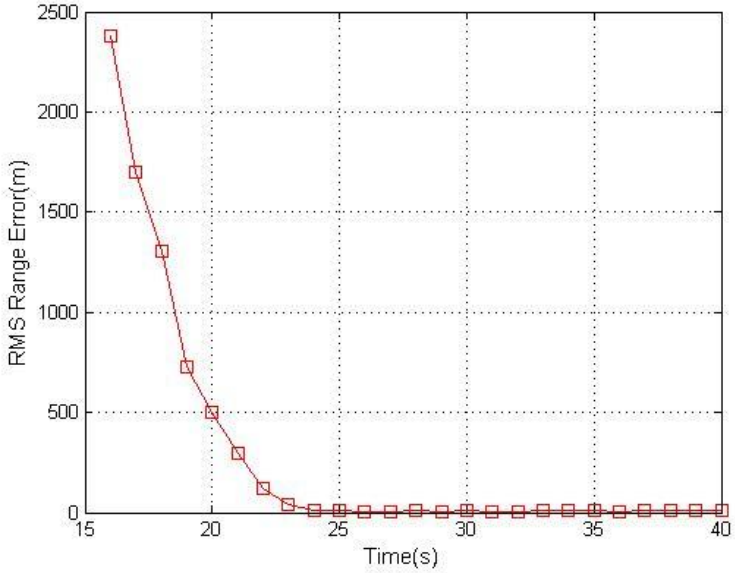


Figure 5.43. The RMS range error for the spawned target obtained by using Algorithm 1 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target)

Figure 5.44 and Figure 5.45 show the RMS Doppler errors for the spawned target obtained by using Algorithm 1 and Algorithm 1 with process noise identification, respectively.

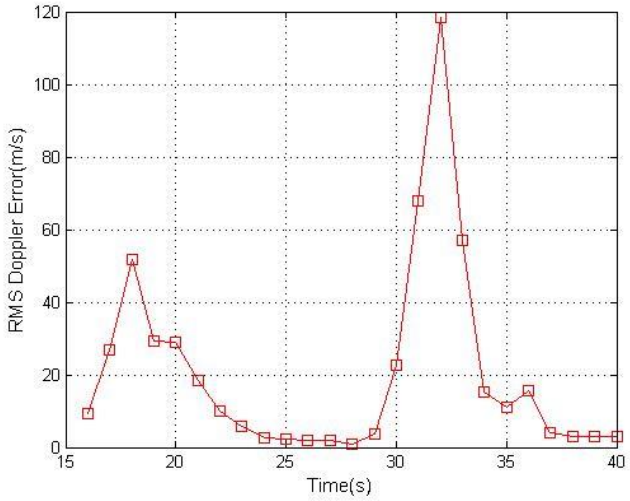


Figure 5.44. The RMS Doppler error for the spawned target (Third scenario, 6 dB initial SNR of the spawned target)

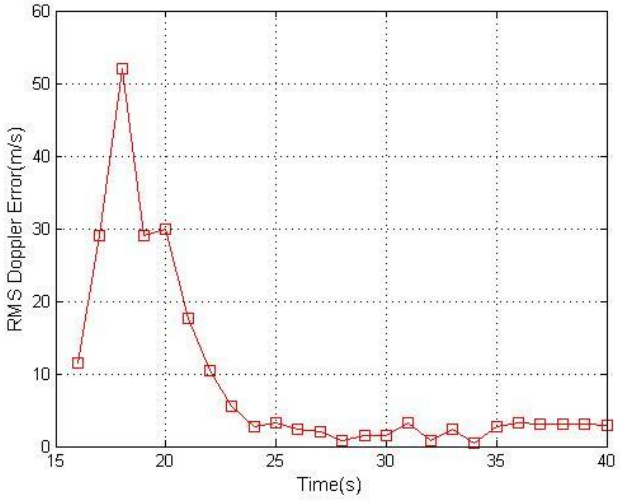


Figure 5.45. The RMS Doppler error for the spawned target obtained by using Algorithm 1 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target)

As seen in figures above, the RMS errors increase with the maneuvers of the spawned target in Algorithm 1; whereas, they don't increase in Algorithm 1 with process noise identification.

Figure 5.46 and Figure 5.47 show the ratio N_{eff}/N for the spawned target obtained by using Algorithm 1 and Algorithm 1 with process noise identification, respectively.

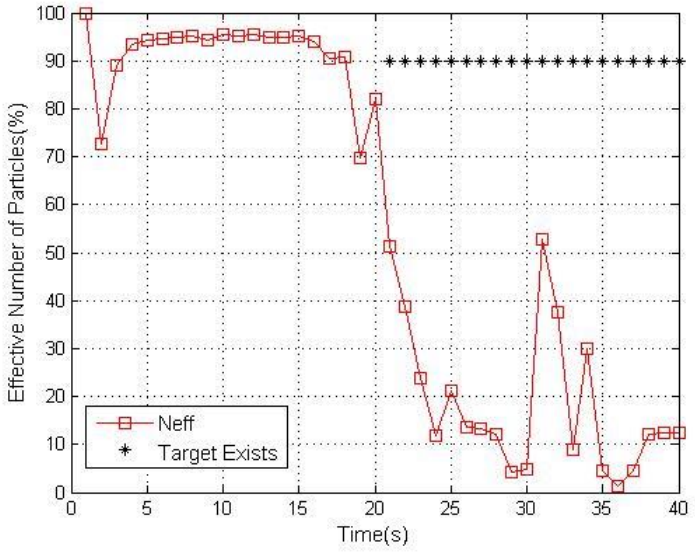


Figure 5.46. N_{eff}/N for the spawned target (Third scenario, 6 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

As mentioned earlier, when the existence of the spawned target is declared, the ratio of the effective number of particles for the spawned target is around 12% which can be said to be low. The reason is that the process noises for the spawned target, $q_k^{r(2)}$, $q_k^{d(2)}$ and $q_k^{u(2)}$, have high variances since the spawned target moves with constant acceleration at first three time steps although the constant velocity model is used as system dynamic model. As seen in Figure 5.47, it is shown that the sample impoverishment problem is solved by the process noise identification

method. Note that the increase in the ratio of N_{eff}/N seen in Figure 5.46 between $t = 30\ s$ and $t = 35\ s$ occurs because the existing particles die and the particles whose mode variable is $m_k^i = 0$ multiply. Since all of the particles whose mode variable is $m_k^i = 0$ have the same weight which is equal to 1, the effective number of particles increases according to (3.23) although the tracking and detection performance of Algorithm 1 decrease.

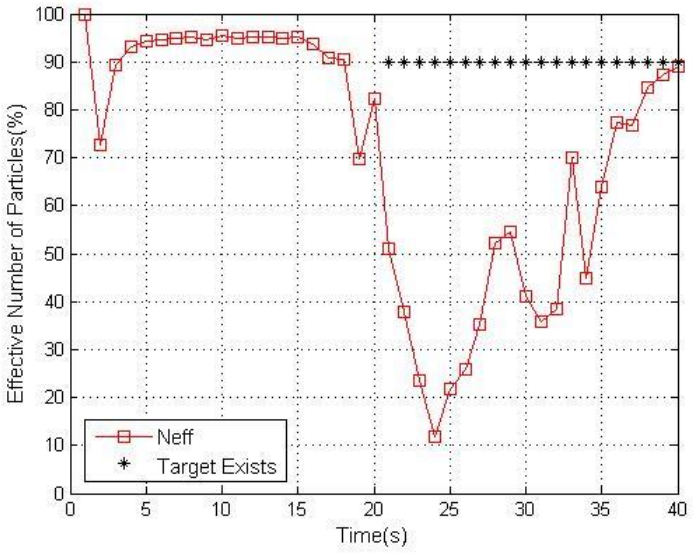


Figure 5.47. N_{eff}/N for the spawned target obtained by using Algorithm 1 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

5.3.1.4 Simulation Results for the Fourth Scenario

As mentioned earlier, in the fourth scenario, Swerling-0 type of targets which move very slowly are used in order to see the detection times independently from the changes in the SNR values of the targets. The mean SNR values change very slightly during the fourth scenario since the targets move with very low velocities. Furthermore, there is no target SNR fluctuations since Swerling-0 type of targets

are used in this scenario. Initial SNR of the spawned target is 6 dB and the SNR value never exceeds 6.4 dB during the scenario.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.48 shows the probability of the spawned target's existence for different number of particles. As it can be seen in the figure, the existence of the spawned target is declared in 8 seconds for 5k particles, 9 seconds for 500 and 1k particles. As mentioned in Section 5.3.1.1.3, in the first scenario in which Swerling-1 type of targets are used and the initial SNR of the spawned target is 6 dB, the existence of the spawned target is declared in 10 seconds for 500 particles, 9 seconds for 1k and 5k particles. In the fourth scenario, the detection times decrease for 500 and 5k particles compared to the first scenario. The reason is that there is no target SNR fluctuations since Swerling-0 type of targets are used in the fourth scenario.

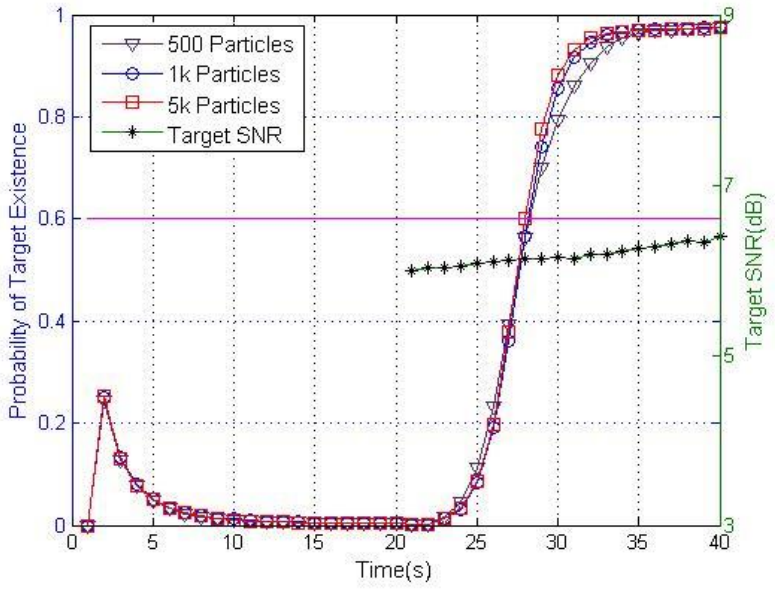


Figure 5.48. The probability of existence of the spawned target and its SNR value (Fourth scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target.

It can be concluded from the results given above that Algorithm 1 is successful at detection and tracking of the spawned targets with SNR values as low as 6 dB.

5.3.1.5 Summary of the Results of Algorithm 1

The detection and tracking performance of Algorithm 1 is analyzed for different scenarios. As mentioned earlier, Swerling-1 model is used in all scenarios. The target declaration times for different initial target SNR and different number of particles are shown for the spawned target in Table 5.4. The SNR value of the spawned target at the declaration of its existence is also shown in parenthesis.

Table 5.4. Target declaration times for different initial SNR values of the spawned target and different number of particles

	10 dB	8 dB	6 dB
500 Particles	4 sec (at 11.5 dB)	5 sec (at 9.35 dB)	10 sec (at 7.08 dB)
1k Particles	3 sec (at 10.5 dB)	5 sec (at 9.35 dB)	9 sec (at 7.91 dB)
5k Particles	2 sec (at 10.2 dB)	5 sec (at 9.35 dB)	9 sec (at 7.91 dB)

It is shown that Algorithm 1 is successful at target detection for initial target SNR as low as 6dB. In Table 5.4, it can be seen that the target declaration times increase as the target SNR decreases; whereas, increasing number of particles reduces the target declaration times as expected.

The performance of Algorithm 1 is analyzed on the maneuvering targets by using the second and third scenarios. As the results of the second scenario shows that Algorithm 1 is successful at detection and tracking of a maneuvering target with $+3g$ maneuver. Furthermore, it is shown that the proposed process noise identification method is very successful for maneuvering targets. It increases the performance of Algorithm 1 considerably and enables it to track and detect a highly maneuvering target without suffering the sample impoverishment problem.

5.3.2 Simulation Results of Algorithm 2

5.3.2.1 Simulation Results for the First Scenario

The performance of the Algorithm 2 is analyzed for different initial SNR values of the spawned target as 10 dB, 8 dB, 6 dB, 4 dB. In all simulations, the initial SNR of the main platform is 18 dB.

Note that 35 Monte Carlo simulations are performed to obtain all of the results given below.

5.3.2.1.1 Initial SNR of the Spawned Target: 10 dB

The analysis of the performance of Algorithm 2 is started with the 10dB initial SNR of the spawned target.

Figure 5.49, Figure 5.50 and Figure 5.51 show the mode probabilities for 500, 1k and 5k particles, respectively. The main platform is detected at the moment that it appears in the surveillance region, e.g. there is a rapid increase of the probability of one target present after the appearance of the main platform because of its high SNR value (18 dB). The existence of the spawned target is also declared in 3 seconds for 500 particles and 1 second for 1k and 5k particles as seen in the figures below.

Figure 5.52 and Figure 5.53 show the range versus Doppler estimates for the main platform and the spawned target, respectively. The accuracies of the estimates for the main platform slightly decrease when the spawned target appears in the surveillance region since some newborn particles belonged to the main platform appears in the vicinity of the spawned target. As seen in Figure 5.52, the estimates between the $range = 20\ km$ and $range = 21\ km$ are the examples of this.

The RMS range and Doppler errors for the main platform are given in Figure 5.54 and Figure 5.55, respectively.

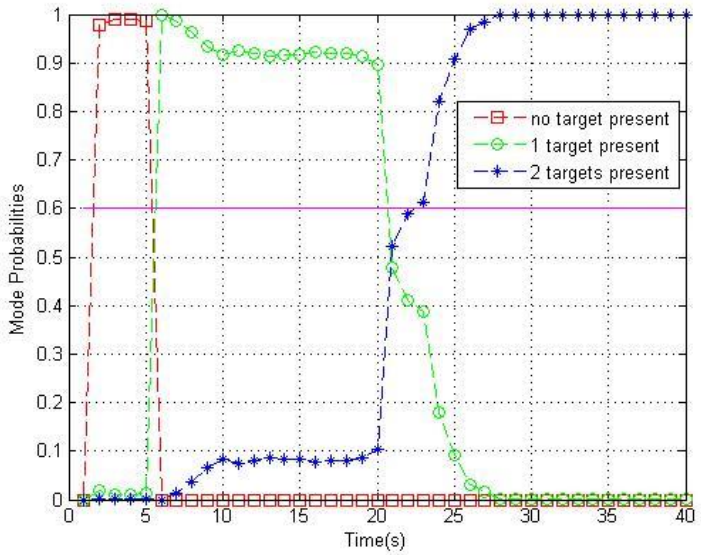


Figure 5.49. The mode probabilities for 500 particles (First Scenario, 10dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

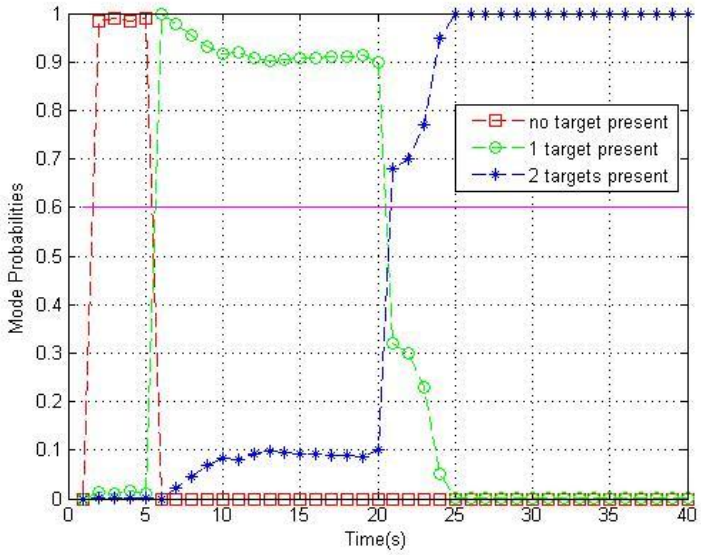


Figure 5.50. The mode probabilities for 1k particles (First Scenario, 10dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

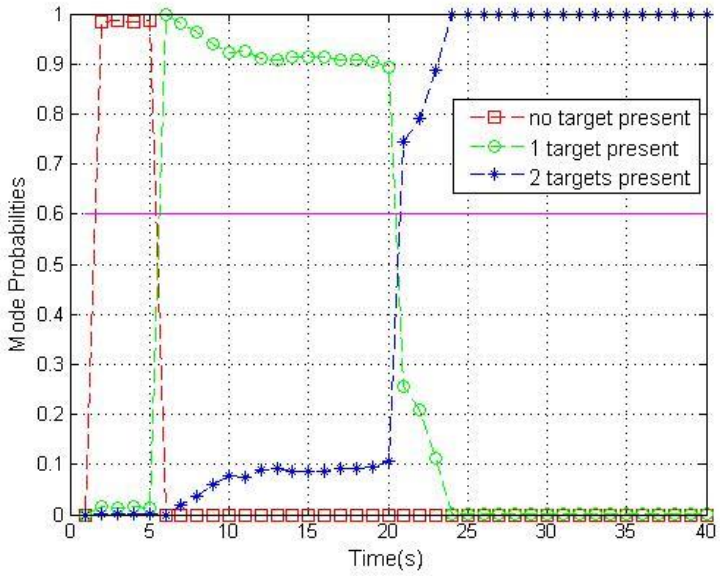


Figure 5.51. The mode probabilities for 5k particles (First Scenario, 10dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

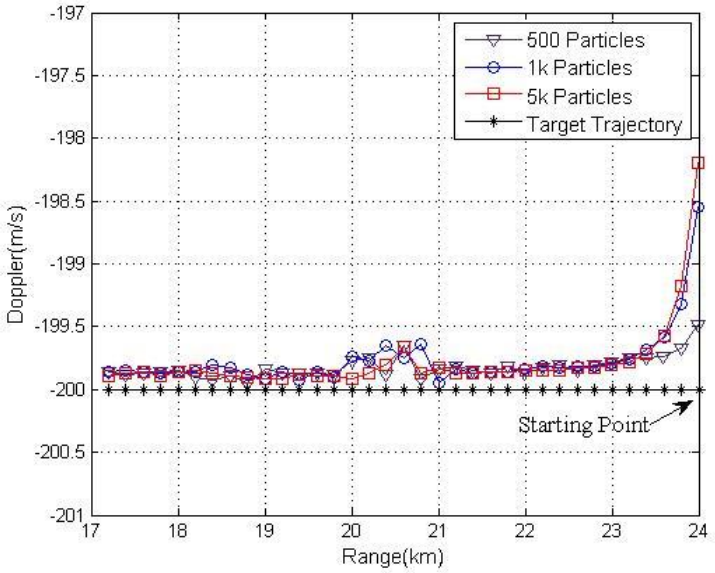


Figure 5.52. Range versus Doppler estimates for the main platform and its trajectory (First Scenario, 10dB initial SNR of the spawned target)

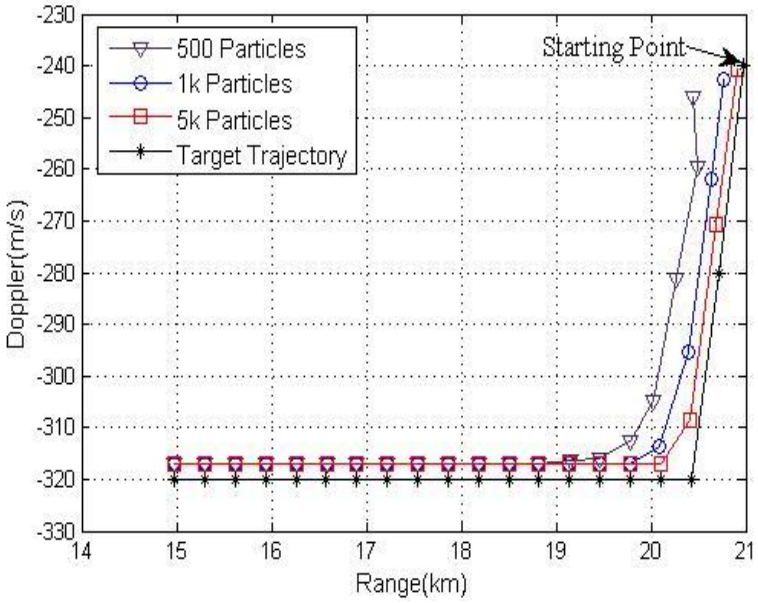


Figure 5.53. Range versus Doppler estimates for the spawned target and its trajectory (First Scenario, 10dB initial SNR of the spawned target)

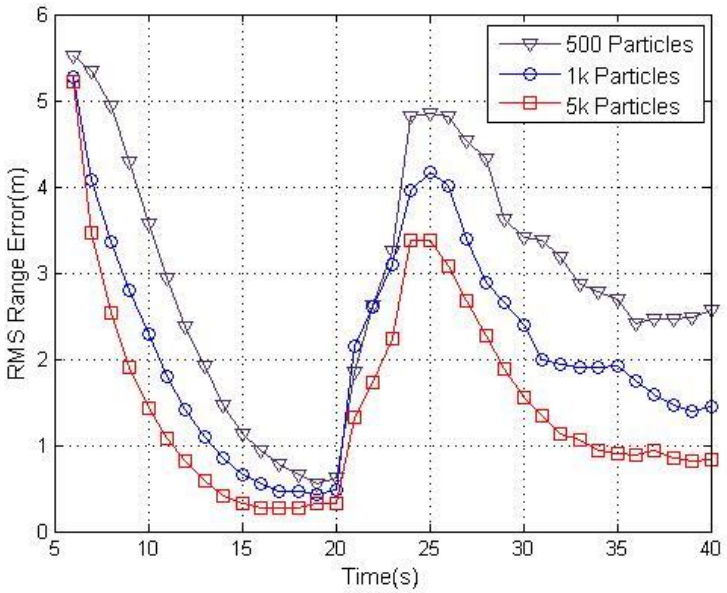


Figure 5.54. The RMS range error for the main platform (First Scenario, 10 dB initial SNR of the spawned target)

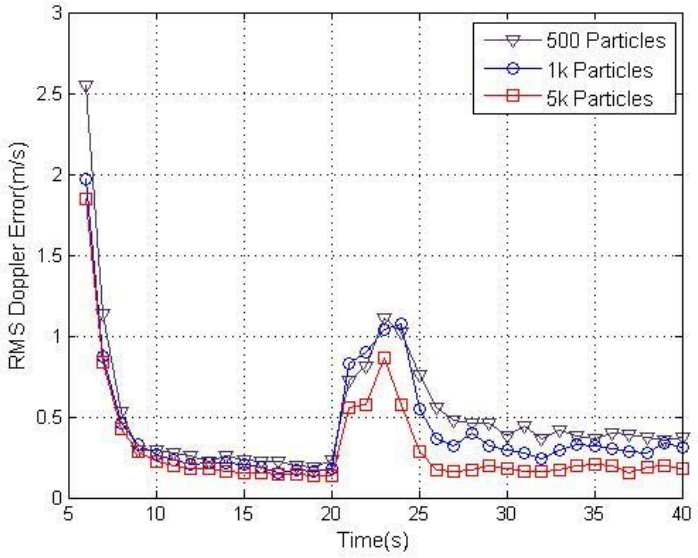


Figure 5.55. The RMS Doppler error for the main platform (First Scenario, 10 dB initial SNR of the spawned target)

The RMS range and Doppler errors for the spawned target are given in Figure 5.56 and Figure 5.57, respectively.

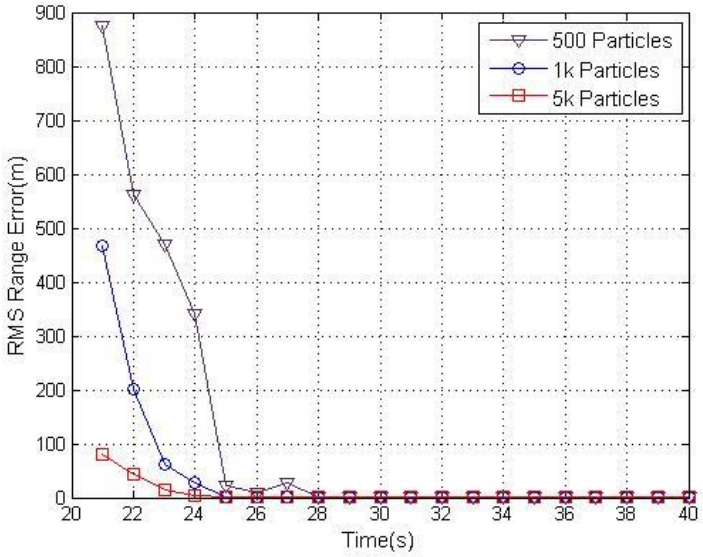


Figure 5.56. The RMS range error for the spawned target (First Scenario, 10 dB initial SNR of the spawned target)

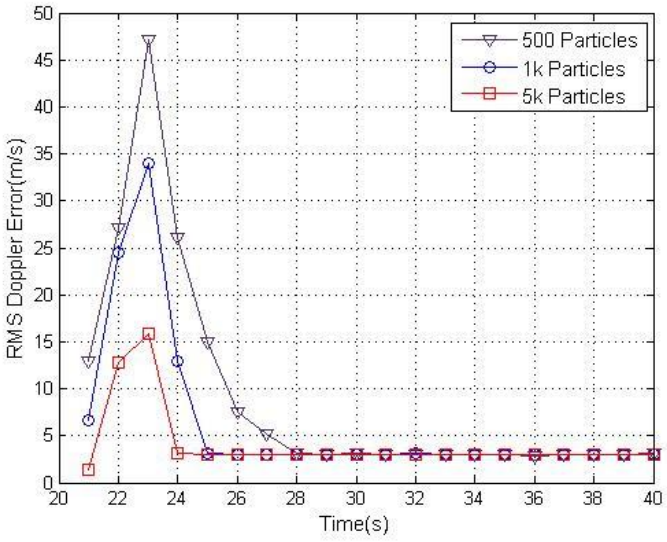


Figure 5.57. The RMS Doppler error for the spawned target (First Scenario, 10 dB initial SNR of the spawned target)

The ratio of N_{eff}/N can be seen in Figure 5.58.

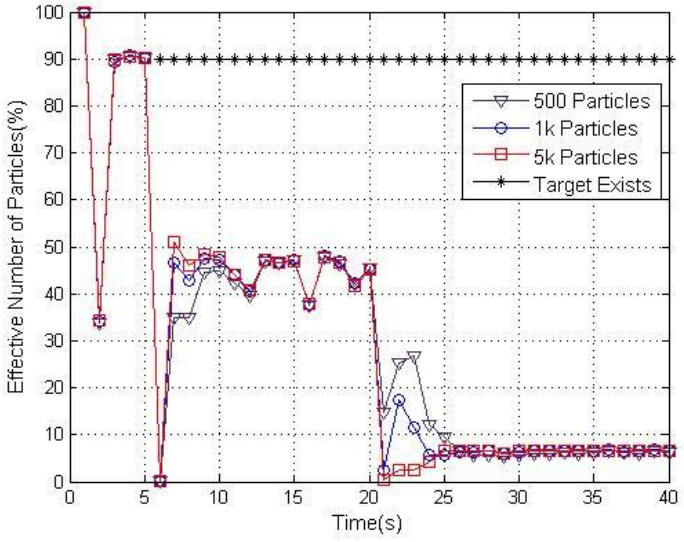


Figure 5.58. N_{eff}/N (First Scenario, 10dB initial SNR of the weak target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

After the existence of the spawned target is declared, the ratio of the effective number of particles is around 7% which can be said to be very low. This value is lower than the value obtained in Algorithm 1 since the number of states in the state vector increases to eight from four in Algorithm 2.

As mentioned earlier, Algorithm 2 is able to estimate the SNR values of the targets. The SNR estimates for the main platform and the spawned target with the actual SNR values of the targets can be seen in Figure 5.59 and Figure 5.60, respectively.

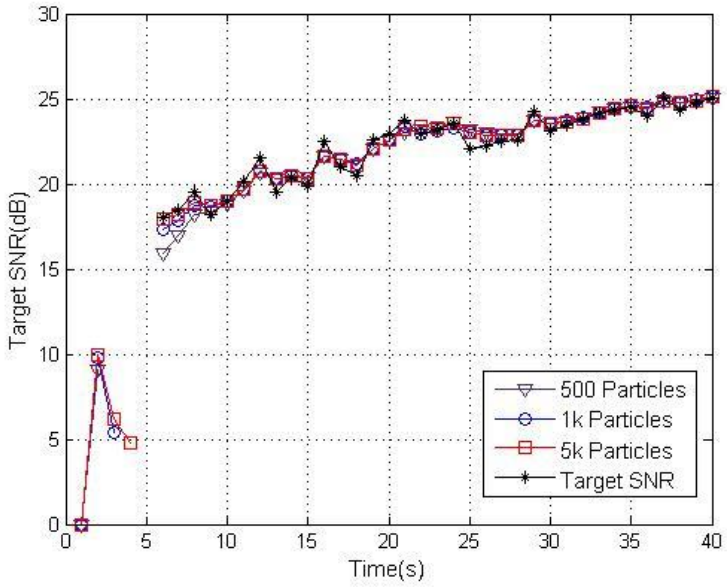


Figure 5.59. The SNR estimates for the main platform (First Scenario, 10 dB initial SNR of the spawned target)

As seen in Figure 5.59, there is no SNR estimate at some time steps. This is because there is not any particles whose mode variables are $m_k^i = 1$ at that time steps. Furthermore, there are also not any SNR estimates at some time steps in Figure 5.60. For this case, the reason is that there are no particles whose mode variables are $m_k^i = 2$ at that time steps.

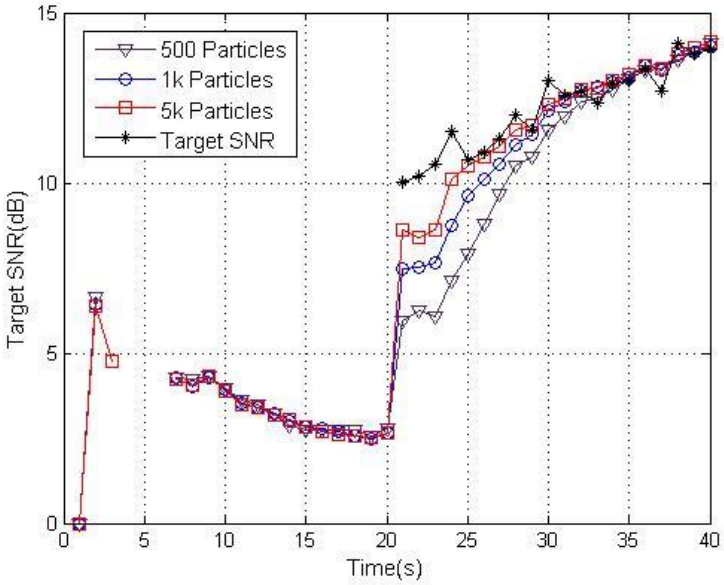


Figure 5.60. The SNR estimates for the spawned target (First Scenario, 10 dB initial SNR of the spawned target)

5.3.2.1.2 Initial SNR of the Spawned Target: 8 dB

The initial SNR of the spawned target is decreased to the value of 8dB to determine the minimum SNR value of the weak target at which Algorithm 2 is successful at detection and tracking of both targets. The results about the main platform aren't shown in this section again since it is observed that decreasing SNR of the spawned target doesn't affect the performance of tracking and detection of the main platform. Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.61, Figure 5.62 and Figure 5.63 show the mode probabilities for 500, 1k and 5k particles, respectively. The detection times for the spawned target increases compared to the case of 10 dB initial target SNR as it is expected. As it can be seen in figures below, the existence of the spawned target is declared in 5 seconds for 500, 4 seconds for 1k and 5k particles after the appearance of the spawned target.

Figure 5.64 shows the range versus Doppler estimates and the actual positions of the spawned target.

The RMS range and Doppler errors for the spawned target are given in Figure 5.65 and Figure 5.66, respectively.

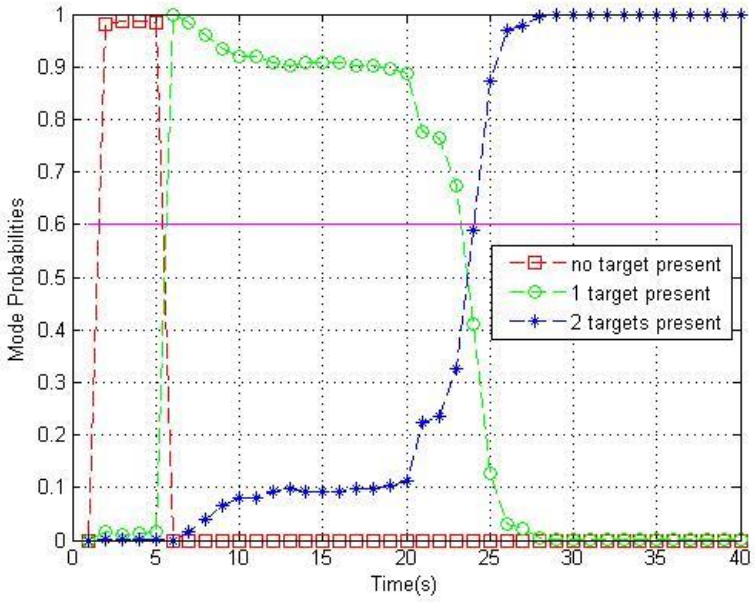


Figure 5.61. The mode probabilities for 500 particles (First Scenario, 8dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

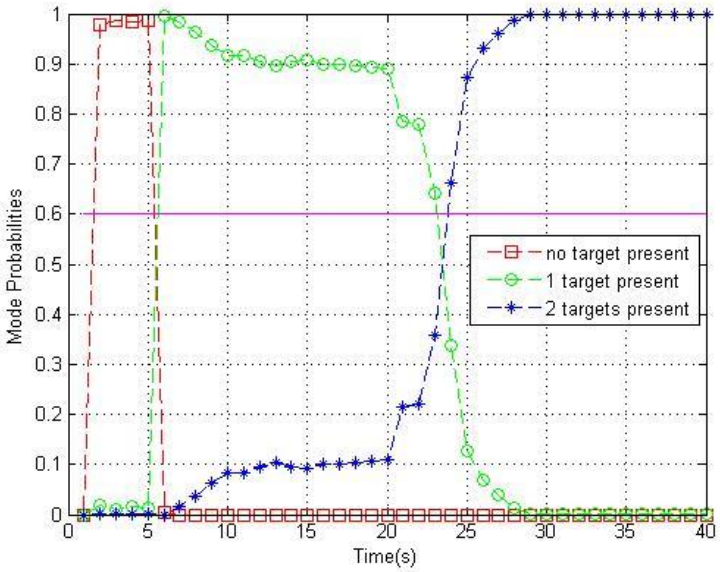


Figure 5.62. The mode probabilities for 1k particles (First Scenario, 8dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

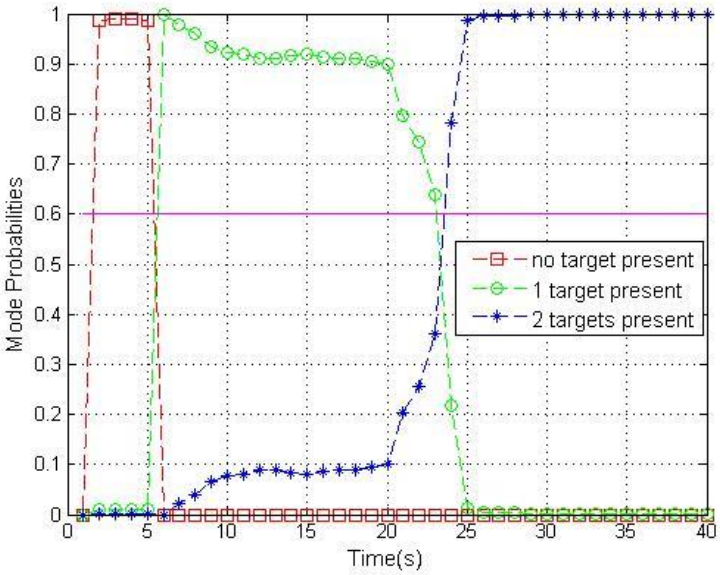


Figure 5.63. The mode probabilities for 5k particles (First Scenario, 8dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

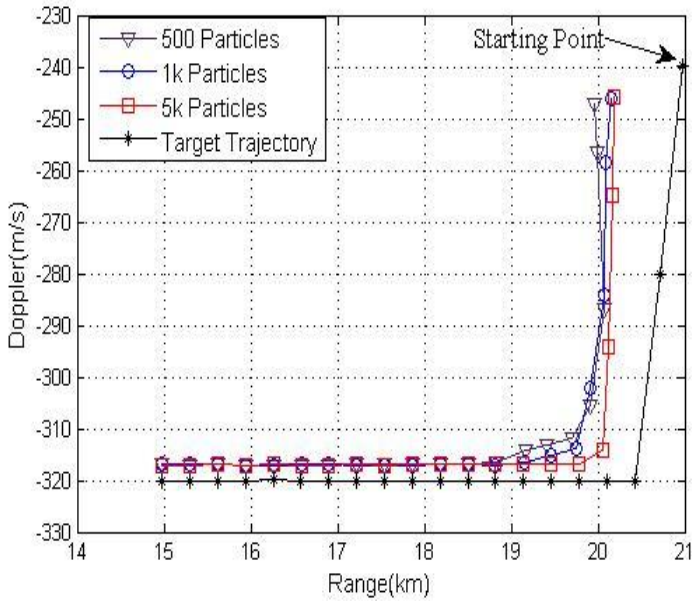


Figure 5.64. Range versus Doppler estimates for the spawned target and its trajectory (First Scenario, 8 dB initial SNR of the spawned target)

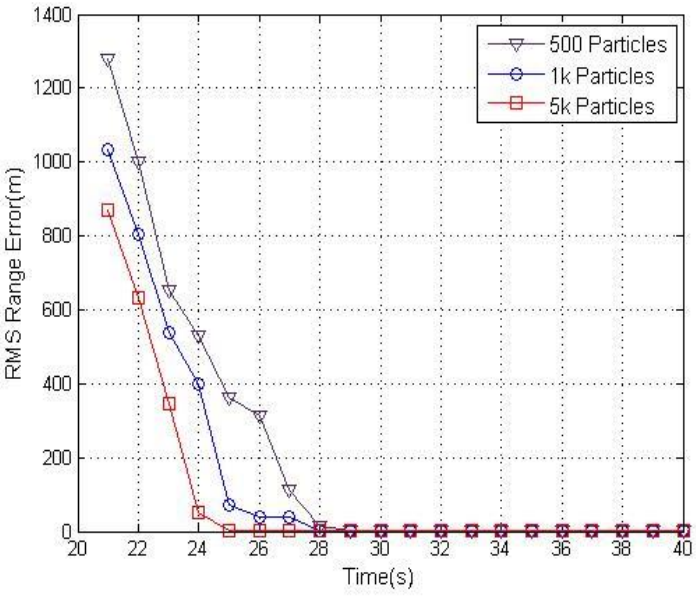


Figure 5.65. The RMS range error for the spawned target (First Scenario, 8 dB initial SNR of the spawned target)

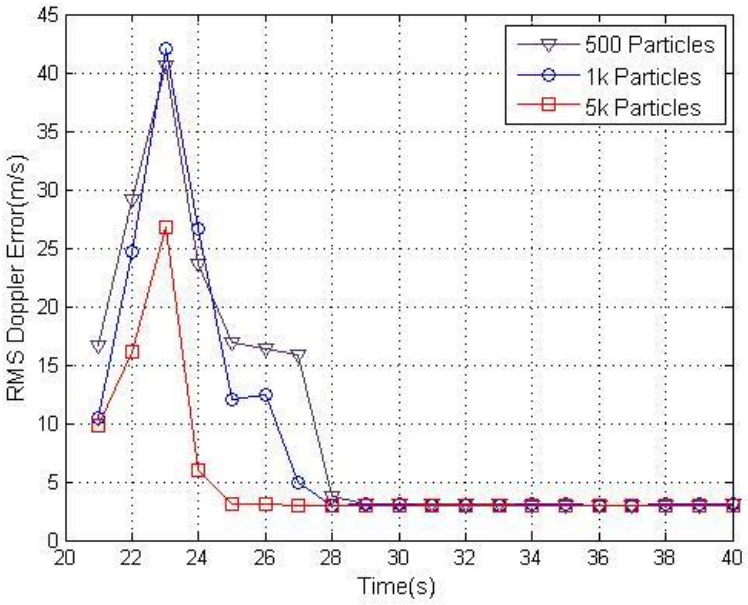


Figure 5.66. The RMS Doppler error for the spawned target (First Scenario, 8 dB initial SNR of the spawned target)

As it can be seen in figures above, in the transient part, the accuracies of the range and Doppler estimates for the spawned target decrease; whereas, the RMS errors for the spawned target increase compared to the case of initial 10 dB SNR of the spawned target.

The ratio N_{eff}/N for the spawned target can be seen in Figure 5.67.

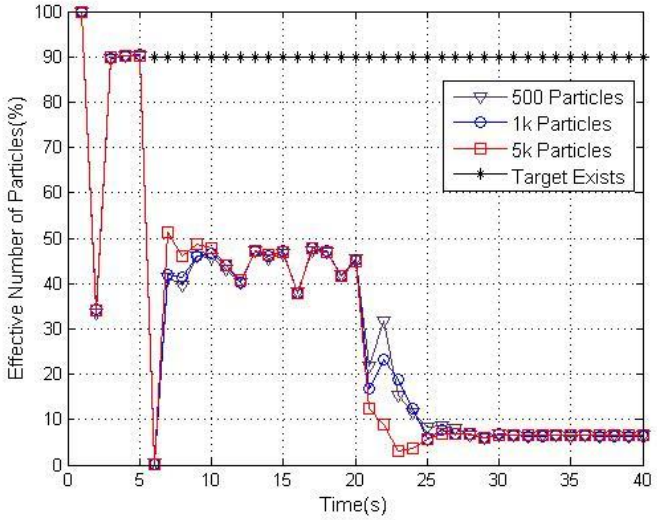


Figure 5.67. N_{eff}/N (First Scenario, 8 dB initial SNR of the weak target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

The SNR estimates for the spawned target and its actual SNR values can be seen in Figure 5.68.

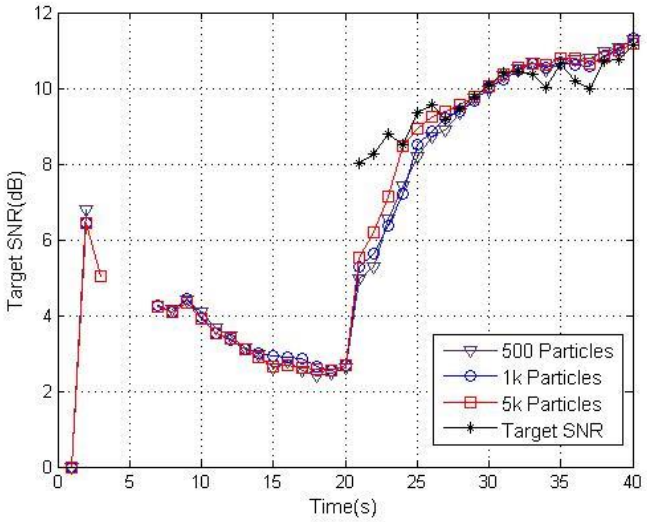


Figure 5.68. The SNR estimates for the spawned target (First Scenario, 8 dB initial SNR of the spawned target)

5.3.2.1.3 Initial SNR of the Spawned Target: 6 dB

The initial SNR of the spawned target is again decreased to the value of 6dB to determine the minimum SNR value of the weak target at which Algorithm 2 is successful at detection and tracking of both targets.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.69, Figure 5.70 and Figure 5.71 show the mode probabilities for 500, 1k and 5k particles, respectively. As it can be seen these figures, the existence of the spawned target is declared in 9 seconds for 500 and 1k particles, 8 seconds for 5k particles.

Figure 5.72 shows the range versus Doppler estimates and the actual positions of the spawned target.

The RMS range and Doppler errors for the spawned target are given in Figure 5.73 and Figure 5.74, respectively.

The ratio N_{eff}/N for the spawned target can be seen in Figure 5.75. Furthermore, the SNR estimates for the spawned target and its actual SNR values can be seen in Figure 5.76.

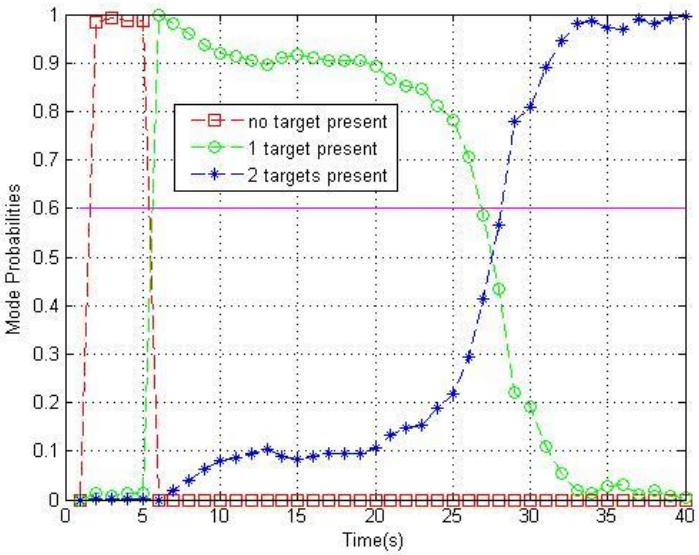


Figure 5.69. The mode probabilities for 500 particles (First Scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

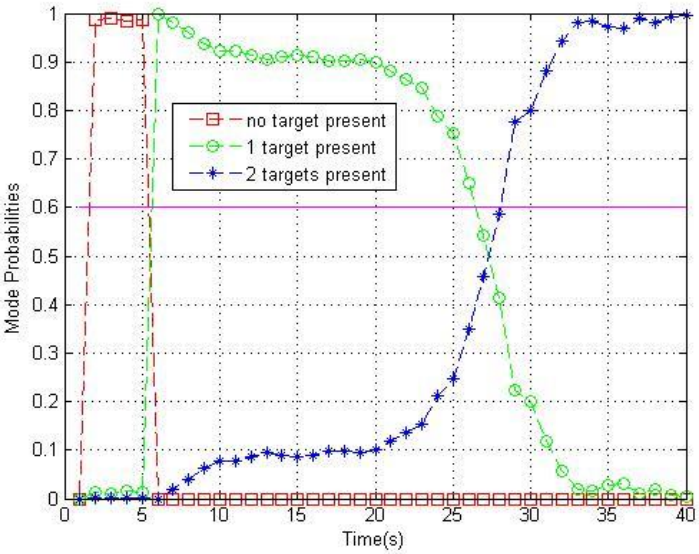


Figure 5.70. The mode probabilities for 1k particles (First Scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

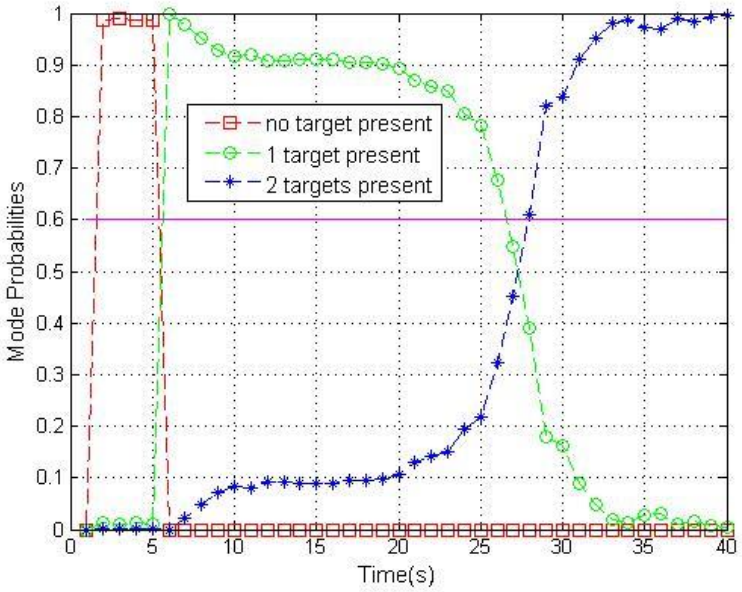


Figure 5.71. The mode probabilities for 5k particles (First Scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

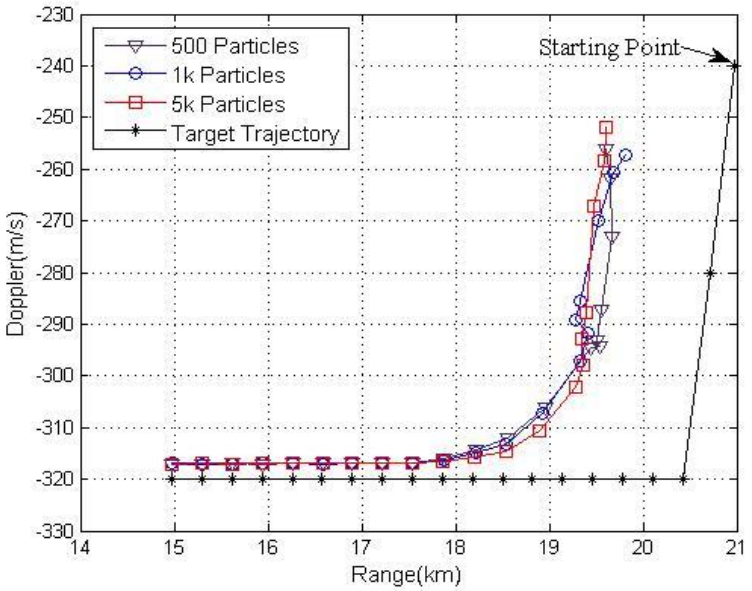


Figure 5.72. Range versus Doppler estimates for the spawned target and its trajectory (First Scenario, 6 dB initial SNR of the spawned target)

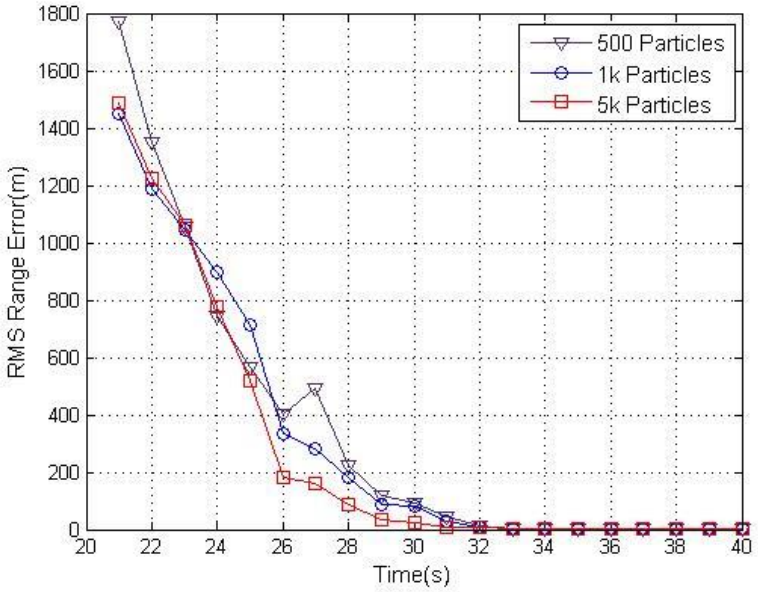


Figure 5.73. The RMS range error for the spawned target (First Scenario, 6 dB initial SNR of the spawned target)

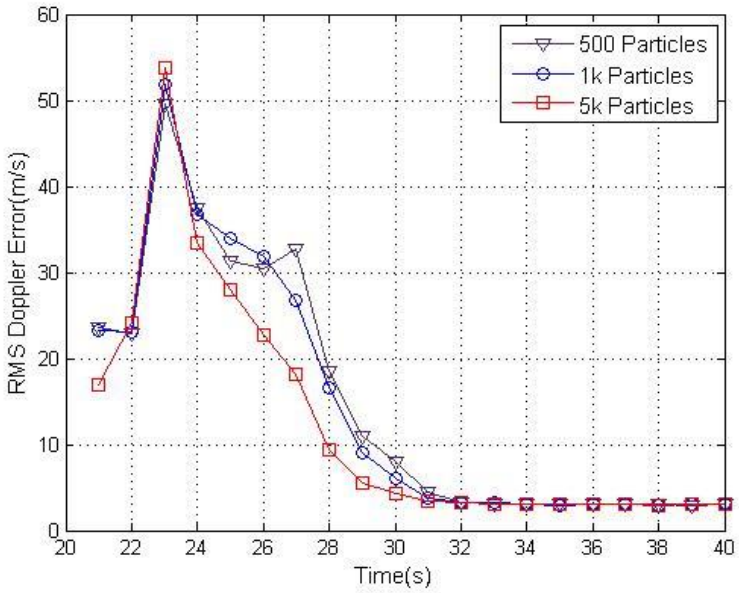


Figure 5.74. The RMS Doppler error for the spawned target (First Scenario, 6 dB initial SNR of the spawned target)

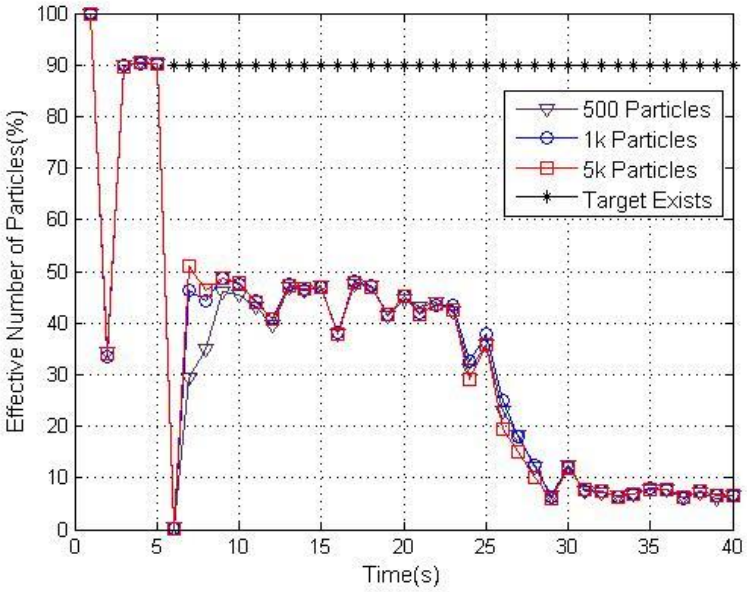


Figure 5.75. N_{eff}/N (First Scenario, 6 dB initial SNR of the weak target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

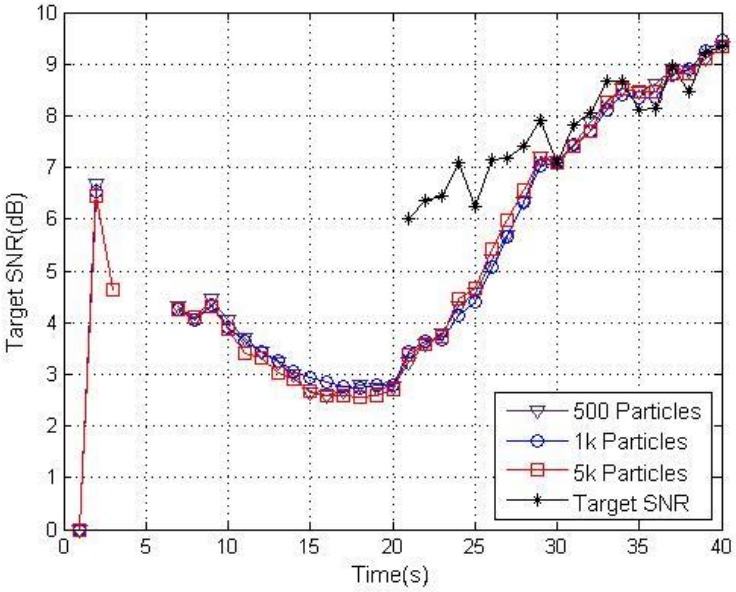


Figure 5.76. The SNR estimates for the spawned target (First Scenario, 6 dB initial SNR of the spawned target)

5.3.2.1.4 Initial SNR of the Spawned Target: 4 dB

The initial SNR of the spawned target is again decreased to the value of 4dB to determine the minimum SNR value of the spawned target at which Algorithm 2 is successful at detection and tracking of both targets. As it can be seen in Figure 5.77, Algorithm 2 is not successful at detecting the spawned target with 4dB initial SNR.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

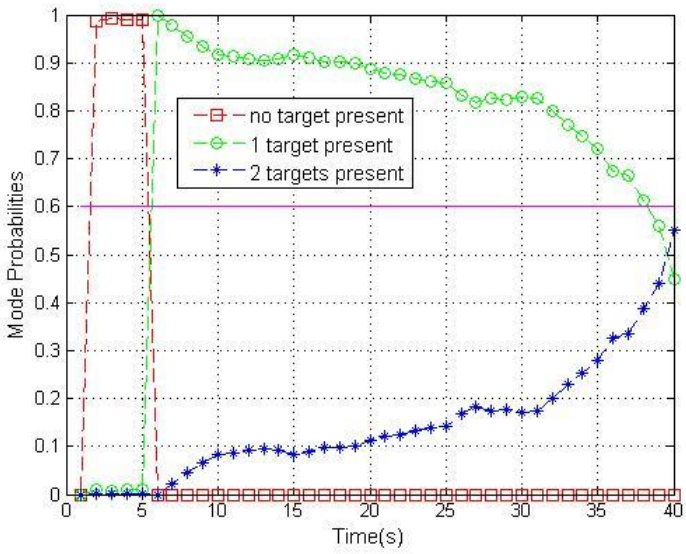


Figure 5.77. The mode probabilities for 5k particles (First Scenario, 4 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

5.3.2.2 Simulation Results for the Second Scenario

As mentioned earlier, the second scenario is used to analyze the performance of the algorithms for the maneuvering targets. The spawned target maneuvers with $+3g$ between $t = 28\ s$ and $t = 32\ s$, $-3g$ between $t = 33\ s$ and $t = 37\ s$ in the second scenario. The trajectories of the targets are shown in Figure 5.2.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.78, Figure 5.79 and Figure 5.80 show the mode probabilities for 500, 1k and 5k particles, respectively. As seen in figures below, there is no change in the probability of existence during the maneuvers of the spawned target. This means that Algorithm 2 can deal with the maneuvers of the spawned target.

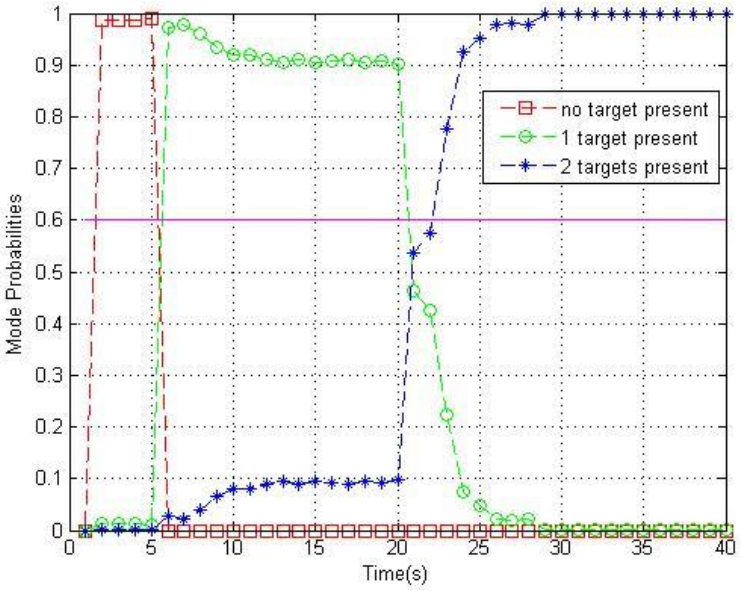


Figure 5.78. The mode probabilities for 500 particles (Second Scenario, 10 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

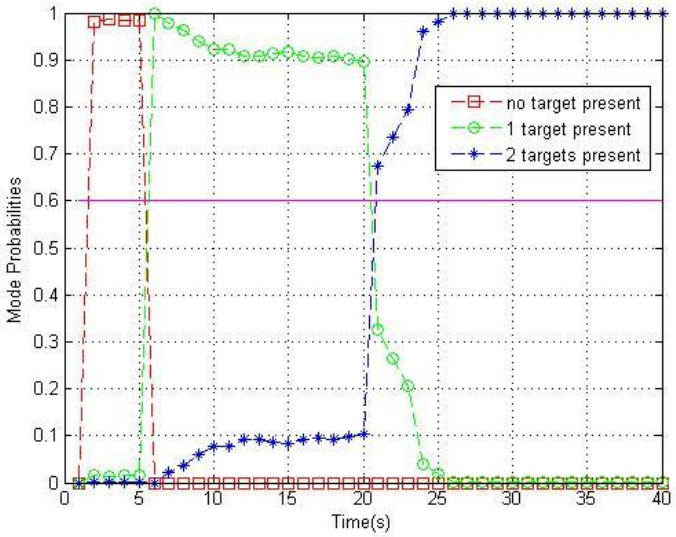


Figure 5.79. The mode probabilities for 1k particles (Second Scenario, 10 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

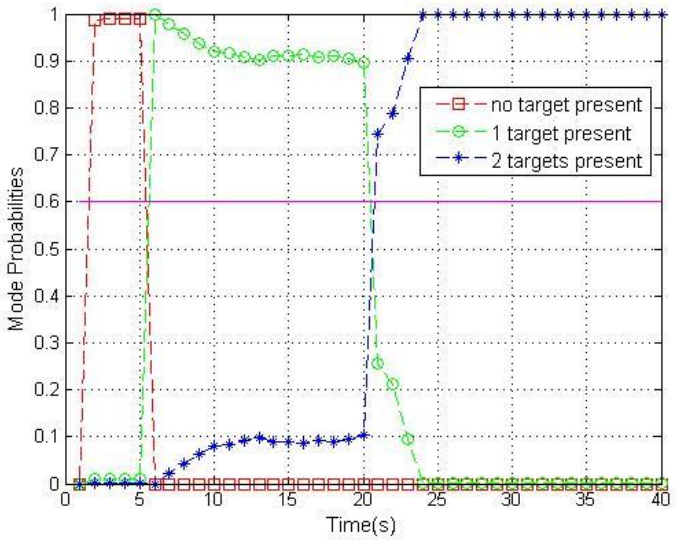


Figure 5.80. The mode probabilities for 5k particles (Second Scenario, 10 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

Figure 5.81 shows the range versus Doppler estimates and the actual positions of the spawned target.

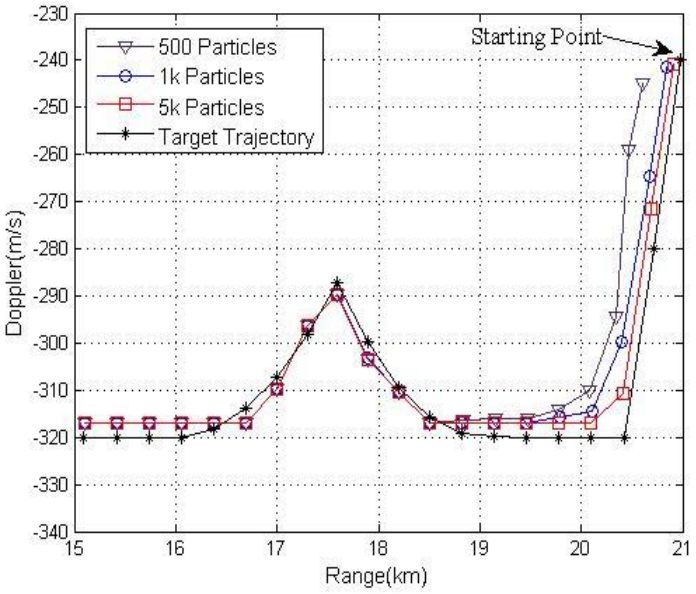


Figure 5.81. Range versus Doppler estimates for the spawned target and its trajectory (Second Scenario, 10 dB initial SNR of the spawned target)

The RMS range and Doppler errors for the spawned target are given in Figure 5.82 and Figure 5.83, respectively. As seen in figures below, the accuracies of the estimates don't decrease with the maneuvers of the spawned target.

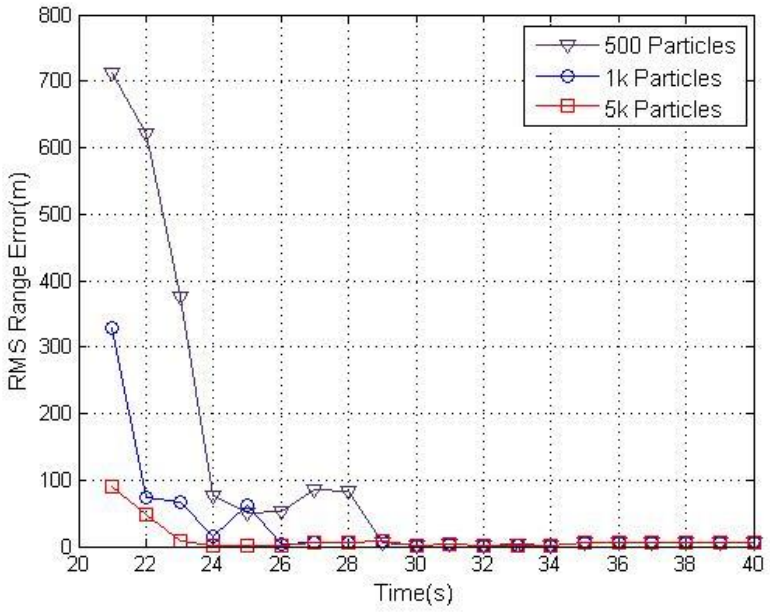


Figure 5.82. The RMS range error for the spawned target (Second Scenario, 10 dB initial SNR of the spawned target)

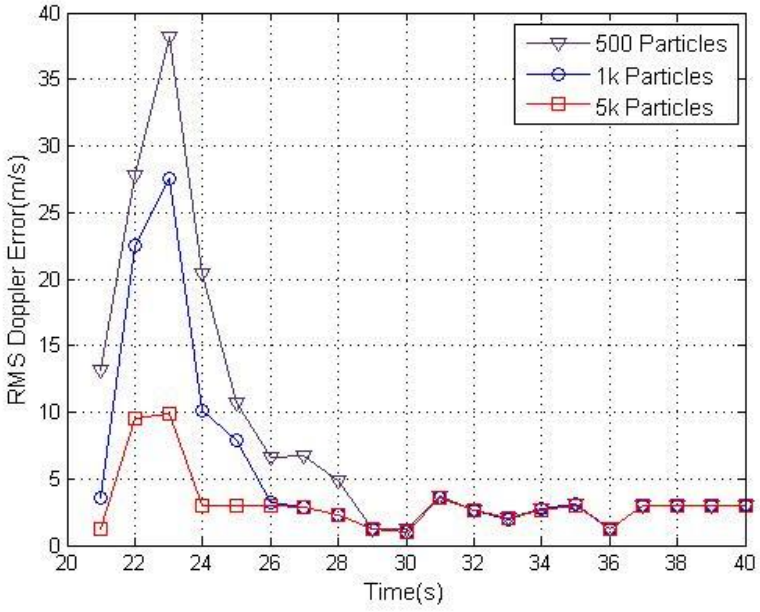


Figure 5.83. The RMS Doppler error for the spawned target (Second Scenario, 10 dB initial SNR of the spawned target)

Figure 5.84 shows the ratio of N_{eff}/N for the spawned target. As it can be seen in Figure 5.84, the ratio of N_{eff}/N decreases to as low as % 3 between $t = 30\ s$ and $t = 35\ s$. This means that the sample impoverishment problem becomes so serious at time steps when the spawned target maneuvers although the detection performance and the accuracies of the estimates don't decrease.

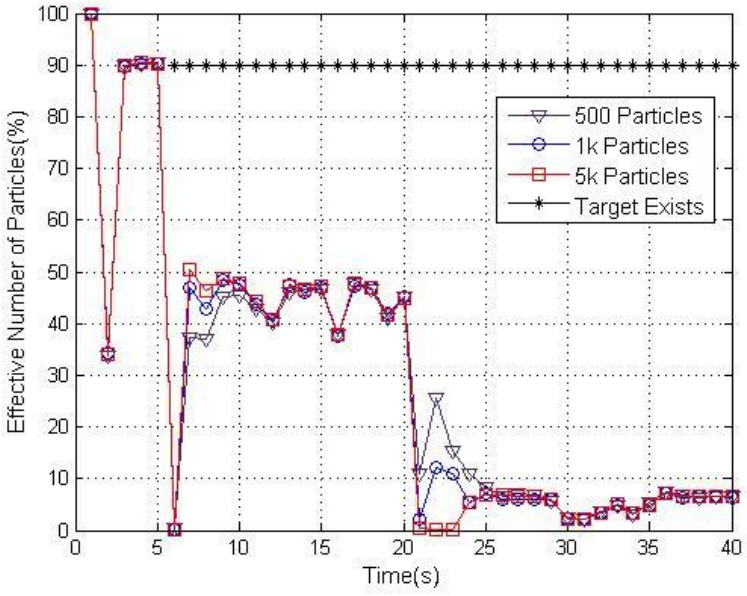


Figure 5.84. N_{eff}/N (Second Scenario, 10 dB initial SNR of the weak target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

Figure 5.85 shows that the SNR estimates for the spawned target and its actual SNR values. It can be seen that the accuracies of the SNR estimates also don't decrease with the maneuvers of the spawned target.

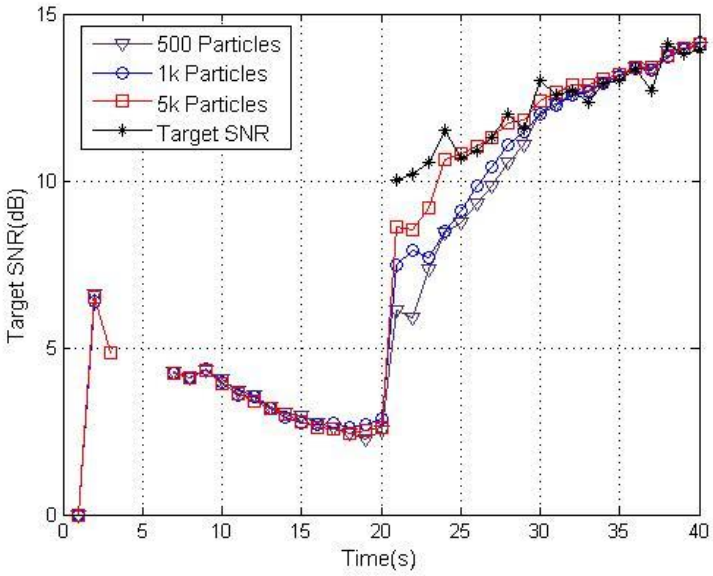


Figure 5.85. The SNR estimates for the spawned target (Second Scenario, 10 dB initial SNR of the spawned target)

5.3.2.3 Simulation Results for the Third Scenario

As mentioned earlier, we want to simulate a highly maneuvering spawned target in the third scenario since the spawned targets may have high maneuvers. Figure 5.3 shows the trajectories of the targets. Note that the spawned target maneuvers with $-7g$ between $t = 28\ s$ and $t = 32\ s$, $+7g$ between $t = 33\ s$ and $t = 37\ s$ in this scenario.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.86 shows the probability of the spawned target's existences obtained by using 10k particles in Algorithm 2. As seen in Figure 5.86, there is an obvious decrease in the probability of the spawned target's existence between $t = 29\ s$ and $t = 38\ s$ because of the high maneuvers. In fact, Algorithm 2 can't detect the spawned target between $t = 32\ s$ and $t = 35\ s$. Figure 5.87 shows the probability of the spawned target's existence obtained by using Algorithm 2 with process

noise identification. As seen in Figure 5.87, there is no decrease in the probability of target existence during the maneuvers of the spawned target. Therefore, it can be said that the proposed process noise identification method can deal with the problem mentioned above as it is the same for Algorithm 1.

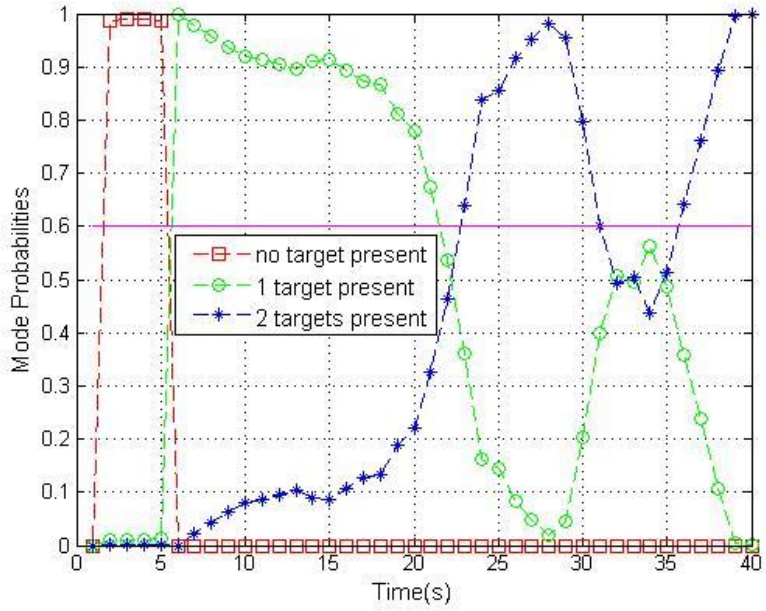


Figure 5.86. The probability of existence for the spawned target and its SNR value (Third scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

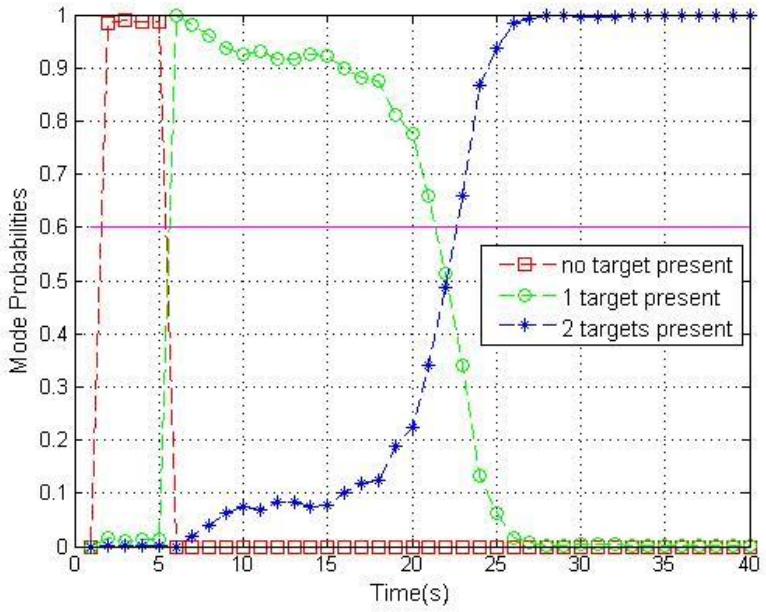


Figure 5.87. Actual SNR values and the probability of existence for the spawned target obtained by using Algorithm 2 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existences.

Figure 5.88 and Figure 5.89 show the range versus Doppler estimates for the spawned target obtained by using Algorithm 2 and Algorithm 2 with process noise identification, respectively. It is obvious that the proposed process noise identification method increases the tracking performance of Algorithm 2 as it is the same for Algorithm 1.

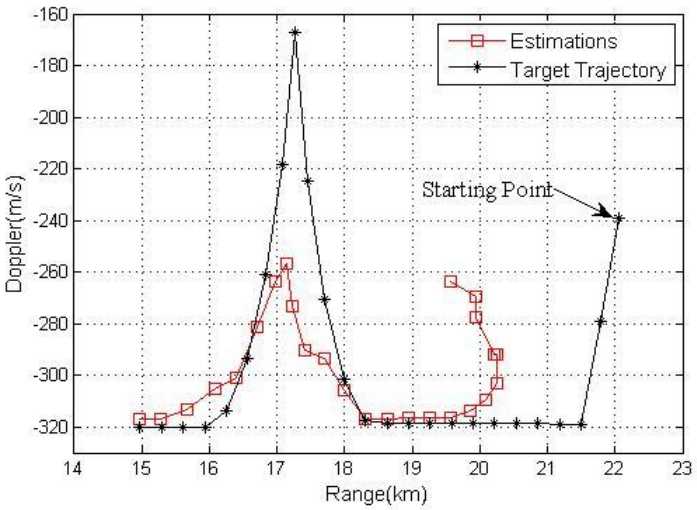


Figure 5.88. Range versus Doppler estimates for the spawned target and its trajectory (Third scenario, 6 dB initial SNR of the spawned target)

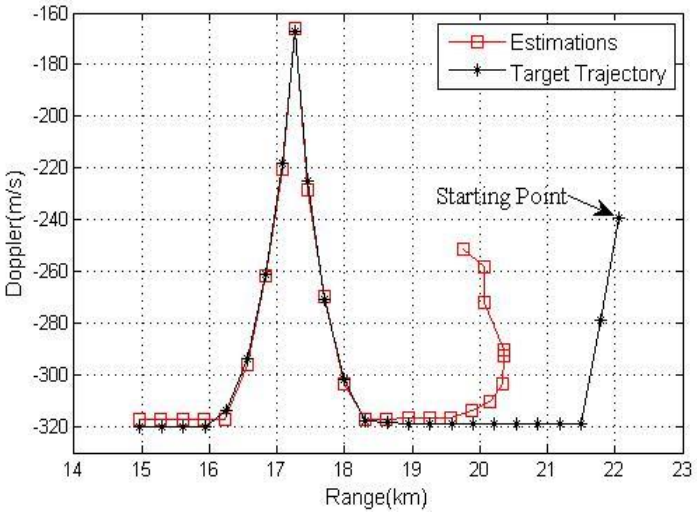


Figure 5.89. Actual trajectory and range versus Doppler estimates for the spawned target obtained by using Algorithm 2 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target)

Figure 5.90 and Figure 5.91 show the RMS range errors for the spawned target obtained by using Algorithm 2 and Algorithm 2 with process noise identification, respectively.

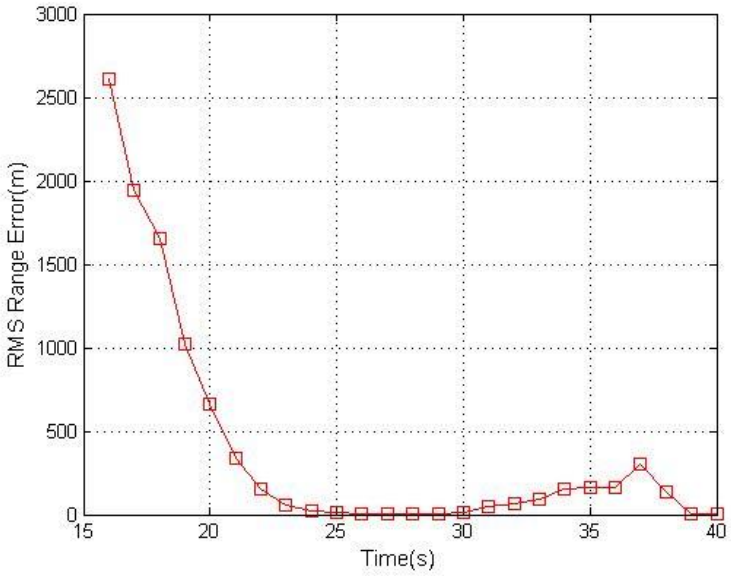


Figure 5.90. The RMS range error for the spawned target (Third scenario, 6 dB initial SNR of the spawned target)

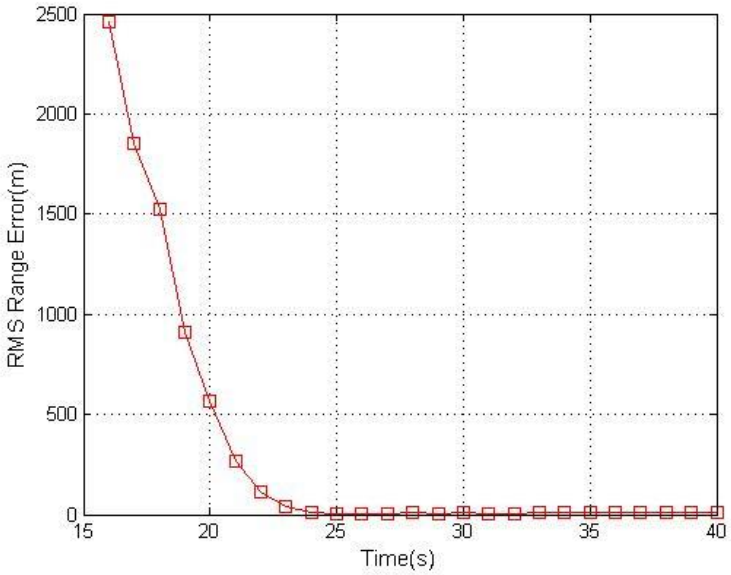


Figure 5.91. The RMS range error for the spawned target obtained by using Algorithm 2 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target)

Figure 5.92 and Figure 5.93 show the RMS Doppler errors for the spawned target obtained by using Algorithm 2 and Algorithm 2 with process noise identification, respectively.

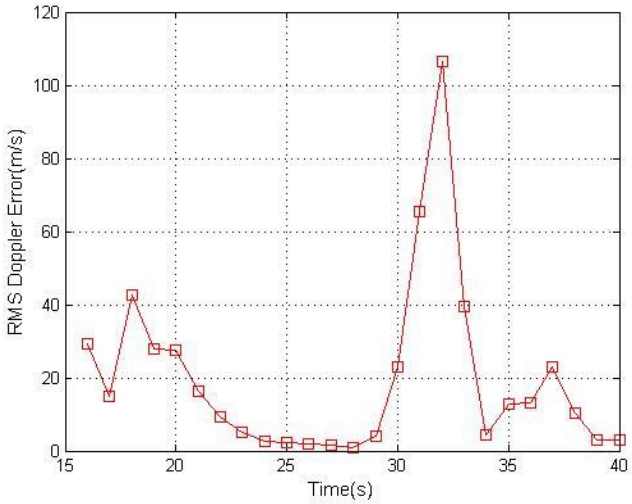


Figure 5.92. The RMS Doppler error for the spawned target (Third scenario, 6 dB initial SNR of the spawned target)

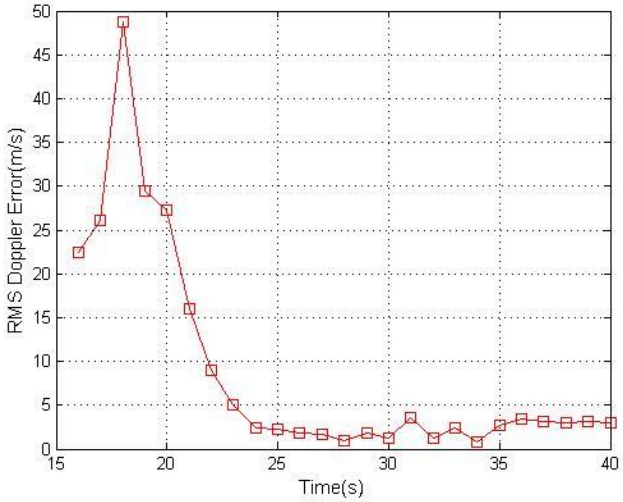


Figure 5.93. The RMS Doppler error for the spawned target obtained by using Algorithm 2 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target)

As seen in figures above, the RMS errors increase with the maneuvers of the spawned target in Algorithm 2; whereas, they don't increase in Algorithm 1 with process noise identification.

Figure 5.94 and Figure 5.95 show the ratio of N_{eff}/N for the spawned target obtained by using Algorithm 2 and Algorithm 2 with process noise identification, respectively.

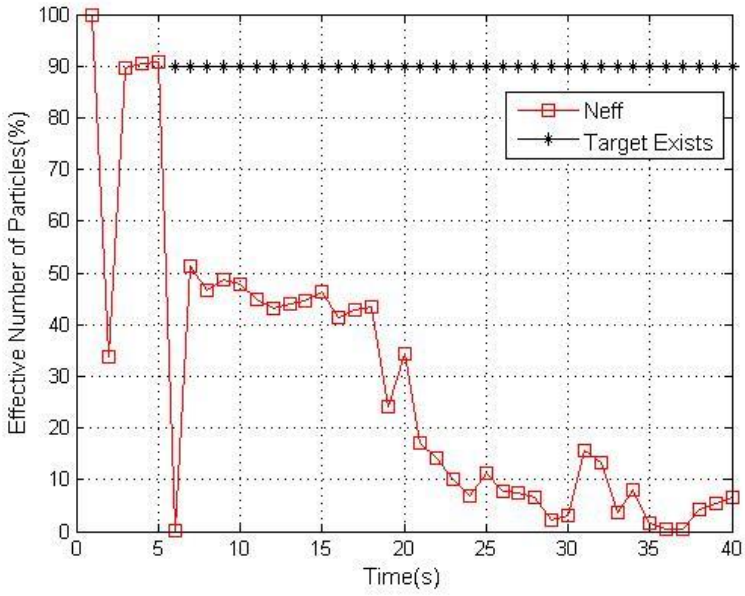


Figure 5.94. N_{eff}/N for the spawned target (Third scenario, 6 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

Figure 5.94 shows that the sample impoverishment problem is so serious for Algorithm 2. However, the sample impoverishment problem is solved by the process noise identification method as seen in Figure 5.95.

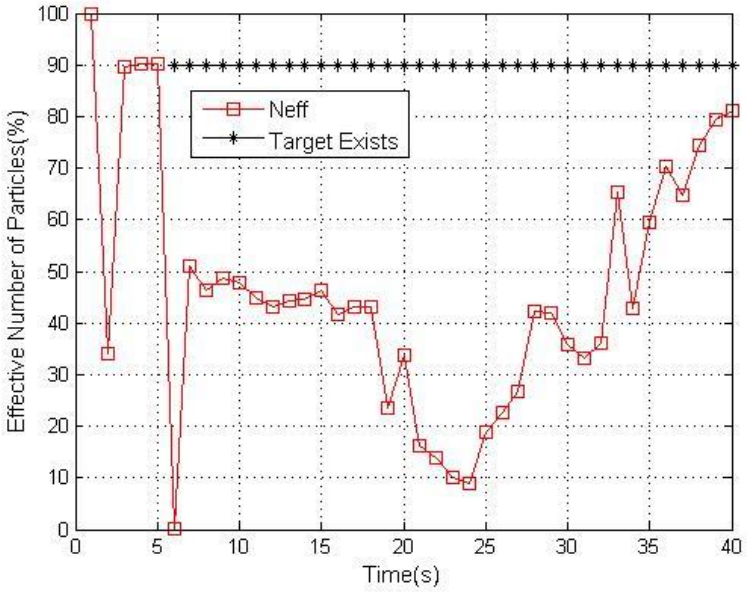


Figure 5.95. N_{eff}/N for the spawned target obtained by using Algorithm 2 with process noise identification (Third scenario, 6 dB initial SNR of the spawned target). The existence of the target is indicated by '*'. For visual clarity, it is shown as a line at $N_{eff}/N = 90\%$.

5.3.2.4 Simulation Results for the Fourth Scenario

As mentioned earlier, in the fourth scenario, Swerling-0 type of targets which move very slowly are used in order to see the detection times independently from the changes in the SNR values of the targets. The mean SNR values change very slightly during the fourth scenario since the targets move with very low velocities. Furthermore, there is no target SNR fluctuations since Swerling-0 type of targets are used in this scenario. Initial SNR of the spawned target is 6 dB and the SNR value never exceeds 6.4 dB during the scenario.

Note that 35 Monte Carlo simulations are performed to obtain each result given below.

Figure 5.96, Figure 5.97 and Figure 5.98 show the probability of the spawned target's existence for different number of particles. As it can be seen in the figures, the existence of the spawned target is declared in 9 seconds for 500 particles, 8 seconds for 1k and 5k particles. As mentioned in Section 5.3.2.1.3, in the first scenario in which Swerling-1 type of targets are used and the initial SNR of the spawned target is 6 dB, the existence of the spawned target is declared in 9 seconds for 500 particles and 1k particles, 8 seconds for 5k particles. In the fourth scenario, the detection time decreases for 1k particles compared to the first scenario. The reason is that there is no target SNR fluctuations since Swerling-0 type of targets are used in the fourth scenario.

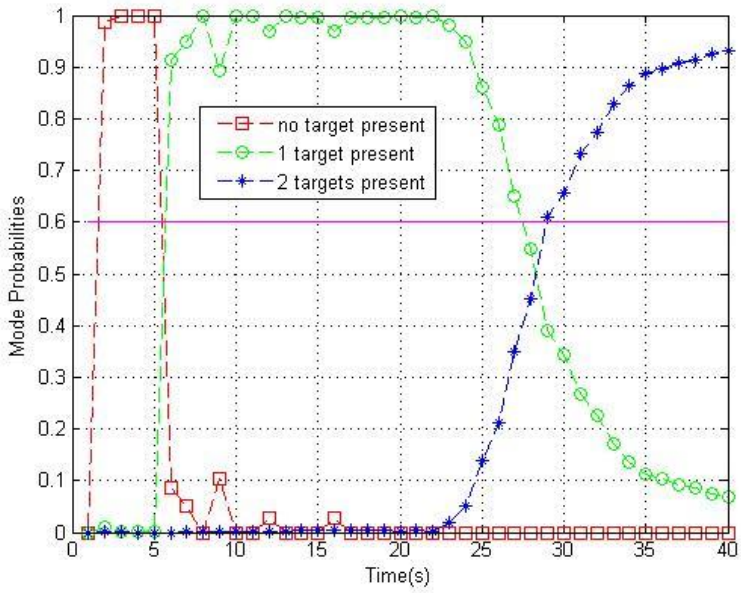


Figure 5.96. The mode probabilities for 500 particles (Fourth Scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

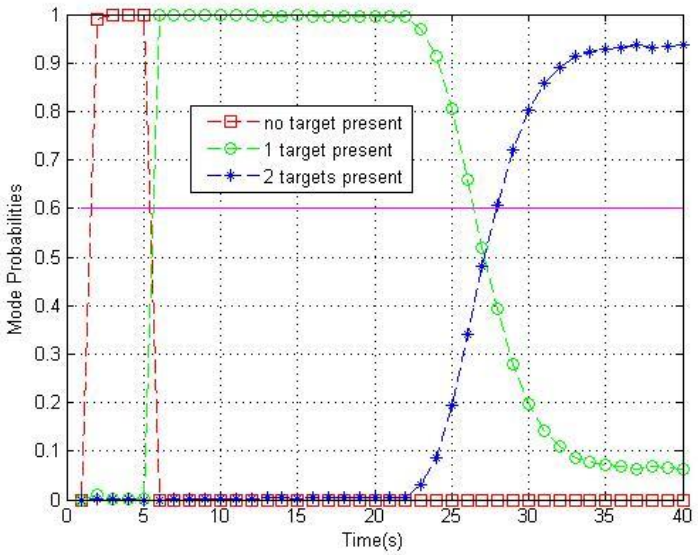


Figure 5.97. The mode probabilities for 1k particles (Fourth Scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

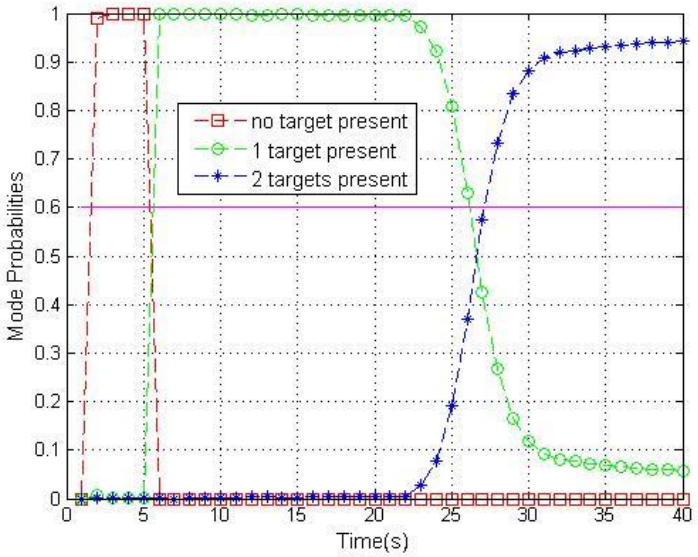


Figure 5.98. The mode probabilities for 5k particles (Fourth Scenario, 6 dB initial SNR of the spawned target). The line at $p = 0.6$ indicates the threshold for the declaration of target existence.

It can be concluded from the results given above that Algorithm 2 is successful at detection and tracking of the spawned targets with SNR values as low as 6 dB.

5.3.2.5 Summary of the Results of Algorithm 2

The detection and tracking performance of Algorithm 2 is analyzed for different scenarios. The target declaration times for different initial target SNR and different number of particles are shown for the spawned target in Table 5.5. The SNR value of the spawned target at the declaration of its existence is also shown in parenthesis.

Table 5.5. Target declaration times for different initial SNR values of the spawned target and different number of particles

	10 dB	8 dB	6 dB
500 Particles	3 sec (at 10.5 dB)	5 sec (at 9.35 dB)	9 sec (at 7.91 dB)
1k Particles	1 sec (at 10 dB)	4 sec (at 8.51 dB)	9 sec (at 7.91 dB)
5k Particles	1 sec (at 10 dB)	4 sec (at 8.51 dB)	8 sec (at 7.41 dB)

It is shown that Algorithm 2 is successful at target detection for initial target SNR as low as 6dB. In Table 5.5, it can be seen that the target declaration times increase as the target SNR decreases; whereas, increasing number of particles reduces the target declaration times as expected.

The performance of Algorithm 2 is also analyzed on the maneuvering targets by using the second and third scenarios. As shown by using the second scenario, Algorithm 2 is successful at detection and tracking of a maneuvering target with $+3g$. Furthermore, it is shown that the proposed process noise identification method is very successful at increasing the performance of Algorithm 2. It enables Algorithm 2 to track and detect a highly maneuvering target without suffering the sample impoverishment problem.

5.3.3 Comparisons between Algorithms

Algorithm 2 can be said to be more successful than Algorithm 1 at detection of the spawned target according to the declaration times in Table 5.4 and Table 5.5. However, Algorithm 2 suffers from the sample impoverishment problem more than Algorithm 1 when the existence of the spawned target is declared as seen in figures which represent the ratio of N_{eff}/N in the previous sections.

As mentioned earlier, the proposed process noise identification method provides significant increase in the tracking and detection performance of both algorithms. The sample impoverishment problem in both algorithms is solved by using this method. Furthermore, it gives a chance to track a highly maneuvering target efficiently.

The execution times of the proposed algorithms are given in Table 5.6. The execution time is the average CPU time needed to execute all time steps in MATLAB 7.13 on a 2.4 GHz Intel Core i7 operating under Windows 8. They are determined by using the first scenario in which the initial SNR of the spawned target is 6 dB. As it can be seen in Table 5.6, the execution time of Algorithm 1 is slightly less than the execution time of Algorithm 2 although Algorithm 2 has a better detection performance than Algorithm 1. Moreover, the addition of the property of process noise identification brings with more computational load as expected.

Table 5.6. Execution times of algorithms when using the first scenario and the initial SNR of the spawned target is 6 dB

	Algorithm 1	Algorithm 1 with Process Noise Identification	Algorithm 2	Algorithm 2 with Process Noise Identification
Execution Time	27.3921 sec	32.556 sec	28.9640 sec	34.1680 sec

CHAPTER 6

CONCLUSIONS

6.1 Summary and Conclusions

In this thesis, particle filters are proposed to develop TBD approaches. Two particle filter based TBD algorithms are proposed to solve the problem of detection and tracking of the spawning targets.

In contrast to the works in literature, a novel reduced order dynamic model is introduced to improve the efficiency of the particle filter. Furthermore, the goal is also to estimate the SNR of the targets. It is also shown that it can be achieved by modifying the state model by adding the target SNR as a state. The algorithms proposed in this work can deal with the target SNR fluctuations according to Swerling-1 model; whereas, in most of the related works in literature, target SNR is assumed to be known.

It is shown that both of the algorithms can deal with the scenario of the maneuvering targets. Moreover, a new process noise identification method [1] proposed for the classical target tracking methods is adapted to the TBD framework in order to deal with the maneuvering target problem. It also has a positive effect on the reduction of the sample impoverishment problem which is serious for tracking of the highly maneuvering targets by particle filters. It is shown that the proposed process noise identification method provides significant increase in the tracking and detection performance of both algorithms.

The contributions of this thesis can be summarized as follows.

- A new reduced order state space model makes the particle filter applications more efficient.
- An adaptation of a new process noise covariance matrix estimation method to the TBD algorithm.

6.2 Future Studies

Some suggested topics for future studies are given as follows.

a) An improvement can be done for the scenario in which the maximum number of possible targets is unknown and more than two.

b) An improvement can be done for the extended main platforms.

c) Particle filters with different proposal densities can be used in the TBD approaches.

REFERENCES

[1] Jing, L., ChongZhao, H. and Vadakkepat, P., *Process noise identification based particle filter: an efficient method to track highly manoeuvring targets*, IET Signal Process, vol. 5, pp.538-546, 2011.

[2] Verschure, F., *Multiple Modal Particle Filters for Track Before Detect*, M.S. thesis, Dept. of Electrical Engineering, Eindhoven University of Technology, 2003.

[3] Carlson, B.D., Evans, E.D. and Wilson, S.L., *Search radar detection and track with the Hough Transform,* IEEE Transactions on Aerospace and Electronic Systems, vol. 30, no. 1, pp. 109–115, 1994.

[4] Moyer, Lee. R., Spak, J. and Lamanna P., *A Multi-Dimensional Hough Transform-Based Track-Before-Detect Technique for Detecting Weak Targets in Strong Clutter Backgrounds*, IEEE Transactions on Aerospace and Electronic Systems, vol. 47, no. 4, pp. 3062–3068, 2011.

[5] Smith, M.C and Winter, E.M., *On the Detection of Target Trajectories in a Multi Target Environment*, IEEE Conference on Decision and Control, San Diego, California, pp. 1189–1194, 1978.

[6] Smith, M.C, *Feature Space Transform for Multitarget Detection*, IEEE Conference on Decision and Control, Alburquerque, New Mexico, pp. 835-836, 1980.

[7] Johnston, L.A. and Krishnamurthy, V., *Performance Analysis of a Dynamic Programming Track Before Detect Algorithm*, IEEE Transactions on Aerospace and Electronic Systems, vol. 38, no.1, pp. 228–242, 2002.

[8] Davey, Samuel J., Rutten, Mark G. and Cheung, B., *A Comparison of Detection Performance for Several Track-before-Detect Algorithms*, Hindawi Publishing Corporation, EURASIP Journal on Advances in Signal Processing, vol. 2008, 2007.

[9] Tonissen, S.M. and Evans, R.J., *Performance of Dynamic Programming Techniques for Track-Before-Detect*, IEEE Transactions on Aerospace and Electronic Systems, vol. 32, no.4, pp. 1440–1451, 1996.

[10] Bruno, M.G.S and Moura, J.M.F, *Multiframe detector/tracker: Optimal performance*, IEEE tran. Aerosp. Electron. Syst., 37(3), 925-945, 2001.

[11] Sabuncu, M., *Particle Filter Based Track Before Detect Algorithm for Tracking of Dim Moving Targets*, M.S. thesis, Dept. Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, 2012.

[12] Boers, Y. and Driessen, J.N., *Multitarget Particle Filter Track Before Detect Application*, IEEE Proceedings – Radar, Sonar and Navigation, 2004.

[13] Mercier, D. E., *An Extended Kalman Filter for Use in a Shared Aperture Medium Range Tracker*, M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB, 1978.

[14] Maybeck, P.S and Mercier, D.E., *A Target Tracker Using Spatially Distributed Infrared Measurements*, IEEE Transactions on Automatic Control, vol. 25, no. 2, pp. 222-225, 1980.

[15] Barniv, Y., *Dynamic Programming Solution for Detecting Dim Moving Targets*, IEEE Transactions on Aerospace and Electronic Systems, vol. 21, no. 1, pp. 144-156, 1985.

[16] Barniv, Y., *Dynamic Programming Solution for Detecting Dim Moving Targets. Part II: Analysis*, IEEE Transactions on Aerospace and Electronic Systems, vol. 23, no. 6, pp. 144-156, 1987.

[17] Barniv, Y., *Dynamic Programming Solution for Detecting Dim Moving Targets, in: Multitarget-Multisensor Tracking*: Advanced Applications, Artech House, Norwood, MA, Y. Bar-Shalom (Ed.), Chapter 4, 1990.

[18] Streit, R.L., *Tracking on Intensity-Modulated Data Streams*, Technical Report 11221, NUWC, Newport, Rhode Island, USA, 2000.

[19] Streit, R.L., Graham, M.L. and Walsh, M.J., *Multitarget Tracking of Distributed Targets Using Histogram-PMHT*, Digital Signal Processing, vol. 12, no. 2, pp. 394-404, 2002.

[20] Bruno, M.G.S., *Bayesian Methods for Multiaspect Target Tracking in Image Sequences*, IEEE Transactions on Aerospace and Electronic Systems, vol. 52, no. 7, pp.1848–1861, 2004.

[21] Salmond, D.J. and Birch, H., *A Particle Filter for Track-Before-Detect*, Proceedings of the American Control Conference, Arlington, VA, USA, pp. 3755–3760, 2001.

[22] Boers, Y. and Driessen, J.N., *Particle Filter Based Detection for Tracking*, Proceedings of the American Control Conference, Arlington, VA, USA, pp. 4393–4397, 2001.

[23] Richards, Mark A., *Fundamentals of Radar Signal Processing*, McGraw-Hill, 2005.

[24] Jazwinski, A.H., *Stochastic Processes and Filtering Theory*, New York: Academic Press, 1970.

[25] Julier, S.J. and Uhlmann, J.K., *Unscented Filtering and Nonlinear Estimations*, Proceedings of the IEEE, vol. 92, no. 3, 2004.

[26] Davis, P.J. and Rabinowitz, P., *Methods of Numerical Integration*, New York: Academic Press, 1984.

[27] Ristic, B., Arulampalam, S. and Gordon, N., *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House, 2004.

[28] Doucet, A., Godsill, S. and Andrieu, C., *On Sequential Monte Carlo Sampling Methods for Bayesian Filtering*, Kluwer Academic, 2000.

[29] Crisan, D. and Doucet, A., *A Survey of Convergence Results on Particle Filtering Methods*, IEEE Transactions on Signal Processing, vol. 50, no 3., pp. 736-746, 2002.

[30] Kong, A., Liu, J.S. and Wong, W.H., *Sequential Imputations and Bayesian Missing Data Problems*, Journal of the American Statistical Association, vol. 89, no. 425, pp. 278-288, 1994.

[31] Liu, J.S. and Chen, R., *Sequential Monte Carlo Methods for Dynamical Systems*, Journal of the American Statistical Association, vol. 93, pp. 1032-1044, 1998.

[32] Boers, Y. and Driessen, H., *Particle Filter Based Track Before Detect Algorithms*, Proceedings of SPIE – Signal and Data Processing of Small Targets, 2003.

[33] Skolnik, M.I., *Introduction to Radar Systems*, McGraw-Hill, Inc, Singapore, 2nd edn., 1981.

[34] Liu, J. and West, M., *Combined parameter and state estimation in simulation-based filtering*, In A. Doucet, N. De Freitas and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, Chapter 10, Springer, New York, 2001.

[35] Storvik, G., *Particle filters for state-space models with the presence of unknown static parameters*, IEEE Transactions on Signal Processing, 50(2):281-289, February 2002.

[36] Saha, S., Özkan, E., Gustafsson, F. and Šmídl, V., *Marginalized particle filters for Bayesian estimation of Gaussian noise parameters*, In Proceedings of 13th International Conference on Information Fusion, July 2010.

[37] Özkan, E., Šmídl, V., Saha, S., Lundquist, C. and Gustafsson, F., *Marginalized adaptive particle filtering for nonlinear models with unknown time-varying noise parameters*, Automatica, vol. 49, pp. 1566-1575, 2013.

[38] Koç, S., *Aselsan – ODTÜ Çalışmaları Radar Simülatörü*, Technical Report, ASELSAN A.Ş., 2005.

[39] Swerling, P., *Radar Probability of Detection for Some Additional Fluctuating Target Cases*, IEEE Transactions on Aerospace and Electronic Systems, 1997.

[40] Gabrielová, M., *Dynamic Model of Spatial Motion of Missile*, Military Academy in Brno, 2004.