Particle Filtering for Track Before Detect Applications

Master's Thesis
Division of Automatic Control
Department of Electrical Engineering
Linköping University, Sweden

Johan Torstensson
Mikael Trieb

Reg no: LiTH-ISY-EX-3683
Linköping 2005

Particle Filtering for Track Before Detect Applications

Master's Thesis
Division of Automatic Control
Department of Electrical Engineering
Linköping University, Sweden

Johan Torstensson
Mikael Trieb

Reg no: LiTH-ISY-EX-3683

Supervisors: **Dr.Ir. Y. Boers THALES NL**
Dr.Ir. J.N Driessen THALES NL
Dr. R. Karlsson

Examiner: **Prof. Fredrik Gustafsson**

Linköping 31st August 2005.

Abstract

Integrated tracking and detection, based on unthresholded measurements, also referred to as *track before detect* (TBD) is a hard nonlinear and non-Gaussian dynamical estimation and detection problem, see e.g., [3] for a good and recent overview. However, it is a technique that enables the user to track and detect targets that would be extremely hard to track and detect, if possible at all with "classical" methods. TBD enables us to be better able to detect and track weak, stealthy or dim targets in noise and clutter and particles filter have shown to be very useful in the implementation of TBD algorithms. This Master's thesis has investigated the use of particle filters on radar measurements, in a TBD approach. The work has been divided into two major problems, a time efficient implementation and new functional features, as estimating the *radar cross section* (RCS) and the extension of the target. The later is of great importance when the resolution of the radar is such, that specific features of the target can be distinguished.
Results will be illustrated by means of realistic examples.

Keywords: Particle Filter, Target Tracking, Track Before Detect, TBD, Bayesian Estimation, Monte Carlo Methods, Nonlinear Filtering.

Acknowledgements

This thesis is submitted for the degree of Master of Science in Applied Physics and Electrical Engineering at Linköping University. The research has been carried out at THALES Nederland in Hengelo.

First of all we would like to thank our supervisors at THALES, Yvo Boers and Hans Driessen for excellent guidance and many very interesting discussions along the road. We could not have asked for better support. We would also like to thank our supervisor at ISY, Rickard Karlsson and our examiner Professor Fredrik Gustafsson for valuable comments, reading and reviewing of the report, and for initializing our interest in the research field of particle filters. A big thanks also goes to the JRS group for hosting us during our stay at THALES.

Hengelo, June 2005
Johan Torstensson
Mikael Trieb

Contents

Notation

Operators and functions

a_{maxx}	Maximum acceleration in x direction
a_{maxy}	Maximum acceleration in y direction
b	Bearing
d	Doppler (Doppler Speed)
δ	Delta-Dirac function
$\mathbb{D}$	The whole range-doppler-bearing cell space
f	State equation transition mapping (discrete-time)
F	Linearized state update matrix
h	Measurement relation
h_A	Reflection form, defined for every range-doppler-bearing cell
h_P	Power distribution from a target
H	Linearized measurement relation matrix
H_0	Hypothesis that no target is present
K	Kalman gain
k	Discrete time subscript
m_k	Modal state
μ_t	Expected measurement when $m_k = 1$
μ_v	Expected measurement when $m_k = 0$
n_Q	Zero-mean white Gaussian noise
n_I	Zero-mean white Gaussian noise
N_{eff}	Number of effective particles
N_{th}	Threshold on effective number of particles
$\mathcal{N}(\cdot)$	Gaussian distribution

p_v	Measurement noise probability density
p_w	Process noise probability density
$P_{reveived}$	Power received at the radar
$p(\cdot)$	Probability density function
$Prob$	Probability
$p(z_k\|s_k)$	Likelihood function
$\mathcal{O}(n)$	Ordo operator
$p_{\frac{s_k}{H_0}}(z_k\|s_k)$	Likelihood ratio $\frac{p(z_k\|s_k)}{p(z_k\|H_0)}$
$\pi(s)$	Importance function
Π	Markov transition matrix
q_k	Particle weight
$\mathbb{R}^n$	Euclidean n-dimensional space
r	Range
$\dot{r}$	Range rate (Doppler speed)
σ	Radar Cross Section RCS
σ_n	Noise variance
s_k	State vector
$\hat{s}_{k\|k}$	Estimate (filtering)
$\hat{s}_{k+1\|k}$	Estimate (one step prediction)
S_k^i	Particle, contains
$\breve{s}_k^i$	Resampled particles
$\tilde{s}_k$	Measurements source
$\mathbb{S}$	Subset of cells
$\mathcal{U}(\cdot)$	Uniform distribution
v_k	Measurement noise
w_k	Process noise
z_k	Measurement
Z_k	Set of ordered measurements: $\{z_1, \ldots, z_k\}$
$\sim$	Similar to
$\propto$	Proportional to
$\in$	Belongs to
$\subset$	Subset of

Constants

A	Amplitude of a target
A_e	Effective antenna aperture
B	Constant related to the size of bearing cells
b_{res}	Resolution in bearing
D	Constant related to the size of Doppler cells
d_{res}	Resolution in Doppler
G	Antenna gain
h_{th}	Threshold for h
L_b	Constant of losses
L_d	Constant of losses
L_r	Constant of losses
M	Number of drawn samples to approximate the extended likelihood
N	Number of particles
N_b	Number of bearing cells
N_d	Number of Doppler cells
N_r	Number of range cells
P_b	Probability of birth of a target
P_d	Probability of death of a target
$P_{transmitted}$	Power transmitted by the radar
R	Constant related to the size of range cells
r_{res}	Resolution in range
T	Sample time

Abbreviations

ATC	Air Traffic Control
CSW	Cumulative Sum of Weights
EKF	Extended Kalman Filter
ETPF	Extended Target Particle Filter
i.i.d	Independent Identically Distributed
IS	Importance Sampling
KF	Kalman Filter
MAP	Maximum a Posteriori
ML	Maximum Likelihood
MMSE	Minimum Mean Square Estimate
pdf	Probability Density Function
PF	Particle Filter
RMSE	Root Mean Square Error
RADAR	Radio Detection and Ranging
SIS	Sequential Importance Sampling
SIR	Sampling Importance Resampling
SNR	Signal to Noise Ratio
TBD	Track Before Detect

Chapter 1

Introduction

Many real-world problems involve estimating unknown dynamic quantities on the basis of observed data. In most of these applications the a priori knowledge of the underlying phenomenon can be modeled. These models enable us to apply statistical methods. With such a method it is possible to optimally estimate quantities, based on the observations. Stated otherwise, by using such an approach, the maximum of information possible can be extracted, given our data. The process of extracting information from a (dynamical) system is generally referred to as filtering.

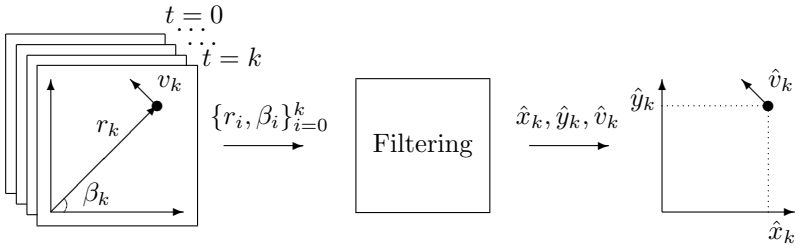
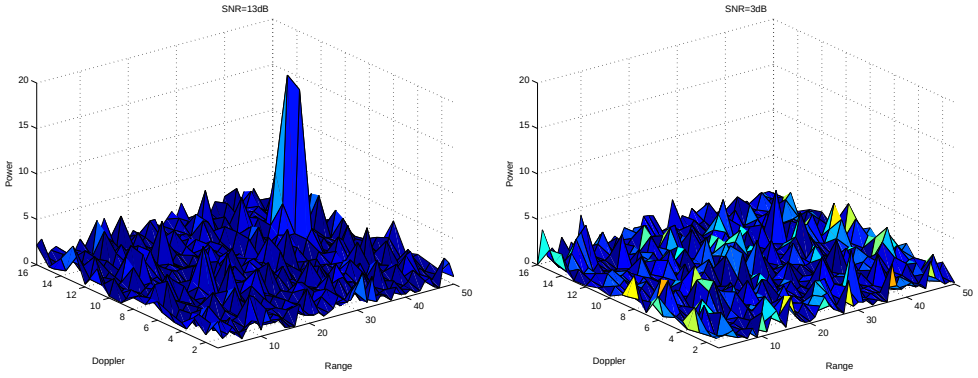


Figure 1.1. An object is moving in the two dimensional space. We want to estimate the position and the velocity of the object. Our measurement consist of the distance r to the target and the angle β. Each time step k we get a new measurement and we want to continuously estimate x, y and v based on these measurements. If we have knowledge about the dynamics of the object and the measurements, this can be modeled and used in the filtering.

Real life examples that fit into this framework are; Tracking a target on the basis of (radar) measurements, estimating a signal in a noisy communication channel, GPS navigation, ship navigation on the basis of (radar) measurements, estimation in mechanically constrained systems, volatility estimation of financial indices on the basis of stock market data, object tracking and location with camera measurements, radar-based distance estimation for collision avoidance (cars) etc.



(a) (b)

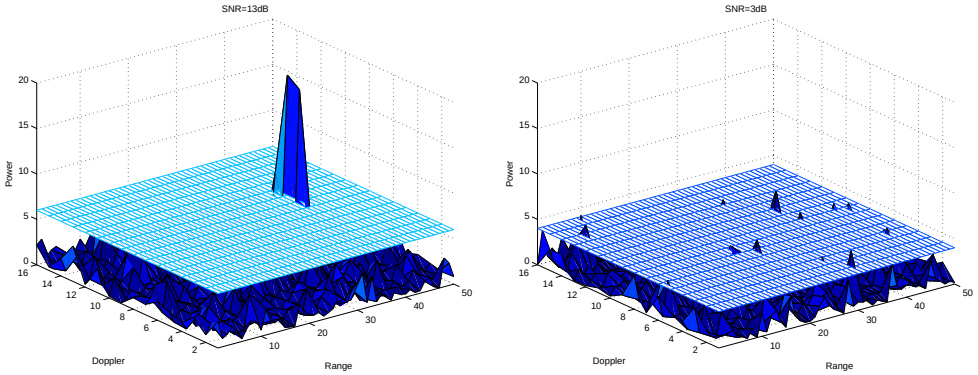
Figure 1.2. Both figures shows the power received by the radar. In a) the SNR is high and it is easy to discern the target as a high peak. In b) the SNR is low and it is not possible to distinguish the target from the noise.

Classical tracking methods take as input plots that typically consists of range-, bearing-, elevation-, and range rate (Doppler) measurements, [1, 2]. In this classical setting, tracking consists of estimating kinematical state properties, e.g., position, velocity and acceleration on the basis of these measurements. In the classical setup the measurements are the output of the extraction, see Figure 2.3. In this setup there is a processing chain before the tracking, which can consist (e.g., in case of radar) of a detection stage, a clustering stage and an extraction stage, Figure 2.3. In the method used in this Master's thesis, the raw measurement data, e.g., reflected power will be used as measurements, see Figure 1.2. This method is referred to as *track before detect* (TBD).

To perform TBD, a particle filter, [3] will be used. A particle filter can, compared to the well known Kalman filter, deal with nonlinear models and non-Gaussian noise. The difference between classical target tracking and TBD will be explained in Chapter 2. A description of a particle filter and the theory behind it will be presented in Chapter 3.

1.1 Problem Formulation

The objective is to investigate and implement a number of improvements on a particle filter for a track before detect application. The different improvements are of such kind, that they can be seen as three independent problems. They will, however, be handled in the order stated below. This because a time efficient particle filter is a necessity to run large simulations on the other two topics within reasonable time.



(a) (b)

Figure 1.3. As an illustration, the two measurements in Figure 1.2 are thresholded. It is easy to see that in the case were the SNR is low, the target might not lead to a detection.

- Implement a time efficient particle filter for a *track before detect application* (TBD) in MATLAB.
- Estimate the *radar cross section* (RCS) of a target, see Section 2.1.2.
- Estimate the extension of a target.

1.2 Restrictions

- No consideration has been taken to the fact that the *signal to noise ratio* (SNR) is range-dependent, $\propto 1/r^4$. This can by motivated by that we only look at target trajectories which are small in comparison to the distance between the sensor and the target.
- In the applications we only consider single target, the models can, however, easily be modified to multiple target models.
- Target trajectories are modeled in two dimensions.
- The coordination system is considered to be fix in the radar and all the estimated parameters are relative to the radar, expressed in these coordinates, see figure 2.1.

1.3 Outline

In Chapter 2 underlying radar theory is presented to give knowledge about how the measurements are obtained. We will also give some background to target tracking and explain the advantages with TBD. Chapter 3 covers the estimation theory with an Bayesian approach in a very brief manner, starting with the Kalman filter and ending with the particle filter. A good tutorial on particle filtering can be found in [24]. In Chapter 4, a time efficient particle filter for a track before detect application is implemented. This is a presumption to be able to run the application with real time feasibility. It will also allow us to do large Monte Carlo simulations within reasonable time. Chapter 5 and Chapter 6 describes how the two estimates of RCS and target extension is made. Simulations and results are presented in the end of each chapter. Finally in Chapter 7 we will summarize the conclusions from Chapter 3 to 6 and discuss future work in this field.

Chapter 2

Radar Theory

Radar is a method to detect, locate, track, and identify distant objects. The principle of Radar (radio detection and ranging) is based on the fact that objects reflect electromagnetic waves. Radar is an active sensor in that way that it is transmitting electromagnetic energy and measure the energy reflected by the target. Because of its all-weather performance and its ability to measure kinematics, the radar has for a long time played a very important role in areas such as *air-traffic-control* (ATC), surveillance, target tracking, ship navigation, and so on. Typical kinematics measured for a specific bearing and elevation are range and range rate (Doppler), see Figure 2.1. For a thorough description of radar, see Skolnik (1980).

2.1 Measurements

Figure 2.1 shows a typical measurement setup for the radar. The range r is the distance between the radar and the target. The elevation angle ϕ is the angle between the horizontal plane and the direction in which the radar points at. The bearing angle θ is the rotation around the z-axis given a reference, here the x-axis.

The transformation from angle and range which the radar measures, to the Cartesian coordinates in which the target dynamics is modeled, brings nonlinearaties into the system. The transformation is given by

$$x = r\cos\theta\cos\phi, \qquad y = r\cos\theta\sin\phi, \qquad z = r\sin\theta \tag{2.1}$$

where ϕ, θ, and r are measured by the radar.
Different types of radar exist, continuous-waves, Doppler etc. The one considered here is of Doppler type. The Doppler-shift gives information about the target speed. The radar transmits a puls of width τ_p, this puls is reflected by the target and is measured by the radar after t seconds. The range to the target is then given by

$$r = \frac{ct}{2}, \tag{2.2}$$

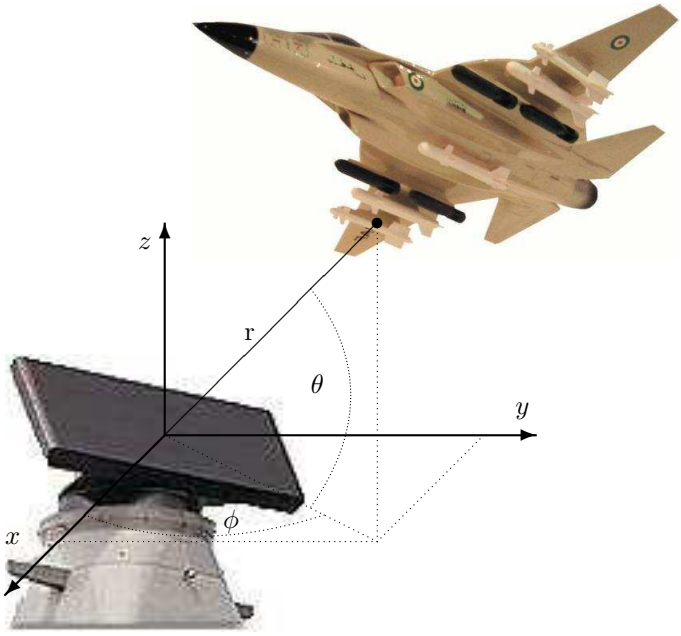


Figure 2.1. Radar Coordinates, relative range r, azimuth θ, and elevation ϕ.

where c is the speed of light and the factor 2 is due to the fact that the pulse has to travel two times r, to and from the target. A practical radar design will check for a returning pulse at a number of discrete ranges, by matched filtering of the returning pulse against a set of discrete delayed versions of the transmitted pulse. These discrete ranges are referred to as bins. A target not centered in a given range bin will have its energy spread across adjacent bins [1]. If the transmitted energy has a wavelength of λ and the radial range speed is $\dot{r}$, the Doppler shift f_d is given by

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

This gives the information about the radial velocity of the target. The Doppler processing is generally performed by computing the discrete Fourier transform of the matched filter samples of a number of returning pulses. In this way the returning energy is separated into a discrete number of doppler bins. As with range, if the Doppler is not centered in one bin, it will spread across adjacent bins. A typical

measurement for a specific bearing and elevation angle, range, and Doppler could look like Figure 2.2.

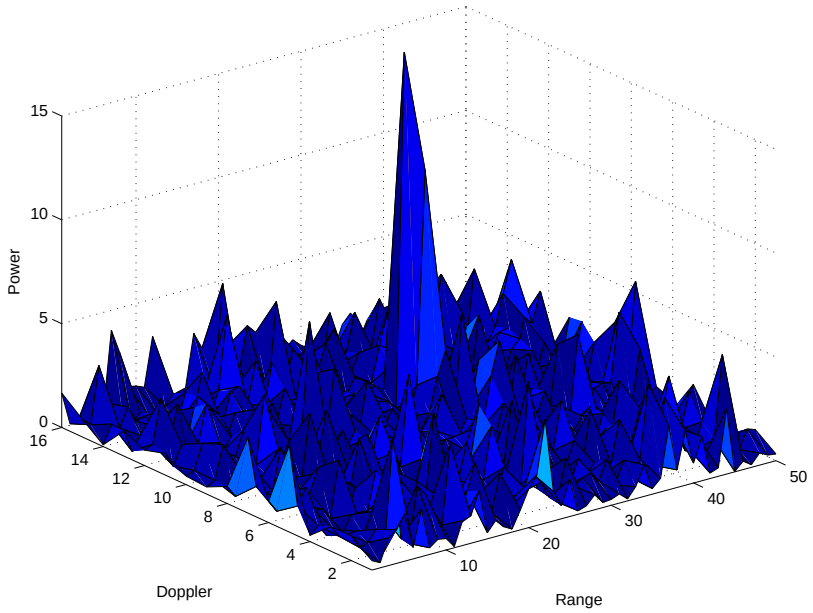


Figure 2.2. A measurement of range and Doppler for a specific bearing and elevation. The received power as a function of the range and Doppler. The high peak correspond to the energy reflected by the target plus noise and the surrounding to just noise.

2.1.1 Resolution

The sensor provides a sequence of two-dimensional images (also referred to as maps or frames), Figure 2.2. The resolution of these images (the size of the rectangular cells) is mainly determined by the physical characteristics of the antenna and the puls transmitted.

$$\Delta r = \frac{c\tau}{2} = \frac{c}{2B}, \qquad \beta \sim \frac{\lambda}{A_e}, \qquad \Delta f_d = \frac{1}{T}. \tag{2.4}$$

It should be clarified that the resolution is not the same as the accuracy. The resolution is a hard constraint depending on the radar.

2.1.2 Radar Cross Section (RCS)

The reflected energy is dependent on the characteristics of the target such as the shape, material, size, and the area exposed to the radar. RCS is a parameter denoted

Table 2.1.

Δr	range resolution	m
c	speed of light	m/s
τ	pulse width	s
B	band width	Hz
β	antenna beam-width	deg
λ	radar wave length	m
A_e	antenna aperture	m
f_d	doppler shift	Hz
T	time of pulse integration	s

by σ used to characterize the scattering properties of radar target. It is defined as an area intercepting that amount of power which, when scattered isotropically, produces at the receiver a density which is equal to that scattered by the actual target.
RCS can, to a certain degree be used to identify targets but in general the RCS also is a function of the electrical properties of the target and the radar frequency. Therefore two targets with the same physical size and shape can have different RCS.

2.1.3 The Radar Equation

The most essential concept in the context of radar, is the radar equation. The radar equation gives the relationship between the transmitted energy $P_{transmitted}$, the received energy $P_{received}$ and the range r. It can be derived starting from an isotropic antenna which radiates in all directions. The radar equation consists of three different factors, according to (2.5).

$$P_{received} = \underbrace{\frac{P_{transmitted}G_t}{4\pi r^2}}_{I} \times \underbrace{\frac{\sigma}{4\pi r^2}}_{II} \times \underbrace{A_e}_{III}. \tag{2.5}$$

The factor G_t is the antenna gain and describes the ability of the antenna to concentrate the transmitted energy in a narrow angular region (a directive beam). The factors in the equation can then be described as

I. The power density at a range of r meter from the radar.

II. The factor σ is the target cross section in square meters and the denominator accounts for the divergence on the return path (from target to radar) of the radiation.

III. The effective antenna aperture area A_e.

The product of the first two factors is then the power density at the radar caused by the reflection. The antenna intercepts a portion of power in an amount given by the product of the three factors.

2.2 Target Tracking

Tracking can be described as *the processing of measurements obtained from targets in order to maintain estimates of their states*. There exist several good books, for instance [2, 1, 3], where sensor models, target models and estimation theory are thoroughly described. Target tracking is an essential requirement for surveillance systems. Sensor data is collected, which besides reflections from the target also consists of noise (e.g., clutter, background noise, and thermal noise).

In this section target tracking will be described, both the classical method and a recent proposed method called *track before detect* (TBD) which is used in this Master's thesis.

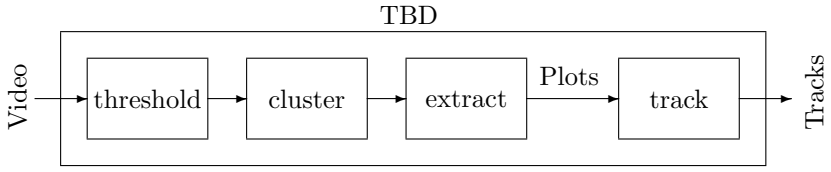


Figure 2.3. Signal processing in target tracking. The small boxes describes the different steps in the classical way of tracking and the big box around them indicates that in TBD everything is done unthresholded and simultaneously.

2.2.1 Classical Radar Detector

Basically, in the classical setting, the measurements, Figure 2.2 are thresholded to obtain a set of plots. These plots are taken as the input to the tracking stage where the tracks are formed, see Figure 2.3.

Detection

In the classical way of radar, detecting decisions is based on comparing the video data to a threshold. If the envelope of the signal exceeds a predefined threshold, a detection is made. In this way the space of all measurements is divided in two regions "detections" and "no detections". When using a threshold to determine whether a target is present or not, two different errors can occur. The first, called *false alarm*, is introduced when threshold is exceeded even though there is no target present. A high measurement of noise can then be detected as a target. The other one is called *miss detection* and occurs when the threshold is not exceeded even

though there is a target present. A target with low signal to noise ratio can then be missed.

The detection process is normally made in three different stages and in each of these stages it is possible to introduce one of the two errors mentioned above. In the first stage, detection is made on *hit level*. Here the energy in the range-Doppler bins, (Figure 2.2) is compared to a threshold. Those bins with energy exceeding the threshold are called hits.

The next step is detection on a *plot level*, only the bins with hits from the previous stage are considered here. In this stage, clusters are made from hits close to each other that probably originate from the same target. The center of the energy in a cluster is taken as the center of the possible target. The output of this stage is the measurements of the cells in which a target probably is. These measurements are called plots and consist of range, elevation, bearing, and doppler speed.

The last decision is made on *track level*. When a predefined number of plots is associated with the same track, a target is declared present. In this way the errors are defined on track level as

false alarm: A target is declared present when there is no target. The false alarm probability is denoted by P_{FA}.

miss detection: The energy reflected by the target does not exceed the threshold level and is not detected. The probability of miss detection $P_{MD} = 1 - P_D$ where P_D is the probability of detection.

Forming Tracks

The next step in the chain is to make tracks from the plots. Because of the errors introduced in the detection previously, some of the plots are not correct. A common used term for plots originating from other than targets, is clutter (waves, trees, birds etc). This means that we have to decide which of the plots that should be put together as tracks. This procedure is referred to as *data association*.

Data Association

The problem is to determine which track a measurement should belong to. There exist several different methods which have been discussed in the literature, (Bar-Shalom and Fortman 1988, Bar-Shalom and Li 1993, Blackman 1986, and Blackman and Popoli 1999). Issues that have to be thought of are for example, how to initiate and terminate tracks and update already existing tracks. A classical association method is *nearest neighbor*, which uses only the closest observation to any given state to perform the measurement update step. Other more advanced algorithms exist that for example use the statistical properties of clutter to avoid that kind of measurements from false alarms. The most general technique is the *multiple hypothesis tracking*, [15].

Filtering

The process of extracting information from a (dynamic) system is generally referred to as filtering. The objective is to construct an estimate of the target state. The output of the tracking filter is the *probability density function* (*pdf*) $p(s_k|Z_k)$ of the target state s_k, where Z_k contains all the measurements up to time k. From $p(s_k|Z_k)$ an estimate $\hat{s}_k$ is made, which contains e.g., position, velocity, and acceleration of the target. For further details see Chapter 3.

2.2.2 Track Before Detect

The use of threshold detection clearly separates the functions of detection and tracking as shown in Figure 2.3. The decision whether a target is present or not is only based on information from single scans. For weak objects the decision of the threshold soon turns into an impossible balancing act between false alarm and no detection. Modern types of stealthy targets such as recently developed fighters and missiles, call for a different approach to detection.

To overcome this problem, a method called *track before detect* (TBD) has recently been proposed. In TBD the detection problem is done using the track output over multiple scans. The detection decision will be made at the end of the processing chain, i.e., when all information has been used and integrated over time, [5, 4]. Although it is called track before detect, the tracking and detection processes occur simultaneously. In this way, the energy of a (weak) target is integrated and correlated over time and position. The concept will lead to a better performance when detecting and tracking weak targets. TBD also implicitly solves the problem with data association.

Chapter 3

Estimation Theory

This chapter provides the background and the basic theory of recursive Bayesian estimation. The problem is to sequentially estimate the state of a dynamic system using a sequence of noisy measurements from the system. A discrete time system will be considered. The state vector contains all relevant information required to describe the system. In this application with TBD that could be e.g., velocity and position of a target. The measurement vector on the other hand, will contain noisy observations that are related to the state vector, for instance range and distance to the target, see Section 2.1.

3.1 Bayesian Estimation

Two models will be needed to describe the dynamical system, a dynamical model (3.1a) and measurement model (3.1b). The first one describes the system dynamics and the second describes the relation between the measurements and the state vector. If the models are available in probabilistic form, the solution is given by recursive Bayesian estimation, Jazwinski (1970).

A recursive filter consists mainly of two stages, a prediction and an update step. The prediction stage predicts the next *pdf* using the system model. Because of the system noise this will make the predicted *pdf* broadened and deformed. The update stage will use the latest measurement to tighten the *pdf* using Bayes' theorem [1] which enables us to update knowledge about the state with the new received information.

Consider the nonlinear discrete-time state space system

$$s_{k+1} = f(s_k, w_k), \qquad k \in \mathbb{N}, \tag{3.1a}$$

$$z_k = h(s_k, v_k), \qquad k \in \mathbb{N}, \tag{3.1b}$$

where $s_k \in \mathbb{R}^n$ is the state of the system at a certain time k. $z_k \in \mathbb{R}^m$ is the measurement model, often nonlinear. The system noise w_k and measurement noise

[1] Bayes 1763. For Bayes' theorem see Appendix A

v_k are introduced to deal with the inaccuracy due to modeling and measurements. These are characterized by the probability densities $p_w(w)$ and $p_v(v)$. Also let

$$Z_k = \{z_0, \ldots, z_k\} \tag{3.2}$$

be the stacked vector of observations up to time k. Given observations up to time k, Section (3.2) we want to find the optimal estimate $s_k \in \mathbb{R}^n$. If $p(s_k|Z_{k-1})$ is assumed to be known, the update of the *pdf* by the measurement z_k can be derived as

$$\begin{aligned} p(s_k|Z_k) &= \frac{p(Z_k|s_k)p(s_k)}{p(Z_k)} = \frac{p(z_k, Z_{k-1}|s_k)p(s_k)}{p(z_k, Z_{k-1})} \\ &= \frac{p(z_k|s_k, Z_{k-1})p(Z_{k-1}|s_k)p(s_k)}{p(z_k|Z_{k-1})p(Z_{k-1})} \\ &= \frac{p(z_k|s_k)p(Z_{k-1}|s_k)p(s_k)}{p(z_k|Z_{k-1})p(Z_{k-1})} = \frac{p(z_k|s_k)p(s_k|Z_{k-1})}{p(z_k|Z_{k-1})}, \end{aligned} \tag{3.3}$$

where Bayes' rule, the Markov property and the definition of $Z_k = \{Z_{k-1}, z_k\}$ have been used. Derivation of the time update equation can be done, if assuming that the *pdf*, $p(s_k|Z_k)$, is known, then

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

Intergrating both sides with respect to s_k yields

$$p(s_{k+1}|Z_k) = \int_{\mathbb{R}^n} p(s_{k+1}|s_k)p(s_k|Z_k)\, ds_k. \tag{3.5}$$

Summarizing, the time-update and the measurements-update for the *pdf* are given by

$$p(s_{k+1}|Z_k) = \int_{\mathbb{R}^n} p(s_{k+1}|s_k)p(s_k|Z_k)\, ds_k, \tag{3.6a}$$

$$p(s_k|Z_k) = \frac{p(z_k|s_k)p(s_k|Z_{k-1})}{p(z_k|Z_{k-1})}. \tag{3.6b}$$

The equations (3.6a) and (3.6b) form the basis for the optimal Bayesian solution. This recursive propagation of the porterior density is only a conceptual solution in the sense that it in general cannot be determined analytically. Only in a few special cases were the system is linear and Gaussian the Kalman filter can be applied. In reality though, the system is often nonlinear and non-Gaussian. The choice is then either to linearize the system and apply the Kalman filter. This would lead to the *extended Kalman filter* (EKF), [3]. Another approach is to find a way to handle nonlinearities. The particle filter is such an approach.

3.1.1 Kalman Filter

A widely used method for Bayesian estimation is the Kalman filter, Kalman 1960, Kailath 2000, B.D.O Anderson 1979. The Kalman filter assumes that the posterior

density at every time step is Gaussian and hence can be completely characterized by two parameters, the mean and the covariance. If we assume the model to be linear with additive Gaussian noise, the optimal recursive estimation is given by the Kalman filter. Such a system is described as

$$s_{k+1} = F_k s_k + G_k w_k, \qquad k \in \mathbb{N}, \tag{3.7a}$$
$$z_k = H_k s_k + v_k, \qquad k \in \mathbb{N}, \tag{3.7b}$$

where F_k and H_k are known matrices defining the linear functions and

$$w_k \sim \mathcal{N}(0, Q_k), \qquad v_k \sim \mathcal{N}(0, R_k), \qquad s_0 \sim \mathcal{N}(\hat{x}_0, P_0).$$

Kalman showed that when Z_k is given, s_k and s_{k+1} are Gaussian distributed according to

$$p(s_k|Z_k) \sim \mathcal{N}(\hat{s}_{k|k}, P_{k|k}), \tag{3.8a}$$
$$p(s_{k+1}|Z_k) \sim \mathcal{N}(\hat{s}_{k+1|k}, P_{k+1|k}), \tag{3.8b}$$

where

$$\hat{s}_{k|k} = \hat{s}_{k|k-1} + P_{k|k-1} H_k^T S_k^{-1} (z_k - H_k \hat{s}_{k|k-1}), \tag{3.9a}$$
$$P_{k|k} = P_{k|k-1} - P_{k|k-1} H_k^T S_k^{-1} H_k P_{k|k-1}, \tag{3.9b}$$
$$S_k = R_k + H_k P_{k|k-1} H_k^T, \tag{3.9c}$$
$$\hat{s}_{k+1|k} = F_k \hat{s}_{k|k}, \tag{3.9d}$$
$$P_{k+1|k} = F_k P_{k|k} F_k^T + W_k Q_k W_k^T, \tag{3.9e}$$

with initial values $\hat{s}_{0|-1} = \hat{s}_0$ and $P_{0|-1} = P_0$. The Kalman filter recursively computes the mean and the covariance of the Gaussian posterior. Note that if the system is linear and Gaussian the optimal solution is given by the Kalman filter. Figure 3.1 shows a Kalman approximation of a non-Gaussian *pdf*.

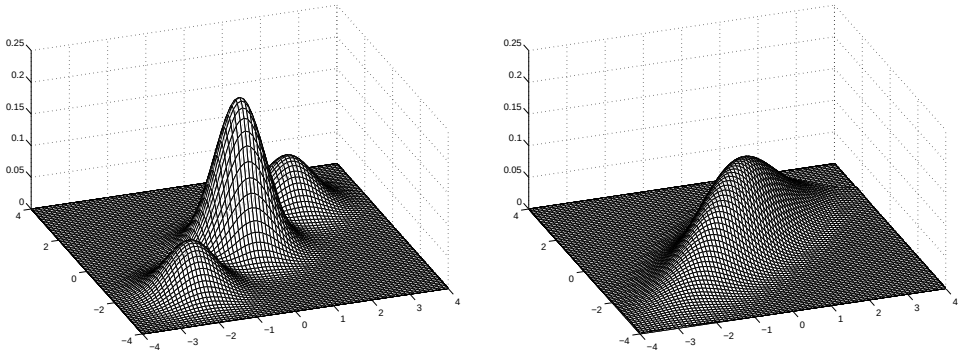
3.1.2 Particle Filter

One approach to the nonlinear non-Gaussian estimation problem is to use a particle filter [3, 18, 9]. The main idea with particle filter is to represent the required posterior density function by a set of random samples with associated weights and to compute estimates based on these samples and weights. The only approximation then is the number of particles used, see Figure 3.2.

Monte Carlo Integration

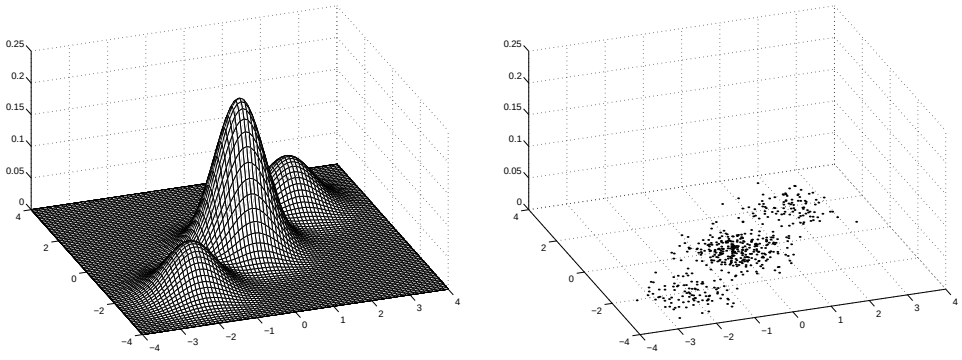
To approximate the integrals in the Bayesian solution we will look at something called *Monte Carlo* (MC) methods. MC methods (simulation-based methods) are an alternatives to deterministic methods. They aim to approximate integrals on the form

$$I = \int g(s) p(s)\, ds, \tag{3.10}$$



(a) (b)

Figure 3.1. True *pdf* and Kalman approximation. The Kalman approximation of the true *pdf* is completely described by its mean and its covariance. Thus it can not describe a non-Gaussian density.



(a) (b)

Figure 3.2. True *pdf* and Particle Filter approximation. The key idea with the particle filter is that it should approximate the true *pdf* with a particle cloud. The cloud will evolve with time as measurements become available. The level of probability is proportional to the density of the cloud.

where $s \in \mathbb{R}^n$ and $p(s)$ are suppose to be interpreted as the underlying probability density such that

$$\int p(s)\,ds = 1, \qquad p(s) > 0 \qquad \forall s. \tag{3.11}$$

If it is possible to draw N samples $\{s^i\}_{i=1}^N$ according to $p(s)$, then, the integral can then be approximated by the sum

$$\hat{I} = \frac{1}{N} \sum_{i=1}^{N} g(s^i). \tag{3.12}$$

The samples $\{s^i\}_{i=1}^N$ automatically come from regions of the state space that are important for the integration result.

Importance Sampling

It is generally not possible to sample directly from the distribution $p(s)$. Still, if we let $\pi(s)$ be a distribution from which samples can be generated, a weighting of these will make the MC estimation possible. An *importance weight* can be defined for each sample from $\pi(s)$ as

$$\tilde{q}(s^i) = \frac{p(s^i)}{\pi(s^i)}. \tag{3.13}$$

The density function $\pi(s)$ is usually referred to as *importance function*. With this new expression, the integral in (3.10) can be written as

$$I = \int g(s) \frac{p(s)}{\pi(s)} \pi(s)\, ds = \int g(s) q(s) p(s)\, ds, \tag{3.14}$$

which accordning to (3.12) can be approximated by

$$\hat{I} = \frac{1}{N} \sum_{i=1}^{N} g(s^i) \tilde{q}(s^i), \tag{3.15}$$

where $\{s^i\}_{i=1}^N$ are samples according to $\pi(s)$. If the normalizing factor of the density $p(s)$ is unknown, a normalization of the importance weights has to be done

$$\hat{I} = \frac{\frac{1}{N} \sum_{i=1}^{N} g(s^i) \tilde{q}(s^i)}{\frac{1}{N} \sum_{j=1}^{N} \tilde{q}(s^i)} = \sum_{i=1}^{N} g(s^i) q(s^i), \tag{3.16}$$

where the normalizing weights are given by

$$q(s^i) = \frac{\tilde{q}(s^i)}{\sum_{j=1}^{N} \tilde{q}(s^j)}. \tag{3.17}$$

Sequential Monte Carlo

In the case of importance sampling, all the importance weights have to be recalculated as a new measure becomes available. Let $S_k = \{s_j\}_{j=1}^k$ represent all target

states up to time k. $p(S_k|Z_k)$ is then the joint posterior density at time k, which can be approximated by

$$p(S_k|Z_k) = \sum_{i=1}^{N} q_k^i (S_k - S_k^i), \tag{3.18}$$

where the normalized weights are chosen according to *importance sampling* and the samples are drawn from an importance density $\pi(S_k|Z_k)$. That is, the importance weights are

$$q_k^i \propto \frac{p(S_k^i|Z_k)}{\pi(S_k^i|Z_k)}. \tag{3.19}$$

If the importance density is assumed to be on the form

$$\pi(S_k|Z_k) = \pi(s_k|S_{k-1}, Z_k)\pi(S_{k-1}|Z_{k-1})., \tag{3.20}$$

it is possible to evaluate the importance weights recursively as new measurements become available. To derive the update equation for the weights, the *pdf* $p(S_k|Z_k)$ is expressed in terms of $p(S_{k-1})$, $p(z_k|s_k)$, and $P(s_k|s_{k-1})$:

$$\begin{aligned} p(S_k|Z_K) &= \frac{p(z_k|S_k, Z_{k-1})p(S_k|Z_{k-1})}{p(z_k|Z_{k-1})} \\ &= \frac{p(z_k|S_k, Z_{k-1})p(s_k|S_{k-1,Z_{k-1}})p(S_{k-1}|Z_{k-1})}{p(z_k|Z_{k-1})} \\ &= \frac{p(z_k|s_k)p(s_k|s_{k-1})}{p(z_k|Z_{k-1})} p(S_{k-1}|Z_{k-1}) \\ &\propto p(z_k|s_k)p(s_k|s_{k-1})p(S_{k-1}|Z_{k-1}). \end{aligned} \tag{3.21}$$

Inserting (3.21) and (3.20) into (3.19), the recursive update of the weights can be written

$$\begin{aligned} q_k^i &\propto \frac{p(z_k|s_k^i)p(s_k^i|s_{k-1}^i)p(S_{k-1}^i|Z_{k-1})}{\pi(s_k^i|S_{k-1}^i, Z_k)\pi(S_{k-1}^i|Z_{k-1})} \\ &= q_{k-1}^i \frac{p(z_k|s_k^i)p(s_k^i|s_{k-1}^i)}{\pi(s_k^i|S_{k-1}^i, Z_k)}. \end{aligned} \tag{3.22}$$

If $\pi(s_k|S_{k-1}, Z_k) = \pi(s_k|s_{k-1}, z_k)$ this result in that not all $\{S_{k-1}^i\}_{i=1}^N$ and Z_{k-1} have to be stored, but only the latest states $\{s_k^i\}_{i=1}^N$. The weight update equation will thus be

$$q_k^i \propto q_{k-1}^i \frac{p(z_k|s_k^i)p(s_k^i|s_{k-1}^i)}{\pi(s_k^i|s_{k-1}^i, z_k)}. \tag{3.23}$$

The posterior filtered density can then be approximated by

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

Algorithm 1 Particle Filter Algorithm

1: *Initialization*
Set $k = 0$, generate N particles $\{s_0^i\}_{i=1}^N$ according to the initial distribution $p_0(s_0)$.
2: *Measurement update*
Compute the weights, $\tilde{q}_k^i = p(z_k|s_k^i)$ and normalize $q_k^i = \frac{\tilde{q}_k^i}{\sum_{j=1}^N \tilde{q}_k^j}$, $i = 1, \ldots, N$.
3: *Resample*
Generate N new particles, $\{\breve{s}_k^i\}_{i=1}^N$ by resampling with replacement N times from $\{S_k^i\}_{i=1}^N$, where $Prob(\breve{s}_k^i = s_k^j) = q_k^j$.
4: *Time update*
Predict new particles, i.e., $s_{k+1}^i = f(\breve{s}_k^i, w_k^i)$, $i = 1, \ldots, N.$, where w_k^i is drawn from the process noise with *pdf* $p_w(w_k)$.
5: Increase k and continue from step 2.

with weights from (3.23). It can be shown that when $N \to \infty$ the approximation in (3.24) approaches the true posterior density $p(s_k|Z_k)$.

The point-estimate for the particle filter can be calculated as

$$\hat{s}_k^{MMSE} = arg\,min\,\mathbb{E}\{(\hat{s} - s)^2|Z_k\} \tag{3.25}$$

which is denoted as the *minimum mean square estimate* (MMSE). It can be shown that the solution is given by the conditional mean

$$\hat{s}_k^{MMSE} = \mathbb{E}(s_k|Z_k) = \int s_k p(s_k|Z_k)\, ds_k \approx \sum_{i=1}^N q_k^i s_k^i. \tag{3.26}$$

The uncertainty region can be calculated as

$$\begin{aligned} P_k &= \int (s_k - \hat{s}_k^{MMSE})(s_k - \hat{s}_k^{MMSE})^T p(s_k|Z_k)\, ds_k \\ &\approx \sum_{i=1}^N q_k^i (s_k - \hat{s}_k^{MMSE})(s_k - \hat{s}_k^{MMSE})^T. \end{aligned} \tag{3.27}$$

Chapter 4

A Time Efficient Implementation of a Track Before Detect Application

4.1 Introduction

Critics of the particle filters hangs on to their argument that the method it is far too computationally intensive. However, since research really took of in this field the computational power has been multiplied many times. Today particle filters are, though it is a new field of research, used in several industrialized applications and, as computers and methods improve a boosting effect is expected, sooner or later. This chapter is dedicated to investigate the complexity i.e., the computational time for a TBD application. Suggestions of improvements as well as results and conclusions will be presented. Although this is done based on simulations in MATLAB, it is a fact that most of the improvements can be used in general.

Roughly the particle filter process can be divided into three main algorithm steps

- Time update - Prediction
- Measurement update - Calculation of weights
- Resampling.

For a non time optimized implementation of a *particle filter* (PF) for TBD applications, the relative computational time is distributed among these steps as in Figure 4.1 From this it follows that a natural approach to this problem would be to initially focus on decreasing the computational time of the measurement update step. If this is carried out successfully the resample step should be dealed with next. In this paper no effort will be put on the time update step since its contribution to the total computational time is relatively very small.

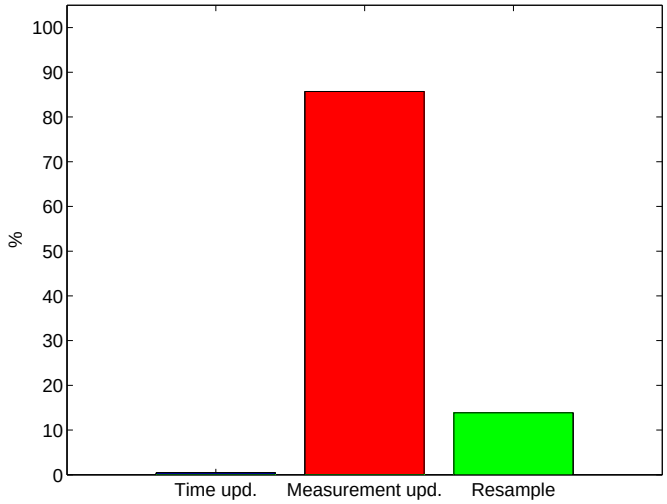


Figure 4.1. Relative time spent on the three main steps, in a non timeoptimized implementation of a particle filter in MATLAB.

If a time improvement is made based on a approximation or a different algorithm, it must also be taken into account whether this affects the quality of the estimations or not. A useful definition of quality for this purpose is also discussed.

4.2 System Setup

Consider the general and nonlinear discrete time system

$$s_{k+1} = f(s_k, m_k, w_k), \qquad k \in \mathbb{N} \tag{4.1}$$

$$Prob\{m_k = i | m_{k-1} = j\} = \Pi_{ij} \tag{4.2}$$

$$z_k = h(s_k, m_k, v_k), \qquad k \in \mathbb{N} \tag{4.3}$$

where

- $s_k \in \mathbb{R}^n$ is the state of the system
- $m_k \in \mathbb{N}$ is the modal state of the system
- $z_k \in \mathbb{R}^p$ are the measurements
- w_k is the process noise
- v_k is the measurement noise
- f is the system dynamic function

- h is the measurement function
- Π is the Markov transition matrix

4.2.1 Target Model

The predicted state of the target moving in the $x-y$ plane is given from the dynamic target model

$$s_{k+1} = f(s_k, m_k, w_k). \tag{4.4}$$

$$s_k = \begin{pmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{pmatrix}. \tag{4.5}$$

Let T be the update time, and the system dynamics is defined as

$$f(s_k, m_k) = \begin{pmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} s_k. \tag{4.6}$$

The process noise w_k is assumed to be white Gaussian noise and the process noise input model is given by

$$g(s_k, m_k) = \begin{pmatrix} \frac{1}{2}(\frac{1}{3}a_{maxx})T^2 & 0 \\ 0 & \frac{1}{2}(\frac{1}{3}a_{maxy})T^2 \\ \frac{1}{3}a_{maxx}T & 0 \\ 0 & \frac{1}{3}a_{maxy}T \end{pmatrix} w_k \tag{4.7}$$

with maximum accelerations a_{maxx}, a_{maxy}. This provides the posterior target state

$$s_{k+1} = \begin{pmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} s_k + \begin{pmatrix} \frac{1}{2}(\frac{1}{3}a_{maxx})T^2 & 0 \\ 0 & \frac{1}{2}(\frac{1}{3}a_{maxy})T^2 \\ \frac{1}{3}a_{maxx}T & 0 \\ 0 & \frac{1}{3}a_{maxy}T \end{pmatrix} w_k \tag{4.8}$$

Furthermore m_k indicates whether there is a target present or not. The probability of transition from absent to present i.e., "birth of target" $Prob\{m_k = 1 | m_{k-1} = 0\} = P_b$ and probability of transition from present to absent i.e., "death of target" $Prob\{m_k = 0 | m_{k-1} = 1\} = P_d$ and the transitional Markov matrix including all modal state transitions can be written as

$$\Pi = \begin{pmatrix} 1 - P_b & P_b \\ P_d & 1 - P_d \end{pmatrix}. \tag{4.9}$$

The Markov transition matrix describes the probabilities of jumps from one mode to another, see Figure 4.2.

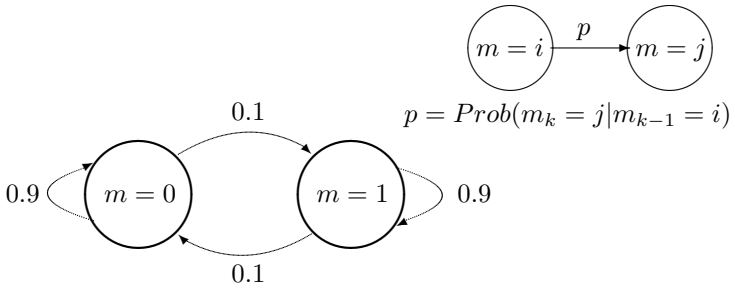


Figure 4.2. Markov transition state graph.

4.2.2 Measurement Model

At each discrete time k the sensor is providing a new set of measurements z_k. These measurements are the power levels in $N_r \times N_d \times N_b$ sensor cells. According to the measurement model in the single target TBD application, it consists of either a target and noise or noise only

$$z_k = v_k, \qquad m_k = 0, \tag{4.10a}$$

$$z_k = h(s_k, v_k), \qquad m_k = 1. \tag{4.10b}$$

In the first case the target is absent and therefore measurements consist of nothing but noise. In the second case there is a target present and its existence will affect the power level in each sensor cell. The power measurements per range-Doppler-bearing cell are defined by

$$z_k^{ijl} \quad = \quad |z_{A,k}^{ijl}|^2 \tag{4.11}$$

where $z_{A,k}^{ijl}$ is the complex amplitude of the target which is

$$z_{A,k} \quad = \quad A_k h_A(s_k) + n_k \tag{4.12}$$

where

$$A_k \quad = \quad \tilde{A}_k e^{i\phi_k} \qquad \phi_k \in (0, 2\pi) \tag{4.13}$$

and n_k is complex Gaussian noise defined by

$$n_k \quad = \quad n_{Ik} + i n_{Qk} \tag{4.14}$$

where n_{Ik} and n_{Qk} are independent, zero mean white Gaussian noise with variance σ_n^2 and related to v_k as

$$v_k \quad = \quad |n_{Ik} + n_{Qk}|^2. \tag{4.15}$$

h_{Ak} is the reflection form that is defined for every range-Doppler-bearing cell by

$$h_{Ak}^{ijl}(s_k) = e^{-\frac{(r_i - r_k)^2}{2R} L_r - \frac{(d_j - d_k)^2}{2D} L_d - \frac{(b_l - b_k)^2}{2B} L_b}, \tag{4.16}$$

where $i = 1, ..., N_r$, $j = 1, ..., N_d$ and $l = 1, ..., N_b$ with

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

$$d_k = \frac{x_k \dot{x}_k + y_k \dot{y}_k}{\sqrt{x_k^2 + y_k^2}} \tag{4.17b}$$

$$b_k = \arctan\left(\frac{y_k}{x_k}\right) \tag{4.17c}$$

R, D and B are related to the size of a range, a Doppler and a bearing cell. L_r, L_d and L_b represent constants of loss.

These measurements, conditioned on s_k, are now exponentially distributed

$$p(z_k|s_k) = \begin{cases} \prod_{ijl \in \mathbb{D}} \frac{1}{\mu_t^{ijl}} e^{-\frac{1}{\mu_t^{ijl}} z_k^{ijl}}, & m_k = 1, \\ \prod_{ijl \in \mathbb{D}} \frac{1}{\mu_v^n} e^{-\frac{1}{\mu_v} z_k^{ijl}}, & m_k = 0, \end{cases} \tag{4.18}$$

where

$$\begin{aligned} \mu_v^{ijl} &= E[(z_k^{ijl}|s_k, m_k = 0)] \\ &= E[|n_{Ik} + in_{Qk}|^2] \\ &= E[n_{Ik}^2 + n_{Qk}^2] \\ &= 2\sigma_n^2. \end{aligned} \tag{4.19}$$

$$\begin{aligned} \mu_t^{ijl} &= E[(z_k^{ijl}|s_k, m_k = 1)] \\ &= E[|\tilde{A}_k e^{i\phi_k} h_{Ak}^{ijl}(s_k) + n_{Ik} + in_{Qk}|^2] \\ &= E[(\tilde{A}_k h_{Ak}^{ijl}(s_k)\cos(\phi_k) + n_{Ik})^2 + (\tilde{A}_k h_{Ak}^{ijl}\sin(\phi_k) + n_{Qk})^2] \\ &= \tilde{A}_k^2 (h_{Ak}^{ijl}(s_k))^2 + 2\sigma_n^2 \\ &= P h_{Pk}^{ijl}(s_k) + 2\sigma_n^2 \end{aligned} \tag{4.20}$$

with

$$\begin{aligned} h_{Pk}^{ijl}(s_k) &= (h_{Ak}^{ijl}(s_k))^2 \\ &= e^{-\frac{(r_i - r_k)^2}{R} L_r - \frac{(d_j - d_k)^2}{D} L_d - \frac{(b_l - b_k)^2}{B} L_b} \end{aligned} \tag{4.21}$$

which describes the power contribution of a target in every range-Doppler-bearing cell, where r_k, d_k, and b_k are given from target state s_k through (4.17).

4.3 Computational Complexity of the TBD Application

Comparing different TBD implementation settings based on computational complexity is a complicated issue, since this is due to the system architecture, memory management and programming language etc. Although only one function is replaced, an analysis of the whole system must be done to find out whether it really affects the total complexity considerably. Therefore the complexity is not calculated in number of flops etc., instead the computational time for each function is measured to gain knowledge about the complexity of the system. The value of the absolute time is depending on the computer capacity as well as the things mentioned above, hence it follows that it is of no interest except as a comparison between the different settings.

4.4 The Quality of the Estimation

As an indication of the quality of the estimations the *number of effective particles* N_{eff} will be considered, see Section 4.6, it is of interest since it gives a hint, whether an improvement in a time perspective affects the approximation of the pdf $p(s_k|Z_k)$ in such a way that the number of particles must be increased. This would cause a larger computational load and might therefore take away the gain in time. The $RMSE$ values of the state variables will also be considered. For M number of Monte Carlo runs the $RMSE$ is calculated as

$$RMSE_k = \sqrt{\frac{1}{M}\sum_{i=1}^{M}||s_k - \hat{s}_k^i||^2}, \tag{4.22}$$

where the estimated state of the target at time k is denoted $\hat{s}_k^i$.

4.5 Measurement Update

A natural way to a faster implementation of the TBD application starts with a faster measurement update step, see Figure 4.1. This section will provide a faster measurement update step based mainly on the approximation that most of the sensor cells can be excluded when the likelihood is calculated. Previous definitions of complexity and quality will be used, a few words will also be said about the detection performance, although it will not be deeply investigated.

4.5.1 Theory

When a particle filter is used to solve the filtering problem from (4.1) and (4.3), the weight for each particle is achieved by first calculating the likelihood

$$p(z_k|s_k) = \begin{cases} \prod_{ijl\in\mathbb{D}} \frac{1}{\mu_t^{ijl}} e^{-\frac{1}{\mu_t^{ijl}} z_k^{ijl}}, & m_k = 1, \\ \prod_{ijl\in\mathbb{D}} \frac{1}{\mu_v^n} e^{-\frac{1}{\mu_v} z_k^{ijl}}, & m_k = 0, \end{cases} \tag{4.23}$$

and then by normalizing

$$q_k^n = \frac{p(z_k|s_k^n)}{\sum_{n=1}^N p(z_k|s_k^n)}. \tag{4.24}$$

An implementation of this is presented in Algorithm 2.

Algorithm 2 Measurement update - Calculation of weights

1: Calculate the estimated range, Doppler and bearing from
$r_k = \sqrt{x_k^2 + y_k^2}$,
$d_k = \frac{x_k\dot{x}_k + y_k\dot{y}_k}{\sqrt{x_k^2+y_k^2}}$,
$b_k = \arctan\left(\frac{y_k}{x_k}\right)$.
2: for $n = 1 : N$
if $m_k^n = 0$
$\mu_0^n = 2\sigma_n^2$.
else
$\mu_0^n = P_k e^{-\frac{(r_i - r_k^n)^2}{R} L_r - \frac{(d_j - d_k^n)^2}{D} L_d - \frac{(b_l - b_k^n)^2}{B} L_b} + 2\sigma_n^2$.
end if.
$p(z_k|s_k^n) = \frac{1}{\mu_0^n} e^{-\frac{1}{\mu_0^n} z_k}$.
end for.
3: $q_k = \frac{p(z_k|s_k^n)}{\sum_{n=1}^N p(z_k|s_k^n)}$.

In one iteration the likelihood for one particle is calculated based on the contribution from all cells, this means that the number of iterations equals the number of particles.

The power from a target (if one present) is distributed to cell $ijl \in \mathbb{D}$ according to the exponentially function h_{Pk} times the power of the target P_k. However the shape of h_{Pk} provides an opportunity that with a slighter approximation only a relatively small number of cells will have to be taken into account. This approximation will be formulated as: The contribution from a target will only be considered for those cells $ijl \in \mathbb{D}$ where h_{Pk}^{ijl} is larger than a certain threshold h_{th} see Figure 4.3. Denote this group of cells as $\mathbb{S} \subset \mathbb{D}$, and for particle n in state s_k^n as $\mathbb{S}^n \subset \mathbb{D}$.

In other words, assume that it is not likely that a target present in a certain state affects all cells considerable. Now look at the likelihood function, and that it

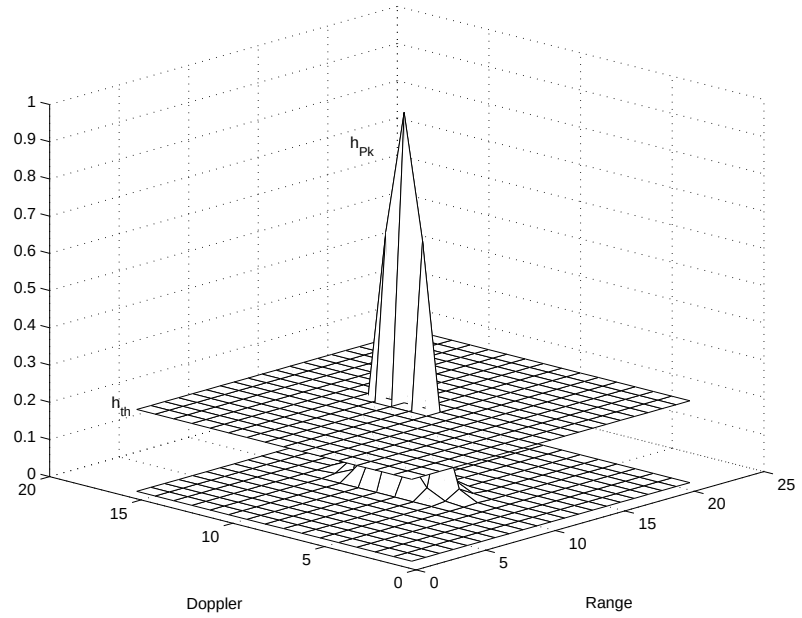


Figure 4.3. The group of cells, $\mathbb{S}$, in the vicinity of a targets location, that are assumed to be considerably affected. The size of $\mathbb{S}$ is determined by threshold h_{th}

can be written as

$$p(z_k|s_k) = \begin{cases} \prod_{ijl\in\mathbb{D}} p_t(z_k^{ijl}|s_k), & m_k = 1, \\ \prod_{ijl\in\mathbb{D}} p_v(z_k^{ijl}|H_0), & m_k = 0. \end{cases} \tag{4.25}$$

$p_t(z_k^{ijl}|s_k)$ is the probability density function of a target and noise in cell ijl, given there is a target in state s_k and $p_v(z_k^{ijl}|H_0)$ is the probability density function in cell ijl, when there is no target present and the power level in the cell is only due to measurement noise. Now use the approximation that a target only affects the cells in its surroundings $\mathbb{S}$

$$p(z_k|s_k) = \begin{cases} \prod_{ijl\in\mathbb{S}} p_t(z_k^{ijl}|s_k) \prod_{ijl\in\mathbb{D}\cap\overline{\mathbb{S}}} p_v(z_k^{ijl}|H_0), & m_k = 1, \\ \prod_{ijl\in\mathbb{D}} p_v(z_k^{ijl}|H_0), & m_k = 0, \end{cases} \tag{4.26}$$

where $\mathbb{D}\cap\overline{\mathbb{S}}$ is the group of cells where the contribution from a target in state s_k is

neglected. The weights are calculated by normalizing the likelihood, which means that we are only interested in the ratio of the likelihood between all states $\{s_k^n\}_{n=1}^N$ i.e., the relative size of the weights, therefore expand (4.26)

$$p(z_k|s_k) = \begin{cases} \prod_{ijl\in\mathbb{S}} p_t(z_k^{ijl}|s_k) \prod_{ijl\in\mathbb{D}\cap\bar{\mathbb{S}}} p_v(z_k^{ijl}|H_0), & m_k = 1, \\ \prod_{ijl\in\mathbb{S}} p_v(z_k^{ijl}|H_0) \prod_{ijl\in\mathbb{D}\cap\bar{\mathbb{S}}} p_v(z_k^{ijl}|H_0), & m_k = 0, \end{cases} \tag{4.27}$$

and introduce the likelihood ratio see [3]

$$p_{\frac{s_k}{H_0}}(z_k|s_k) = \frac{p(z_k|s_k)}{p(z_k|H_0)} = \begin{cases} \prod_{ijl\in\mathbb{S}} \frac{p_t(z_k^{ijl}|s_k)}{p_v(z_k^{ijl}|H_0)}, & m_k = 1 \\ 1, & m_k = 0 \end{cases} \tag{4.28}$$

and normalize to get

$$q_k^n = \frac{p_{\frac{s_k}{H_0}}(z_k|s_k^n)}{\sum_{n=1}^N p_{\frac{s_k}{H_0}}(z_k|s_k^n)}. \tag{4.29}$$

This provides the opportunity to a more time efficient implementation, see Algorithm 3. Now the loop can be performed over the limited cells $\mathbb{S}$ for all particles in the same iteration. The number of iterations are equal to the number of cells in $\mathbb{S}$ and independent of how many particles that are used.

4.5.2 Simulations and Results

A single target model will be used with a known SNR value of 10 dB. Measurements are simulated based on that a target appears at time $k = 6$ at a range of 89.6 km flying with a constant velocity of 200 m/s directly to the sensor. At each time k, measurements are provided from $N_r \times N_d \times N_b = 50 \times 16 \times 1$ sensor cells. That means, only one bearing cell is considered in this example. Initially, particles are uniformly distributed in the state space, in an area between $[85, 90]$ km, $[-0.22, -0.10]$ km/s in the x direction and $[-0.1, 0.1]$ km, $[-0.10, 0.10]$ km/s in the y direction. Initially the particles are also uniformly distributed over the two modes. As a reference and comparison of time efficiency, a particle filter with an ordinary Measurement Update step as the one in Algorithm 2 will be used, this setup is referred to as Setup 1. The size of the restricted region $\mathbb{S}$, where the contribution from a target will be calculated is, as mentioned, due to the threshold h_{th}. For a certain setup of the constant parameters the size of region $\mathbb{S}$ is shown in Table 4.1. The rectangular regions (in a range - Doppler plane) are chosen with a length and width that is required to fulfill $h_i > h_{th}$, $i \in \mathbb{S}$. Those values of h_{th} for which simulations will be

Algorithm 3 Measurement update - Efficient calculation of weights

1: Calculate the estimated range, Doppler and bearing from
$r_k = \sqrt{x_k^2 + y_k^2}$,
$d_k = \frac{x_k \dot{x}_k + y_k \dot{y}_k}{\sqrt{x_k^2 + y_k^2}}$,
$b_k = \arctan\left(\frac{y_k}{x_k}\right)$.

2: Decide which cells that should be considered as the center of $\mathbb{S}$
$r_{pos} = \text{round}\left(\frac{r_k - r_{min}}{r_{res}}\right)$,
$d_{pos} = \text{round}\left(\frac{d_k - d_{min}}{d_{res}}\right)$,
$b_{pos} = \text{round}\left(\frac{b_k - b_{min}}{b_{res}}\right)$.
where r_{res}, d_{res} and b_{res} are the resolution in range, Doppler and bearing respectively.

3: for $i, j, l \in \mathbb{S}$
if $m_k = 0$
$\mu_0 = 2\sigma_n^2$.
else
$\mu_0 = P_k e^{-\frac{(r_{pos_i} r_{res} - r_k)}{R} L_r - \frac{(d_{pos_j} d_{res} - d_k)}{D} L_d - \frac{(b_{pos_l} b_{res} - b_k)}{B} L_b} + 2\sigma_n^2$.
end if
$p^{ijl}_{\frac{s_k}{H_0}}(z_k^{ijl}|s_k) = \frac{\frac{1}{\mu_0} e^{-\frac{1}{\mu_0} z_k^{ijl}}}{\frac{1}{\mu_v} e^{-\frac{1}{\mu_v} z_k^{ijl}}}$.
end for

4: $p_{\frac{s_k}{H_0}}(z_k|s_k) = \prod_{ijl \in \mathbb{S}} p^{ijl}_{\frac{s_k}{H_0}}(z_k^{ijl}|s_k)$.

5: $q_k = \frac{p_{\frac{s_k}{H0}}(z_k|s_k)}{\sum_{n=1}^{N} p(z_k|s_k^n)}$.

Note! $p^{ijl}_{\frac{s_k}{H_0}}(z_k^{ijl}|s_k) = \left[p^{ijl}_{\frac{s_k}{H_0}}(z_k^{ijl}|s_k^1), \ldots, p^{ijl}_{\frac{s_k}{H_0}}(z_k^{ijl}|s_k^n)\right]$.

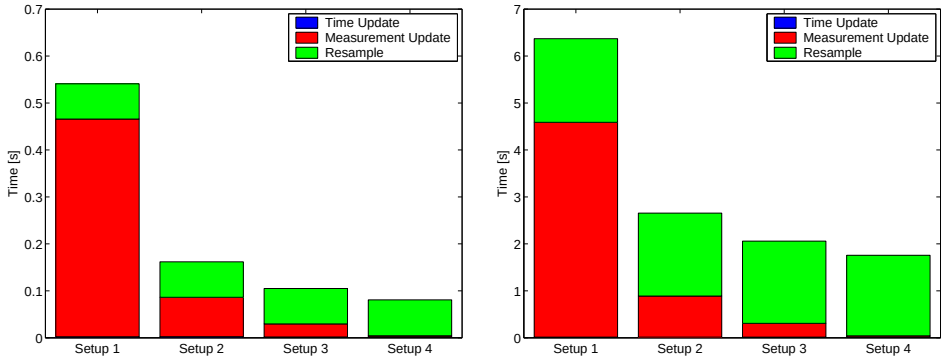
performed are written in boldface letters. These settings are referred to as Setup 2, Setup 3, and Setup 4 respectively.

Time Efficiency

Figure 4.4 shows time spent on one iteration for the different setups, and how it is distributed between the three particle filter steps. This is calculated as a mean value of a simulation that runs for 30 time steps. From Setup 1 to Setup 2 one might expect a likelihood function that is $\frac{N_r \times N_d \times N_b}{size(\mathbb{S})} = \frac{50 \times 16}{9 \times 5} \approx 18$ times faster, because of the cells that are excluded in the likelihood calculation. Results from simulations shows however that it is in reality 6 times faster. This is mainly due to the fact that the likelihood is calculated under two hypotheses and that it is finally given as a product of those ratios received in each iteration, but also because additional effort are put on things as calculating sets of subscripts. Still, with such a slight approximation, see Table 4.1, it is a noticeable difference in computational

Table 4.1. Size of sub areas $\mathbb{S} \subset \mathbb{D}$ for different values of the threshold h_{th}. In the range - Doppler plane a rectangular region is chosen with a length and width that is required to fulfill $h_i > h_{th}$, $i \in \mathbb{S}$.

$\mathbf{h_{th}}$	**size(S)**, $(r \times d \times b)$	$\frac{\sum_{i\in\mathbb{S}} h_i}{\sum_{j\in\mathbb{D}} h_j}$	**Setup**
0.0001	$\mathbf{5 \times 9 \times 1}$	**0.99988**	2
0.001	$3 \times 9 \times 1$	0.99921	
0.01	$\mathbf{3 \times 7 \times 1}$	**0.99432**	3
0.1	$3 \times 5 \times 1$	0.84725	
0.2	$1 \times 3 \times 1$	0.55892	
0.7	$\mathbf{1 \times 1 \times 1}$	**0.28224**	4



(a) (b)

Figure 4.4. Computational time for different setups of the likelihood calculation, and how it is distributed between the three particle filter steps: (a) $N = 1000$ particles and (b) $N = 10000$ particles. Note! The computational time for the Time Update step is hardly visible because it is relatively low compared to the other steps.

time. From Setup 2 to Setup 3, the likelihood function is approximately as expected $\frac{9\times5}{7\times3} \approx 2$ times faster, and a comparison of Setup 2 and Setup 4 shows that this ratio also approximately equals the theoretical value of $\frac{9\times5}{1} = 45$.

Tracking Performance

From Setup 1 to Setup 2 and Setup 3 there is no discernible difference in N_{eff} or $RMSE$, see Figure 4.5 and Figure 4.6, but neither are the approximations big, again see Table 4.1. In Setup 4 however, where the likelihood is calculated based on one sensor cell only, the approximation takes effect in the N_{eff}, but also in

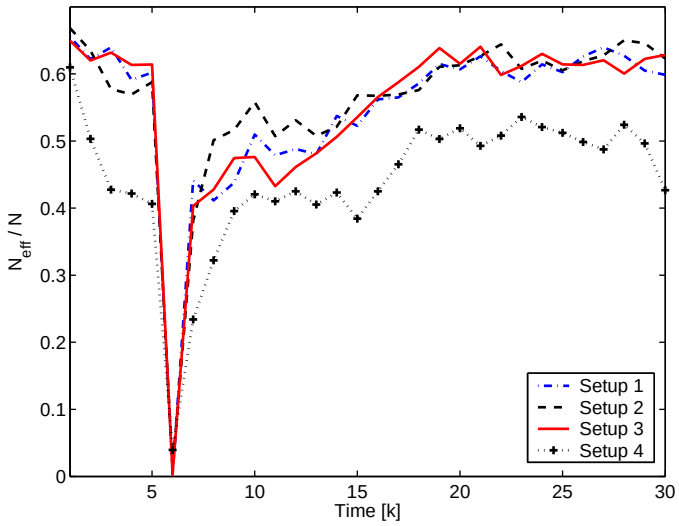
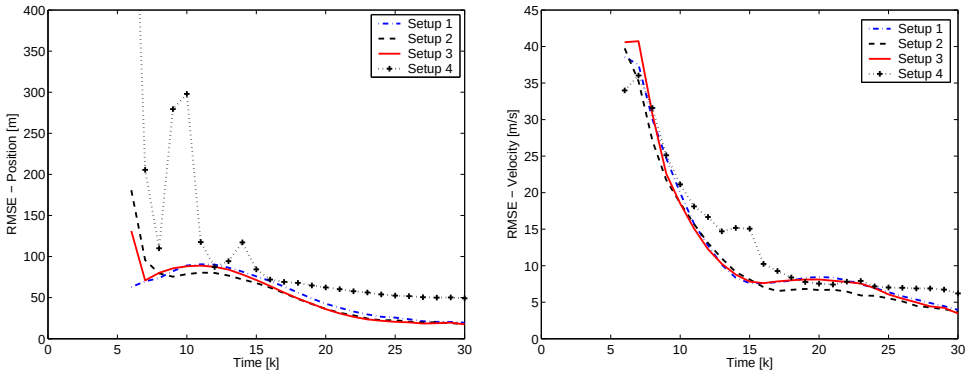


Figure 4.5. Number of effective particles $\frac{N_{eff}}{N}$, $N = 1000$.



(a) (b)

Figure 4.6. RMSE over 100 Monte Carlo runs, $N = 1000$ particles: (a) RMSE in position, (b) RMSE in velocity.

estimating the position. This is due to the fact the likelihood function is no longer calculated based on a comparison between the shapes of two surfaces, but instead as a comparison between two scalar values, which easily results in an offset.

Detection Performance

Apart from the problem of tracking a target there is also the problem of deciding whether a target is present or not. This problem is called the detection problem see [4, 25]. In a single target TBD application this can be done by calculating the mean value of the modal state for all particles, the probability of target existence, $Prob_k^{ex}$. At each time step k it is given as

$$Prob_k^{ex} = \frac{1}{N}\sum_{n=1}^{N} m_k^n, \tag{4.30}$$

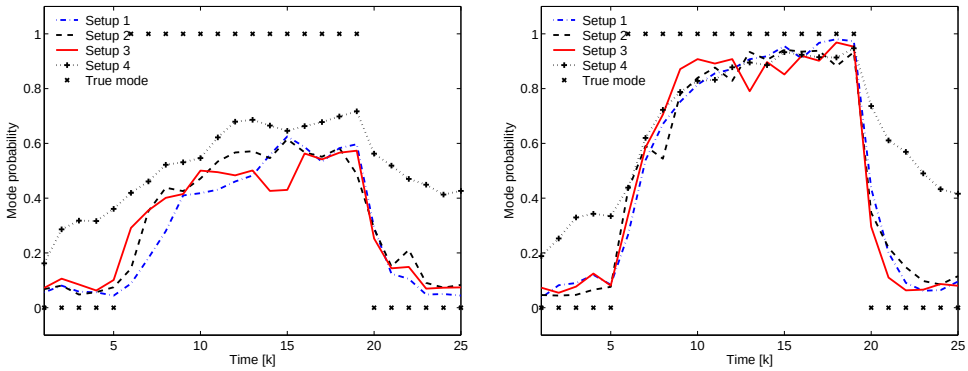
and whenever this value is larger than a certain threshold τ, a target is assumed to be detected

$$Prob_k^{ex} > \tau. \tag{4.31}$$

The choice of the threshold τ is far from trivial, it is a balancing act between so called false alarms i.e., exceeding of the threshold when no target is present and the risks of no detections i.e., the value falls bellow the threshold although a target is present. This problem also occurs in classical tracking. The probability of false alarm is also very low which means that a lot of simulations must be done to get a somehow reliable result. This cannot be done within a reasonable frame of time, because in a meaningful investigation the inefficient measurement update step should be compared with the efficient ones.

However, in Figure 4.7, $Prob_k^{ex}$ is presented as a mean value from 25 Monte Carlo runs, where target strengths are chosen corresponding to SNR values of 3 dB and 5 dB. The number of particles is set to 5000, $k = 1, \ldots, 25$, and target appears in frame $k = 6$ and disappears in frame $k = 20$.

In both $SNR = 3\,\text{dB}$ and $SNR = 5\,\text{dB}$, Setup 1, Setup 2, and Setup 3 are quite determined that there is no target present from $k = 1, \ldots, 5$. In $SNR = 5\,\text{dB}$ all three of them needs just a few frames to observe the presence of a target, and the probability remains high until $k = 20$ when target disappears, while in the $SNR = 3\,\text{dB}$ they are a bit indecisive on the existence. Setup 4 has larger problems to decide about the absence, most likely due to the fact that single peaks in noise will be interpreted as targets. However from these runs nothing can be said about a threshold that would be in favor of the "inefficient" likelihood in comparison with the two more efficient ones, Setup 2, and Setup 3. A loss in detection performance should also be reflected in a problem of tracking the target, which from results in Section 4.5.2 does not occur.



(a) $SNR = 3\,\mathrm{dB}$ (b) $SNR = 5\,\mathrm{dB}$

Figure 4.7. Probability of existence.

4.6 The Effect of Resampling

4.6.1 Theory

Resampling means that all particles and their normalized importance weights $\{\tilde{s}_k^i, q_k^i\}_{n=1}^N$ are replaced with a new set of particles with uniformed weights $\{s_k^i, N^{-1}\}_{n=1}^N$. This is obtained by drawing N samples from $\{\tilde{s}_k^i, q_k^i\}_{n=1}^N$, where q_k^i is the probability of getting particle $\tilde{s}_k^i$. The selection of $s_k^j = \tilde{s}_k^i$ is schematically shown in Figure 4.8. CSW is the cumulative sum of weights.

The resampling step in the particle filter is applied to avoid degeneracy. The degeneracy characteristics can be described as the problem that with time most of the importance weights will get a value of negligible size, see [11]. This means that our approximation $p(s_k|Z_k)$ will be described with few or only one particle because of its dominating weight and we will spend most of the computational time on particles who's contribution to the approximation of $p(s_k|Z_k)$ is marginal or none. A way of measure this degeneracy is to calculate the N_{eff}

$$N_{eff} = \frac{1}{\sum_{n=1}^{N}(q_k^i)^2}. \tag{4.32}$$

N_{eff} can vary from 1 to N, where $N_{eff} = 1$ means that the $p(s_k|Z_k)$ is approximated with one particle only and $N_{eff} = N$ that all particles have the same importance weight and therefore contributes equally. So as soon as the N_{eff} gets below a certain threshold N_{th} the resampling is required. The resample will however be performed in all iterations, since it follows from results later on, that the computational time for the resample step only takes about 5% or less of the total time. This under the assumption that an efficient one is used. In other applications

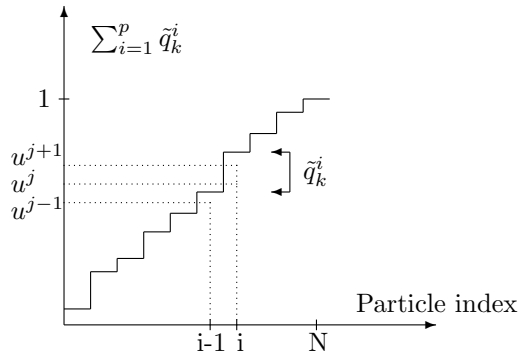


Figure 4.8. The selection of s_k^j from $\tilde{s}_k^i$ is performed by a random number $\tilde{u} \sim \mathcal{U}(0,1)$, where the probability of selecting $\tilde{s}_k^i$ is q_k^i.

it might be useful to study for which value of the threshold N_{th} the resample could be excluded and replaced with a recursive updating of the weights $\{q_k^i\}_{n=1}^N$. A straight on pseudocode for a *simple random resample* is presented in Algorithm 4

Algorithm 4 Simple Random Resampling

1: Create a CSW $\{c^n\}_{n=1}^N$ from $\{\tilde{q}^n\}_{n=1}^N$.
2: for $j = 1 : N$
  Draw a sample from a uniform distribution $\tilde{u} \sim \mathcal{U}(0,1)$.
  $i = 1$.
  while $u > c_i$
    $i = i + 1$.
  end while
  $s_k^j = \tilde{s}_k^i$.
  $q_k^j = N^{-1}$.
end for

However this resampling method is very time demanding, approximately it has a complexity of $\mathcal{O}(N^2)$. This soon provides a limited maximum number of particles that can be chosen for the particle filter implementation i.e., the TBD application. This *simple random resample* algorithm can though, easily be modified to a more efficient one that has the complexity of $\mathcal{O}(N)$ see Algorithm 5.

This algorithm will be considered in the simulations, but other even more time efficient ones are suggested, such as *stratified resampling* [10] [22], *residual resampling* [23] and *systematic resampling* [22] [24] etc. However the *simple random resampling* will also be replaced with the *systematic resampling*, since it is found

Algorithm 5 Efficient Simple Random Resampling

Draw N samples from a uniform distribution $\{\tilde{u}_k^n\}_{n=1}^N \sim \mathcal{U}(0,1)$.
Normalize and create a CSW, $\{u_k\}_{l=1}^N$.
Create a CSW $\{c^n\}_{n=1}^N$ from $\{\tilde{q}^n\}_{n=1}^N$.
Set $i = 1$.
for $j = 1 : N$
    while $u_j > c_i$
        $i = i + 1$.
    end while
    $s_k^j = \tilde{s}_k^i$.
    $q_k^j = N^{-1}$.
end for

favorable in both resampling quality and computational complexity, see [21]. A pseudocode for the *systematic resampling* algorithm is presented in Algorithm 6.

Algorithm 6 Systematic Resampling

Create a CSW $\{c^n\}_{n=1}^N$ from $\{\tilde{q}^n\}_{n=1}^N$.
Draw one sample from $\tilde{u} \sim \mathcal{U}(0,1)$.
Generate ordered uniform numbers $u_k = \frac{(k-1)+\tilde{u}}{N}$.
Set $i = 1$.
for $j = 1 : N$
    while $u_j > c_i$
        $i = i + 1$.
    end while
    $s_k^j = \tilde{s}_k^i$.
    $q_k^j = N^{-1}$.
end for

4.6.2 Simulations and Results

In this section simulations will be performed for three different resample algorithms, *simple random*, *efficient simple random* and *systematic resampling*. The aim is to get a good overview of how time is distributed between the three particle filter steps when a time efficient resample method is chosen. The quality of the estimates from the particle filter, such as N_{eff} and $RMSE$ will also be presented, these are indicators of resampling quality, the relation is complex and lots of other factors influence the particle filter estimates as well, this is not investigated in this study. However one should be able to determine that no deterioration of the quality of the estimates should occur, as a result of a more time efficient resample algorithm.

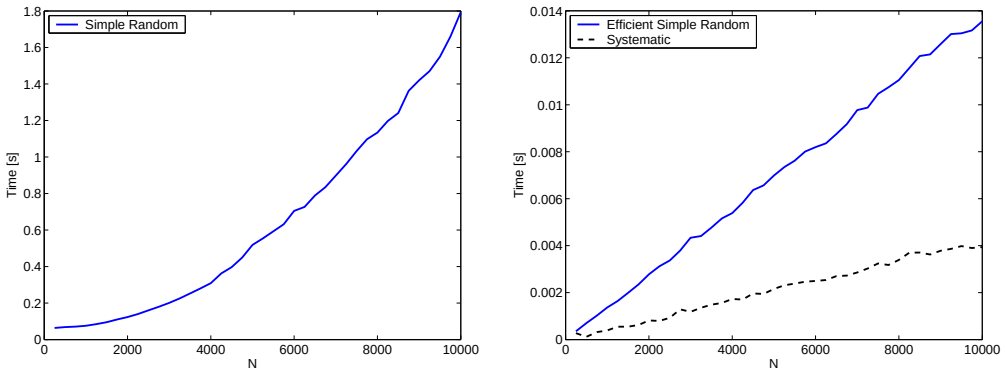
The same target and sensor model as in Section 4.5.2 are used, and the Measurements are generated based on the same scenario. From Setup 1 to Setup 2 ,and

3 in Section 4.5.2 no noticeable differences in quality was observed, therefore the likelihood will be calculated as in Setup 3 since it is the most time efficient one. The resample methods will be referred to as in Table 4.2.

Table 4.2. Three PF with different resample methods. The measurement update step in Setup 3, see Table 4.1, will be used to calculate the weights.

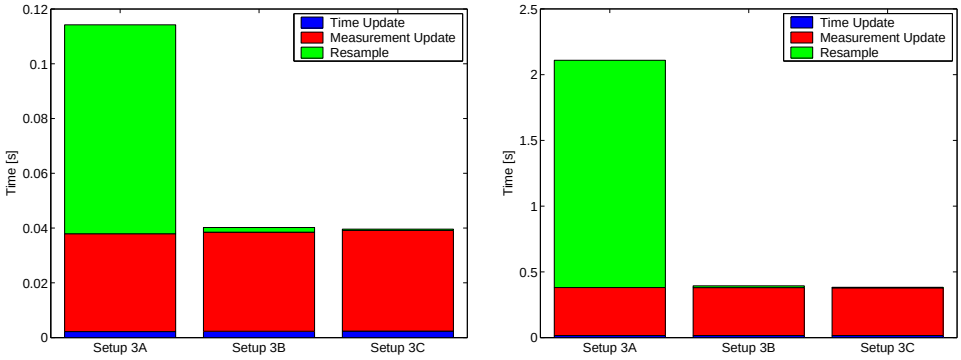
Resampling Method	**size(S)**, $(r \times d \times b)$	**Setup**
Simple Random	$3 \times 7 \times 1$	3A
Efficient Simple Random	$3 \times 7 \times 1$	3B
Systematic	$3 \times 7 \times 1$	3C

First simulations are performed to measure the computational time for the different resample steps only, see Figure 4.9. The weights are generated by drawing N samples from a uniform distribution $\mathcal{U}(0,1)$ and then normalized to sum to one. The number of particles are increased in steps of 250 and in each step the time is calculated as a mean value from 100 runs. Notice the grade on the vertical axis, as expected there is a magnificent difference in calculation time between the *simple random* with the complexity of $\mathcal{O}(N^2)$ and *efficient simple random* and *systematic resampling*. These two are both of complexity $\mathcal{O}(n)$. Nevertheless it is apparent that *systematic resampling* is a more time efficient one. This since only one random sample needs to be drawn in *systematic resampling*, compared to N samples in *efficient simple random resampling*. Furthermore, Figure 4.10 shows how



(a) (b)

Figure 4.9. Computational effort as a function of the number of particles for three different resample methods, weights are generated by drawing N samples from a uniform distribution.



(a) (b)

Figure 4.10. Total computational time for one iteration in a particle filter for TBD, using three different resample methods: (a) $N = 1000$ particles, (b) $N = 10000$ particles.

the main computational load moves from the Resample step to the Measurement Update step. One might not be surprised that a straight on Resample step as the *simple random resampling* was replaced with a great gain of time, since it is of complexity $\mathcal{O}(N^2)$. However, results from this section states that it should not be considered for a time efficient particle filter implementation. Although the outcome

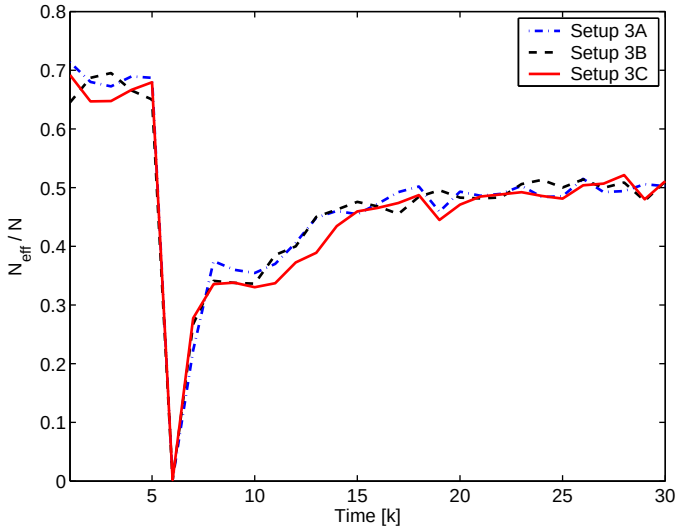
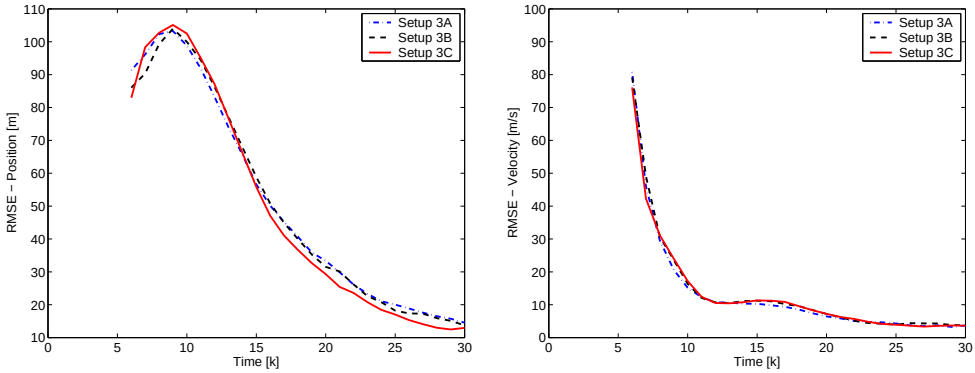


Figure 4.11. Number of effective particles $\frac{N_{eff}}{N}$, $N = 1000$.



(a) (b)

Figure 4.12. RMSE over 100 Monte Carlo runs, $N = 1000$ particles: (a) RMSE in position, (b) RMSE in velocity.

of the particle filter estimations are an indirect way of measuring the quality of the resample step, one can still see from Figure 6.4, that the particle filter behaves somewhat equivalent, independent of which resample step that is applied, neither are there any reasons to suspect a degeneracy trend by looking at the N_{eff} in Figure 4.11. However the quality of the resampling steps can be further investigated and the definition of quality for this purpose extended.

4.7 Conclusions

In this section the goal was to decrease the computational time for a particle filter for TBD, so that large off line Monte Carlo simulations could be run within a reasonable frame of time, but also to investigate a real time feasibility. It is shown that a great gain of time can be made without affecting the performance of the particle filter, see Figure 4.13. From Setup 1 to Setup 3C most of the contribution (99.5%) from an expected target is considered when calculating the likelihood, in other words, the effectiveness is made with a very little approximation, and still results in a ≈ 15 times faster implementation. However, even more efficient implementation can be done by using a larger approximation, this of course finally affects the outcome of the estimations, but it is shown that a minimum number of cells, as few as one, can be used with a somewhat maintained quality of the particle filter. The key to a faster implementation turned out to be the calculation of the likelihood, the measurement update step. Although a great gain of time is made in this step more work should be done, since it is still this part that mainly restricts an even more time efficient implementation.

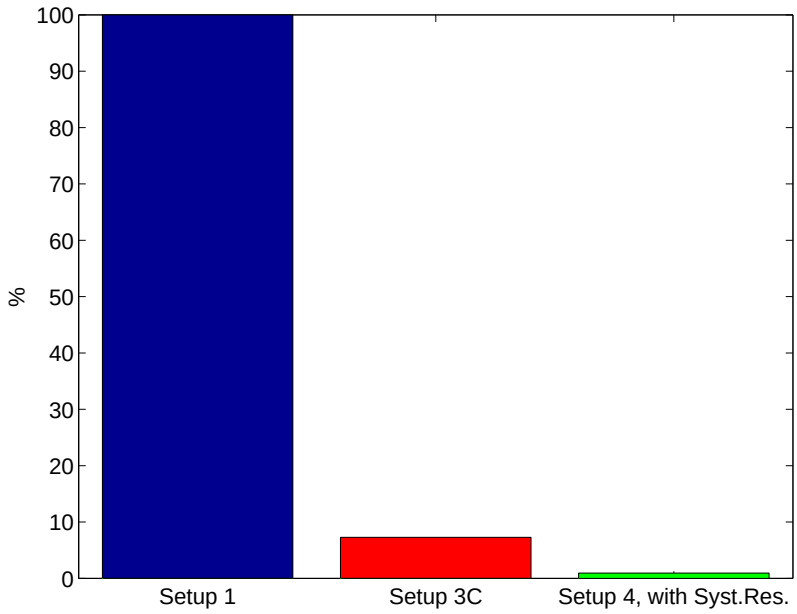


Figure 4.13. The computational time for one iteration in three different particle filters given in relative measures, with respect to Setup 1, the non optimised particle filter implementation that this paper started from. Setup 3C is an implementation with the Systematic Resampling and a restricted likelihood in the measurement update step, where 99.5% of the expected "contribution" from a target is considered. Setup 4 is the same as Setup 3C but with only one cell considered in the likelihood calculation, that is 28.2% of the contribution.

Chapter 5

Estimation of the radar cross section (RCS)

5.1 Introduction

In this chapter an algorithm will be presented that estimates the amount of power received by the radar originating from a target. This is done by including the *radar cross section* (RCS) properties of a target into the state vector. The RCS property is in proportion to the area of a target that is exposed to the sensor, so over time a fluctuating characteristic can be due to a turning or manoeuvring target, see Section 2.1.2. The algorithm will be applied on simulated radar measurements. Monte Carlo simulations will be used on a target with a constant RCS value to determine the performance of detection and tracking, and finally results from a single run will be presented to test the ability to capture a fluctuating behaviour. Measurements will be generated for a considerably high number of sensor cells, and therefore we will make use of the time efficient methods investigated in Chapter 4 to maintain a real time feasibility.

5.2 System Setup

Consider the general and nonlinear discrete time system

$$s_{k+1} = f(s_k, m_k, w_k), \qquad k \in \mathbb{N} \tag{5.1}$$

$$Prob\{m_k = i | m_{k-1} = j\} = \Pi_{ij} \tag{5.2}$$

$$z_k = h(s_k, m_k, v_k), \qquad k \in \mathbb{N} \tag{5.3}$$

where

- $s_k \in \mathbb{R}^n$ is the state of the system
- $m_k \in \mathbb{N}$ is the modal state of the system

- $z_k \in \mathbb{R}^p$ are the measurements
- w_k is the process noise
- v_k is the measurement noise
- f is the system dynamic function
- h is the measurement function
- Π is the Markov transition matrix

5.2.1 Target Model

The predicted state of the target moving in the $x - y$ plane is given from the dynamic target model

$$s_{k+1} = f(s_k, m_k, w_k). \tag{5.4}$$

The power observed by the radar is proportional to the RCS properties of the target according to

$$P = \frac{P_t G_t}{4\pi R^2} \times \frac{RCS}{4\pi R^2} \times A_e$$

where P_t is the transmitted power, G_t is the antenna gain and A_e is the effective antenna aperture. If one assume that the range measurement boundaries $R = \{R_{min} \dots R_{max}\}$ and that $R \gg R_{max} - R_{min}$ this can be written as

$$P = \rho \times RCS \tag{5.5}$$

where ρ is due to radar constants. Now the RCS can be included in the state space vector

$$s_k = \begin{pmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \\ RCS_k \end{pmatrix}. \tag{5.6}$$

The system dynamics is defined as

$$f(s_k, m_k) = \begin{pmatrix} 1 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} s_k. \tag{5.7}$$

The process noise w_k is assumed to be white Gaussian noise and the process noise input model is given by

$$g(s_k, m_k) = \begin{pmatrix} \frac{1}{2}(\frac{1}{3}a_{maxx})T^2 & 0 & 0 \\ 0 & \frac{1}{2}(\frac{1}{3}a_{maxy})T^2 & 0 \\ \frac{1}{3}a_{maxx}T & 0 & 0 \\ 0 & \frac{1}{3}a_{maxy}T & 0 \\ 0 & 0 & \sigma_{RCS}T \end{pmatrix} w_k \tag{5.8}$$

with maximum accelerations a_{maxx}, a_{maxy} and variance σ^2_{RCS}. This provides the posterior target state

$$s_{k+1} = \begin{pmatrix} 1 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} s_k + \begin{pmatrix} \frac{1}{2}(\frac{1}{3}a_{maxx})T^2 & 0 & 0 \\ 0 & \frac{1}{2}(\frac{1}{3}a_{maxy})T^2 & 0 \\ \frac{1}{3}a_{maxx}T & 0 & 0 \\ 0 & \frac{1}{3}a_{maxy}T & 0 \\ 0 & 0 & \sigma_{RCS}T \end{pmatrix} w_k$$

5.2.2 Measurement Model

At each discrete time k the sensor is providing a new set of measurements z_k. These measurements are the power levels in $N_r \times N_d \times N_b$ sensor cells. According to the measurement model in the single target TBD application, it consists of either a target and noise or noise only, as in (4.10)

$$z_k = v_k, \qquad m_k = 0, \tag{5.9a}$$

$$z_k = h(s_k, v_k), \qquad m_k = 1. \tag{5.9b}$$

In the first case the target is absent and therefore measurements consist of nothing but noise. In the second case there is a target present and its existence will affect the power level in each sensor cell. In this investigation a considerably number of sensor cells will be taken into account, therefore the likelihood ratio from (4.28) will be used to maintain a real time feasibility,

$$p_{\frac{s_k}{H_0}}(z_k|s_k) = \begin{cases} \prod_{ijl \in \mathbb{S}} \frac{p(z_k^{ijl}|s_k)}{p(z_k^{ijl}|H_0)}, & m_k = 1 \\ 1, & m_k = 0 \end{cases} \tag{5.10}$$

where

$$p(z_k^{ijl}|s_k) = \frac{1}{\mu_t^{ijl}} e^{-\frac{1}{\mu_t^{ijl}} z_k^{ijl}}, \tag{5.11}$$

and

$$p(z_k^{ijl}|H_0) = \frac{1}{\mu_v^{ijl}} e^{-\frac{1}{\mu_v^{ijl}} z_k^{ijl}}. \tag{5.12}$$

Furthermore,

$$\begin{aligned}\mu_v^{ijl} &= E[(z_k^{ijl}|s_k, m_k = 0)] \\ &= E[|n_{Ik} + in_{Qk}|^2] \\ &= E[n_{Ik}^2 + n_{Qk}^2] \\ &= 2\sigma_n^2. \end{aligned} \tag{5.13}$$

$$\begin{aligned}\mu_t^{ijl} &= E[(z_k^{ijl}|s_k, m_k = 1)] \\ &= E[|\tilde{A}_k e^{i\phi_k} h_{Ak}^{ijl}(s_k) + n_{Ik} + in_{Qk}|^2] \\ &= E[(\tilde{A}_k h_{Ak}^{ijl}(s_k)\cos(\phi_k) + n_{Ik})^2 + (\tilde{A}_k h_{Ak}^{ijl}\sin(\phi_k) + n_{Qk})^2] \\ &= \tilde{A}_k^2 (h_{Ak}^{ijl}(s_k))^2 + 2\sigma_n^2 \\ &= P h_{Pk}^{ijl}(s_k) + 2\sigma_n^2 \end{aligned} \tag{5.14}$$

with

$$\begin{aligned} h_{Pk}^{ijl}(s_k) &= (h_{Ak}^{ijl}(s_k))^2 \\ &= e^{-\frac{(r_i - r_k)^2}{R} L_r - \frac{(d_j - d_k)^2}{D} L_d - \frac{(b_l - b_k)^2}{B} L_b} \end{aligned} \tag{5.15}$$

which describes the power contribution of a target in every range-Doppler-bearing cell, where r_k, d_k, and b_k are given from target state s_k through (4.2.2). Notice that P_k is time dependent and also conditioned on s_k as $P_k = \rho \times RCS_k$.

5.3 Simulations and Results

This section will present results from Monte Carlo runs, where the power reflected from a target received at the radar is estimated according to the suggested method provided in this chapter. Tracking is done on a single target scenario. Where the target appears at $k = 6$, at a distance of 89.6 km, flying with a constant velocity of 200 m/s towards the sensor. In the first two cases the RCS value is constant, causing a received power at the sensor of 10 Pu and 5 Pu, the measurement noise level $2\sigma_n^2 = 1$ Pu, hence it follows that the signal to noise ratio is 10 dB and 7 dB since

$$SNR = 10\log\left(\frac{P_k}{2\sigma_n^2}\right) dB \tag{5.16}$$

In a third case the RCS value is time varying, which should correspond to the characteristics of a manoeuvring target, based on the knowledge that the power received at the radar changes in relation with the area of the object exposed to the sensor. Furthermore, the measurement space is divided into $N_r \times N_d \times N_b = 17600$

cells, a considerably number, therefore the efficient method from Chapter 4 will be used to calculate the weights, with a restricted region $\mathbb{S}$ of $3 \times 5 \times 5$ cells ($range \times Doppler \times bearing$). Initially particles are uniformly distributed in an area between $[85, 90]$ km, $[-0.34, -0.1]$ km/s in the x direction, and $[-1, 1]$ km,$[-0.2, 0.2]$ km/s in the y direction, they are also uniformly distributed over the two modes. Analysis of results and conclusions are based on 100 Monte Carlo runs. More constants used in the particle filter setting and for simulating measurements are presented in Table 5.1, and 5.2.

Table 5.1. Particle filter settings

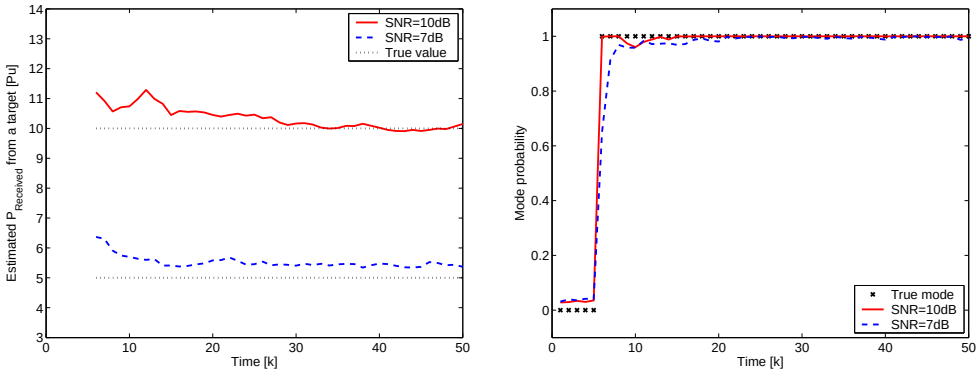
	Constant	***Value***
Number of particles	N	10000
Process noise	a_{maxx}	$10\,\mathrm{m/s^2}$
	a_{maxy}	$10\,\mathrm{m/s^2}$
	σ^2_{RCS}	1 Pu

Table 5.2. Constants used for generating measurements

	Constant	***Value***
Number of range cells	N_r	100
Number of Doppler cells	N_d	16
Number of bearing cells	N_b	11
Measurement space boundaries:		
Range		$[70, 90]$ km
Doppler		$[-0.34, -0.1]$ km/s
Bearing		$[-17.45, -17.45]$ mrad
Signal to noise ratio	SNR	10 dB/7 dB/Fluctuating
Update time	T	1 s

Tracking and Detection Performance

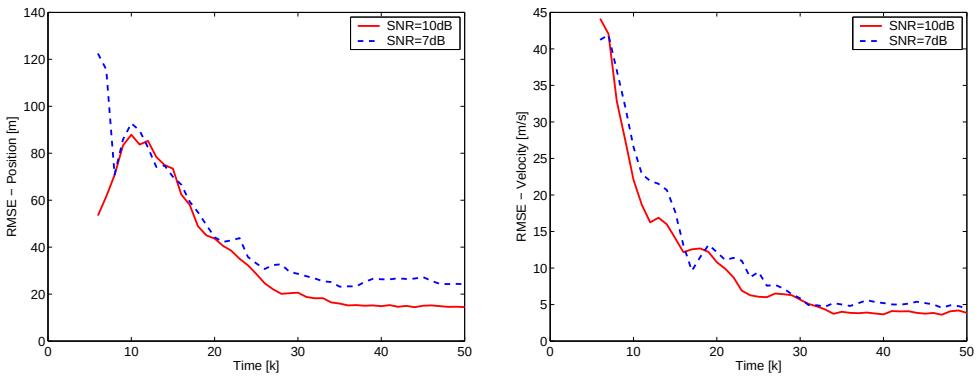
At $k = 6$ when the target appears, the mode probability is rapidly rising, see Figure 5.1 b, and the filter is well aware of the existence. Although the target is picked up good the filter is initially a bit indecisive in the estimation of the power received from the target, Figure 5.1 a. However, with time the filter learns from the data and produces an output that is closing in on the true target state. The error in position converges to a level of about 20 m depending on the SNR, a level well below the size of a range cell, Figure 5.2 a.



(a) Estimated power originating from a target.

(b) Probability of existence

Figure 5.1.



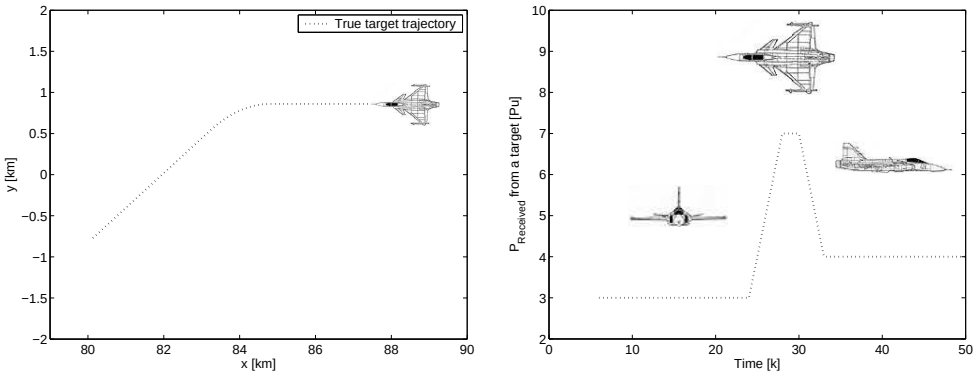
(a) Tracking accuracy in position

(b) Tracking accuracy in velocity

Figure 5.2. RMSE performance

An example of a target with a time varying RCS characteristic

In this section the particle filter will be applied on a set of measurements Z_k, that is generated based on a target scenario with time varying RCS characteristic. A target appears at a distance of about 89 km, initially moving straight towards the sensor, after 20 s it begins a left turn, and exposes its belly to the sensor. The manoeuvre is accomplished in 8 seconds, and then it is again moving straight forward but now with its side to the sensor, see Figure 5.3 a. Initially as the target is moving face



(a) True target trajectory, the radar sensor is positioned at $x = 0\,\text{km}$, $y = 0\,\text{km}$.

(b) Expected power received at the radar sensor, due to a maneuvring target.

Figure 5.3. Scenario used for generating measurements. No attempt is made to model a realistic target amplitude in the relation to the trajectory, however, one should be able to see if the principal idea of the algorithm works with a fluctuating target.

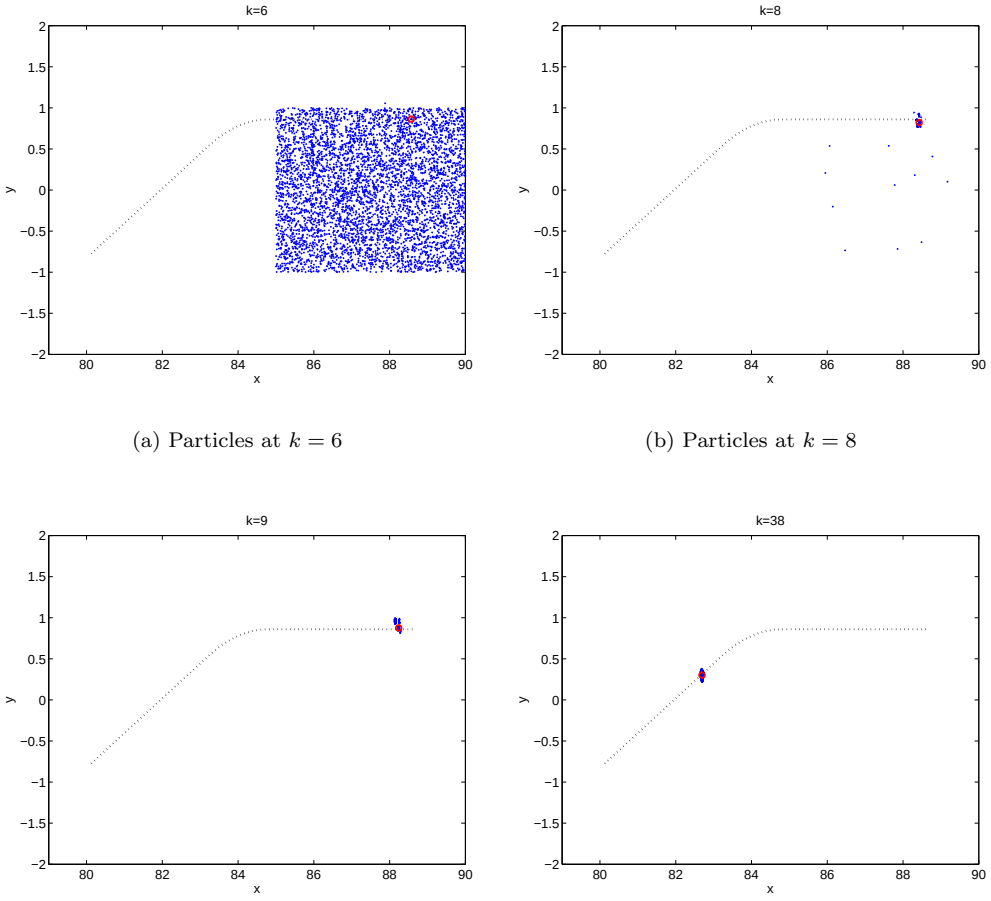
first onto the sensor it causes a received power of 3 Pu. During the turn it is rising to a peak of 7 Pu, and when the turn is accomplished it is back at a lower level at 4 Pu, Figure 5.3 b.

Now, if the particle filter described in this chapter is applied on this example, we should be able, according to the investigation of its possibility to estimate the RCS characteristics, combined with the knowledge of position and velocity, to get a descent overview of the situation.

Initially particles are uniformly distributed, see Figure 5.4. As soon as the target appears at $k = 6$ the filter is determined about its presence, see mode probability in Figure 5.6. However, it takes a few scans before particles are well gathered around the target. But once they are gathered, they stay there throughout the trajectory. The estimated trajectory is also presented in Figure 5.5 a, where the position and velocity direction is shown for every third timestep.

5.4 Conclusions

This chapter suggested a method to estimate the power originating from a target, simply by putting the RCS characteristic of a target into the state vector. Good results was achieved first when measurements from more than one bearing angle was taken into account. A more extensive investigation of whether this is necessary to maintain the observability, might contribute to this assumption. Furthermore, the algorithm should be applied on a multi target scenario to investigate its ability to perform in this situation, one could start off from two targets as the one presented



(a) Particles at $k = 6$

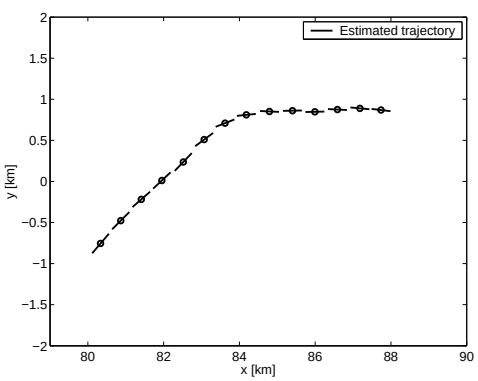
(b) Particles at $k = 8$

(c) Particles at $k = 9$

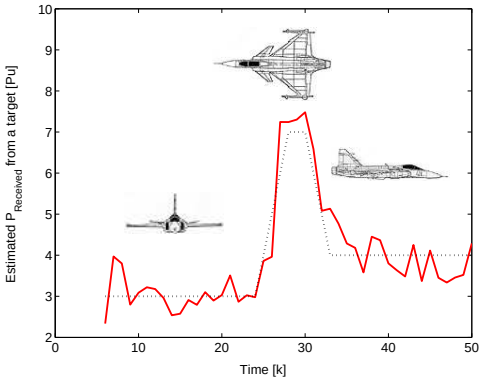
(d) Particles at $k = 38$

Figure 5.4. Distribution of particles at four different time steps.

in [7]. In the target model it is neglected that the power received by the radar changes in relation with the distance to the target. This can be implemented but would cause a nonlinear target model. The simplification might not however, give an unrealistic model or have a big impact on the result, since we are only considering a limited range of $[70, 90]$ km. It is also shown that it is possible to observe fluctuating characteristics, this provides an interesting opening where it can be combined with, for example, the manoeuvring state of a target to improve the tracking performance, or with an extended target situation, see Chapter 6.



(a) Estimated trajectory, the position and velocity direction is shown for every third time step that at target is estimated as present.



(b) Estimated received power from a target with time varying RCS characteristic.

Figure 5.5. Particle filter outputs, Compare with Figure 5.3.

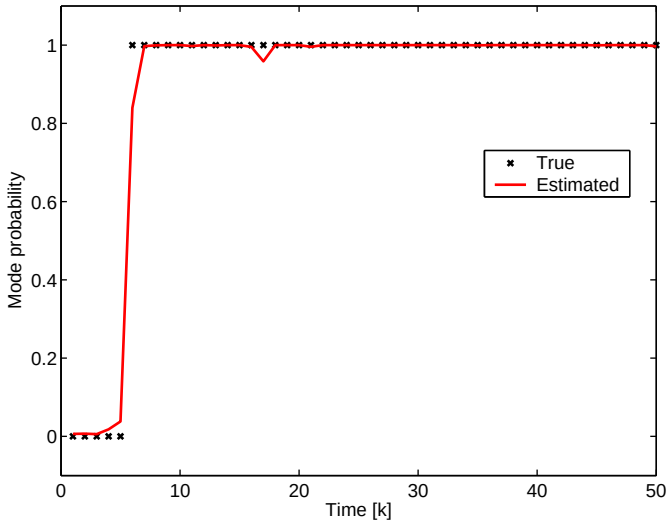


Figure 5.6. Probability of existence

Chapter 6

Tracking of Extended Targets

6.1 Introduction

In previous sections it has been assumed that each scan of measurement data consists of one reflection originating from a "point" target. This model might not be correct in all situations. The body of a target might due to its extension, reflect the radar signal from different points of its structure from time to time. There is also the possibility of multiple reflections originating from the same target at the same time. If this is taken into account in our model, it provides an opportunity to gain some knowledge about, not only position and velocity, but also its physical structure and orientation. The basic idea is to create the compound *pdf* of a measurement originating from a target as

$$p(z_k|s_k) = \int_{EXT} p(z_k|\tilde{s}_k)p(\tilde{s}_k|s_k)d\tilde{s},$$

where $p(\tilde{s}_k|s_k)$ is the *pdf* of a source $\tilde{s}_k$ given a target in state s_k, and $p(z_k|\tilde{s}_k)$ is the *pdf* of a measurement z_k arising from source $\tilde{s}_k$. EXT is the extension of a target, length, area, or volume depending on our model.

The extended target model can be applied in cases where the radar is of such a resolution, that we expect a target to occupy more than one sensor cell, and where depending on the extension, a point target model most probably leads to divergence. A general but crude solution in filtering problems with unmodeled parts, is to increase the process noise to maintain robustness of the filter.

In practice the divergence can be avoided by increasing the amount of process noise, to a level where the reflections originating from different parts of the body will be compensated with an uncertainty in the prediction of the target. However, now using an incorrect target model we can not expect an optimal estimate of the states.

The extended target model, on the other hand, should be well adaptable to measurements originating from a point target, since the length would be estimated to what is considered a point. The extended model is no longer useful to detect a body size or orientation, but the principle can be used, for example, to detect a group of targets in a tight formation splitting up. Following sections will provide an *extended target particle filter* ETPF, based on a scenario that each scan of measurement data consists of one reflection, originating from a random point of an extended target body.

Previous work has been done on plot basis, see [26]. However, in these sections an ETPF will be used on raw measurement data, the TBD approach. We will state the problem of measurements originating from an extended target, derive the Bayesian solution, and implement it as a particle filter. Results from Monte Carlo simulations as well as openings for further developments will be discussed in concluding chapters.

6.2 System Setup

As before, the problem lies in estimating the target state s_k, based on all available measurements up to time k, $z_1 \dots z_k$, with knowledge about the target model evolving according to

$$s_{k+1} = f(s_k, m_k, w_k), \qquad k \in \mathbb{N}, \tag{6.1}$$

$$Prob\{m_k = i | m_{k-1} = j\} = \Pi_{ij}, \tag{6.2}$$

and the measurement model

$$z_k = h(s_k, m_k, v_k), \qquad k \in \mathbb{N}. \tag{6.3}$$

6.3 Target Model

The reflection from a target is distributed along its body due to its extension, physical structure, and orientation. Let's introduce the *pdf* of a source $\tilde{s}_k$ given a target state s_k as $p(\tilde{s}_k|s_k)$. The shape of $p(\tilde{s}_k|s_k)$ can be modified, depending on the amount of information of a target that in an initial stage is available. In a case where the target is unknown $p(\tilde{s}_k|s_k)$ can, for example, be approximated with a normal or uniform distribution, along the estimated extension. In this approach the reflection point $\tilde{s}_k$ will be generated as

$$\tilde{s}_{k_\gamma} = s_k + G(s_k)H_X, \qquad \gamma = 1 \dots N_{NOR}, \tag{6.4}$$

where $G(s_k)$ is a vector or matrix of coefficients that is due to the dimensional extension of a target, and H_X can consist of one or several stochastic distributions. N_{NOR} are the number of sources i.e., number of reflections from a target achieved from a target at each time k. N_{NOR} can also be a stochastic variable, varying from time to time. However, as mentioned, in this setting one reflection is expected,

$N_{NOR} = 1$. Any possible input and initial knowledge of a target, concerning its extension, physical structure, or orientation will be used in $G(s_k)H_X$. In this application the target is modeled as a 1D stick pointing at the radar. Hence, it follows that s_k consists of

$$s_k = \begin{pmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \\ L_k \end{pmatrix}, \tag{6.5}$$

where L_k is the spatial extension in range or x direction. The dynamics that provides the posterior target state are similar to the one in Chapter 5,

$$s_{k+1} = \begin{pmatrix} 1 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} s_k + \begin{pmatrix} \frac{1}{2}(\frac{1}{3}a_{maxx})T^2 & 0 & 0 \\ 0 & \frac{1}{2}(\frac{1}{3}a_{maxy})T^2 & 0 \\ \frac{1}{3}a_{maxx}T & 0 & 0 \\ 0 & \frac{1}{3}a_{maxy}T & 0 \\ 0 & 0 & \sigma_l T \end{pmatrix} w_k.$$

Moreover, $\tilde{s}_k$ will be generated from s_k as

$$\tilde{s}_k = s_k + \begin{pmatrix} \frac{L_k}{2} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \mathcal{U}(-1, 1), \tag{6.6}$$

where, in this case, $\tilde{x}_k$, $\tilde{y}_k$, are the reflection point on the body of a target. The reflections are uniformly distributed in the x direction and bounded to the estimated length L_k.

6.4 Measurement Model

As before, each set of measurements achieved at time step k, consists of a target and noise or noise only,

$$z_k = v_k, \qquad m_k = 0, \tag{6.7a}$$
$$z_k = h(s_k, v_k), \qquad m_k = 1. \tag{6.7b}$$

In this setting it is assumed, that if a target exists, there is one reflection achieved at the radar due to its presence, $N_{NOR} = 1$. The reflection can originate from anywhere along its body. Hence it follows that the likelihood function is given as a compound *pdf* by the convolution

$$p(z_k|s_k) = \int_{EXT} p(z_k|\tilde{s}_k)p(\tilde{s}_k|s_k)\, d\tilde{s} \tag{6.8}$$

$p(z_k|\tilde{s}_k)$ is the *pdf* of a measurement z_k arising from the source $\tilde{s}_k$, and $p(\tilde{s}_k|s_k)$ is *pdf* of a source $\tilde{s}_k$ given a target in state s_k. Under some assumptions the integral can be replaced by analytically combining the two probability density functions, and the weights can be calculated as usual directly from $p(z_k|s_k)$. However, in a more general approach the integral (6.8) can be approximated numerically with

$$p(z_k|s_k) = \frac{1}{M}\sum_{j=1}^{M} p(z_k|\tilde{s}_k^j). \tag{6.9}$$

where M is the number of drawings, this is the same approximation as the one used in (3.12). The complexity of the measurement update step will approximately increase with a factor times M. Therefore the likelihood ratio is used again, introduced in (4.28),

$$p_{\frac{\tilde{s}_k}{H_0}}(z_k|\tilde{s}_k) = \frac{p(z_k|\tilde{s}_k)}{p(z_k|H_0)} = \begin{cases} \prod_{ijl\in\mathbb{S}} \frac{p(z_k^{ijl}|\tilde{s}_k)}{p(z_k^{ijl}|H_0)}, & m_k = 1, \\ 1, & m_k = 0, \end{cases} \tag{6.10}$$

where

$$p(z_k^{ijl}|\tilde{s}_k) = \frac{1}{\mu_t^{ijl}} e^{-\frac{z_k^{ijl}}{\mu_t^{ijl}}}, \tag{6.11}$$

and

$$p(z_k^{ijl}|H_0) = \frac{1}{\mu_v^{ijl}} e^{-\frac{z_k^{ijl}}{\mu_v^{ijl}}}, \tag{6.12}$$

with

$$\mu_t^{ijl} = P\sum_{\gamma=1}^{N_{NOR}} e^{-\frac{(r_i - r_{k_\gamma})^2}{R}L_r - \frac{(d_j - d_{k_\gamma})^2}{D}L_d - \frac{(b_l - b_{k_\gamma})^2}{B}L_b} + 2\sigma_n^2, \tag{6.13}$$

and

$$\mu_v^{ijl} = 2\sigma_n^2. \tag{6.14}$$

Furthermore,

$$r_{k_\gamma} = \sqrt{\tilde{x}_{k_\gamma}^2 + \tilde{y}_{k_\gamma}^2}, \tag{6.15a}$$

$$d_{k_\gamma} = \frac{\tilde{x}_{k_\gamma}\dot{\tilde{x}}_{k_\gamma} + \tilde{y}_{k_\gamma}\dot{\tilde{y}}_{k_\gamma}}{\sqrt{\tilde{x}_{k_\gamma}^2 + \tilde{y}_{k_\gamma}^2}}, \tag{6.15b}$$

$$b_{k_\gamma} = \arctan\left(\frac{\tilde{y}_{k_\gamma}}{\tilde{x}_{k_\gamma}}\right). \tag{6.15c}$$

6.5 Simulations

The tracking of the extended target is done with three different particle filters, a point target PF and two ETPF's. One of the ETPF is well tuned in and at this level of process noise a point target particle filter certainly leads to divergence, this might be reason enough to choose an ETPF instead of a point target PF, on measurements from a radar where "extended" targets are expected. However, in a second ETPF the amount of process noise is increased to a level where a point target PF does not diverge to state that a point target PF is outperformed in every aspects by an ETPF. Finally the ETPF will be applied on measurements with different SNR values. All filters are applied on "extended" target measurements and results and conclusions are based on 100 Monte Carlo runs.

6.5.1 Constants

During simulations constants has been used as in Table 6.1 and Table 6.2. Measurements are generated based on that from time $k = 6$, there is one reflection originating from a target, and over time reflections are uniformly distributed along the extension of its body. Target appears at a distance of 9.6 km and moves with a constant velocity of 10m/s towards the sensor. Initially particles are uniformly distributed in the state space, in an area between $[9.45, 9.65]$ km, $[-0.034, -0.005]$ km/s in the x direction, and $[-0.05, 0.05]$ km, $[-0.01, 0.01]$ km/s in the y direction, over a target length of $[5, 30]$ m, and they are also uniformly distributed over the two modes.

Table 6.1. Particle filter settings.

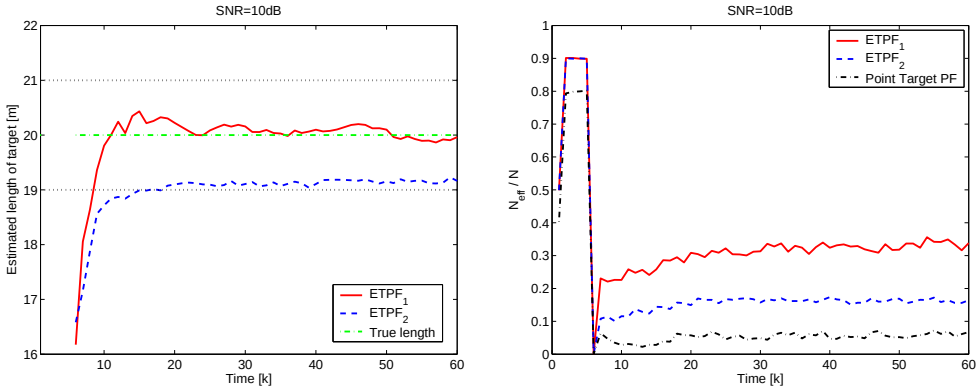
	Constant	**Point target PF**	**ETPF$_1$**	**ETPF$_2$**
Number of particles	N	1000	1000	1000
Number of drawings used in (6.9)	M	10	10	10
Process noise	a_{maxx}	$20\,\mathrm{m/s^2}$	$2\,\mathrm{m/s^2}$	$20\,\mathrm{m/s^2}$
	a_{maxy}	$2\,\mathrm{m/s^2}$	$2\,\mathrm{m/s^2}$	$2\,\mathrm{m/s^2}$
	σ_l^2	-	1.5 m	1.5 m

6.6 Results

Figure 6.2 shows estimated position and extension of a target at four different scans. As soon as a target appears in scan 6.2 a it is picked up by the ETPF. It is though indecisive about its length and position, due to the fact that only one reflection is achieved, considered originating from a target. Already in scan 6.2 b we have a decent estimate of the length, and in scan 6.2 c particles are well gathered around

Table 6.2. Constants used for generating measurements.

	textbf*Constant*	
Number of range cells	N_r	500
Number of doppler cells	N_d	16
Number of bearing cells	N_b	1
Measurement space bounderies:		
Range		$[9, 10]$ km
Doppler		$[-0.034, -0.005]$ km/s
Bearing		-
Signal to noise ratio	SNR	10 dB/7 dB
Length of target	L	20 m
Update time	T	1 s



(a) Estimated length from the ETPF's against true value from 100 Monte Carlo runs, dotted black lines are a reminder of the resolution of a range cell.

(b) A comparison in N_{eff} between two ETPF's and a point target PF, when they are applied on extended target measurements.

Figure 6.1. Estimated length and number of effective particles.

the true center of the target. Notice that from scan to scan particles are centered and the target is estimated without particles "jumping" after the reflection point. In Figure 6.1a it is shown how well the two ETPF performs in estimating the length. After about 5 scans the filters are already quite certain. As expected the ETPF_1 setting produces a better estimate, due to the fact that a low value of a_{maxx} is preventing particles from "jumping" after reflection points and consider those as the center of a target. In ETPF_2 a higher value of a_{maxx} causes the filter to estimate deviating reflections as an accelerating behavior (The target model in (6.1) evolves

according to a constant velocity model, however, acceleration is possible in relation to the process noise.). The same reasoning can be used to explain why ETPF_2 stays at a level of about one meter below the true target length.

Now consider Figure 6.3, initially ETPF_2 appears to converge as good as ETPF_1 or even better, this can be explained with the quite limited number of particles that are used, causing a bias in the scan of target appearance $k = 6$, see Figure 6.2 a. This bias will be eliminated faster when using a higher process noise, since we are less certain in the prediction of our particles, but with time a lower process noise makes a visible difference in estimating both position and velocity. However, besides the fact that ETPF_1 and ETPF_2 are able to estimate the length, they are both outperforming the point target PF in every aspect of estimation accuracy see Figure 6.3.

As expected, it also appears a clear degeneracy trend in N_{eff} for the point target PF, see Figure 6.1b (For an explanation of the N_{eff} see Section 4.6). In mode transition at $k = 6$ there is a plain dip in N_{eff} for all filters , initially no target is present and particles are uniformly distributed, bounded by the state space limits, leaving only a few particles in the vicinity of a targets appearance. Furthermore, before resampling the average number of particles in mode "target present" $m_k = 1$, is only N times transition probability "birth of target" P_b. In this case 10% of the total number of particles. A simple solution to this problem would be to considerably increase the number of particles. However, this leads to an unnecessary large number of particles in the mode with the highest probability and of course, a time inefficient implementation. An alternative and recent solution is presented in [8]. Based on that the number of particles in each mode is fully controlled by the designer and independent of the actual transition probability, without distorting or violating the Markov property.

6.7 Conclusions

In this chapter the problem of estimating a target with a spatial extent was considered. Once again a particle filter was used to solve the filtering problem. The proposed algorithm modeled the target as a 1D stick with a spatial extent in range. It is shown that an extended target model outperforms the point target model on measurements from a sensor where target has occupied more that one sensor cell. Although improvements in tracking accuracy is an important issue it might not be the main reason to chose a ETPF. The filter could do a even better job to capture features and identify targets. In order to do this the "stick" could be replaced with a 2D area or a volume.

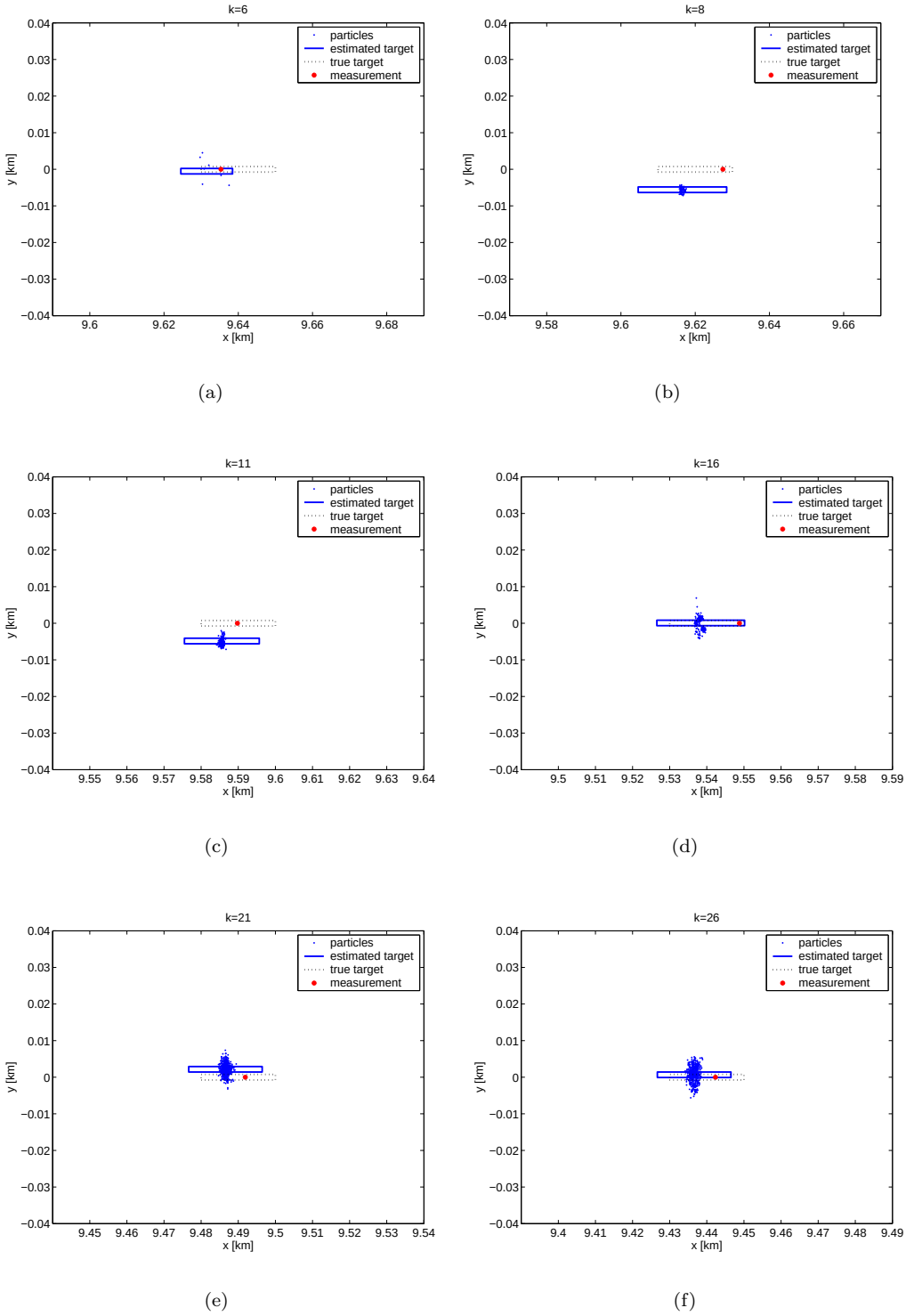


Figure 6.2. The position and length of a target are estimated using an *extended target particle filter* (ETPF). Although only one reflection is originating from a target at each scan, the filter manage to get a decent estimate of both the length and position in just a few scans. Notice how the particles are well gathered around the center of the estimated target and not "jumping" after the reflection point.

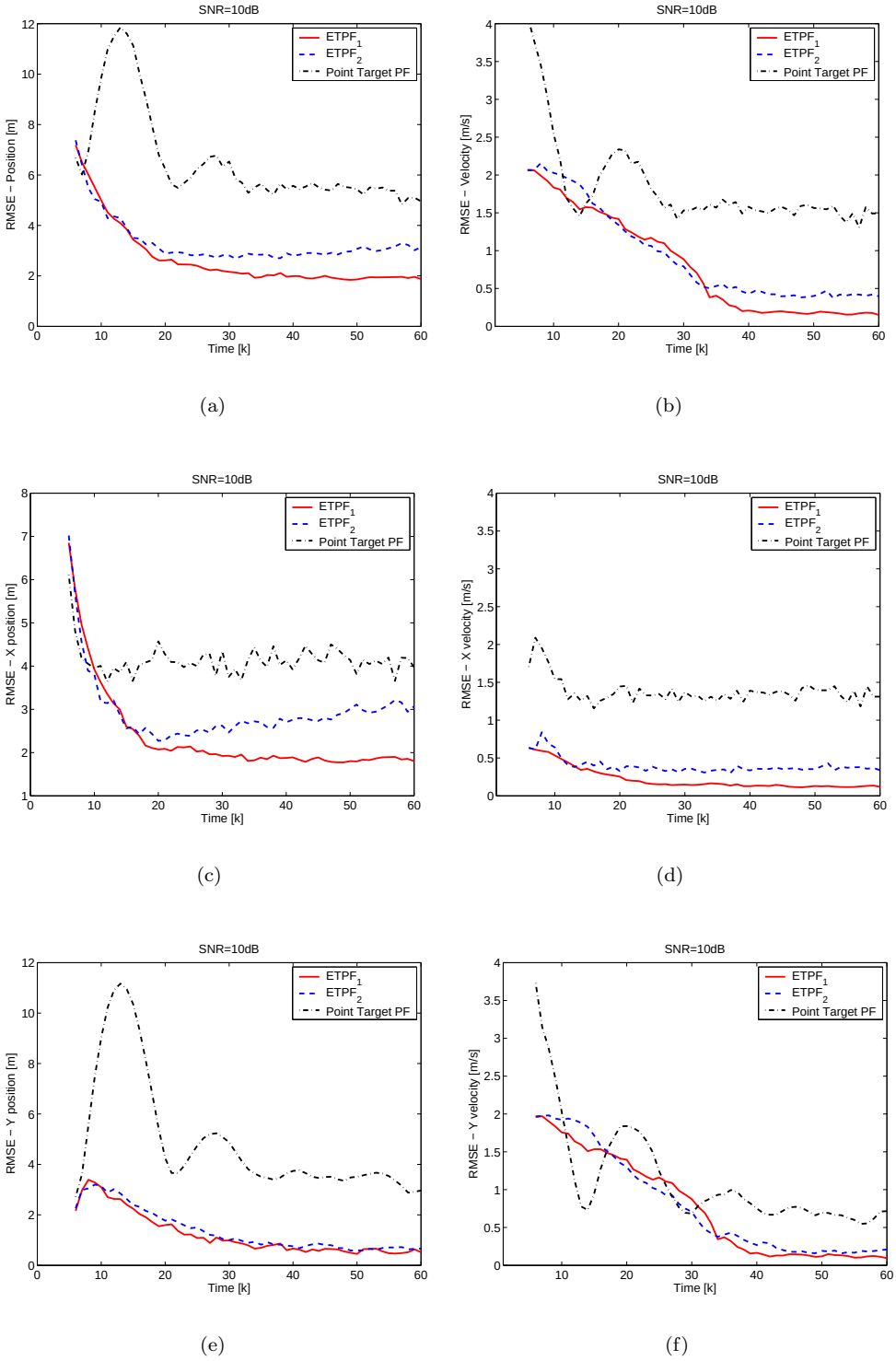


Figure 6.3. ETPF's and point target PF performance for (a),(c), and (e) position, and (b),(d), and (f) velocity. 100 Monte Carlo runs.

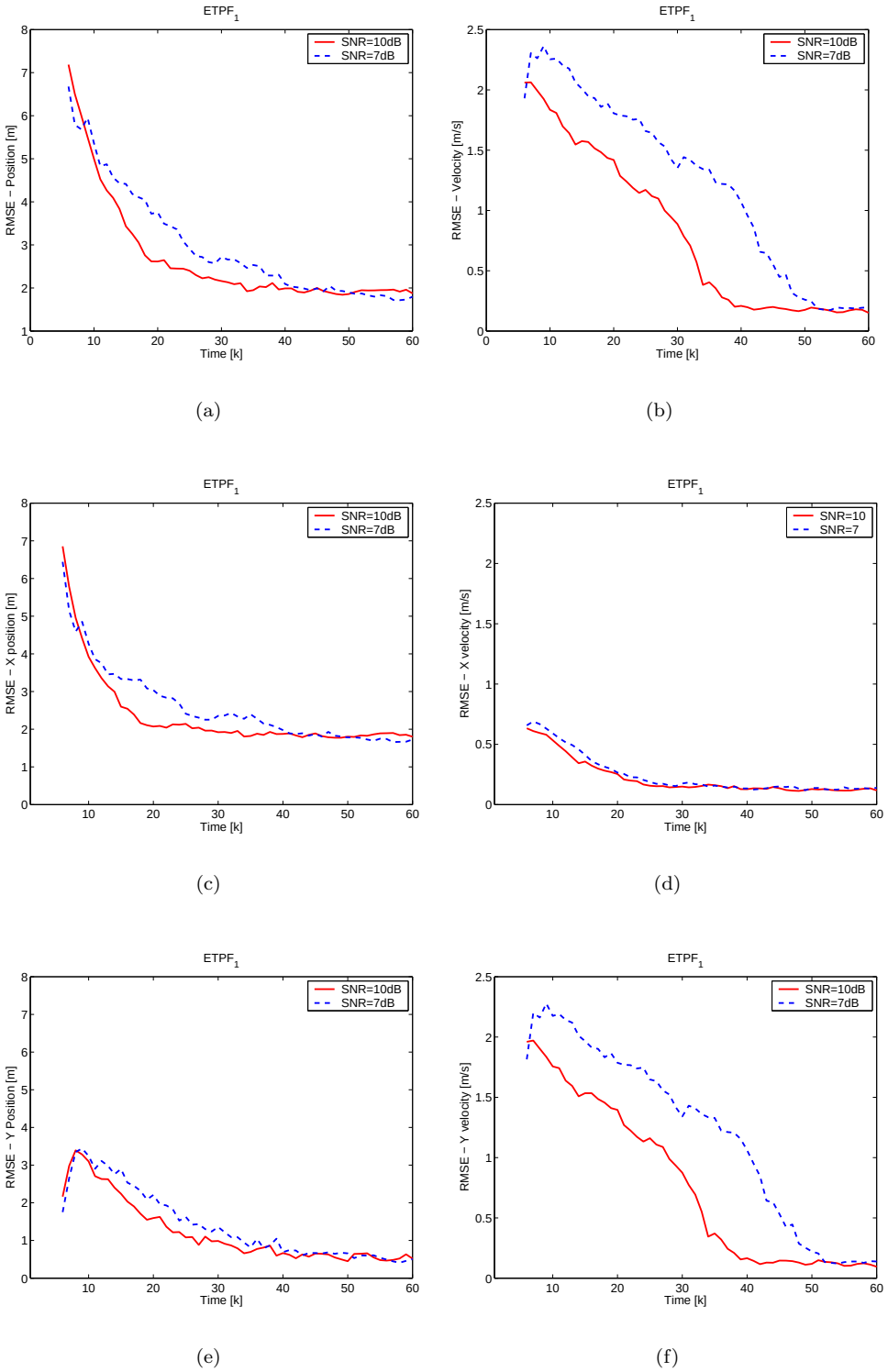


Figure 6.4. ETPF performance for (a),(c), and (e) position, and (b),(d), and (f) velocity for two differnt SNR values. 100 Monte Carlo runs.

Chapter 7

Concluding Remarks

7.1 Conclusions

This Master's thesis has investigated the use of particle filters on radar measurements, in a *track before detect* (TBD) approach. The work was divided into two major problems, a time efficient implementation and new functional features. The suggestions to improvements of the implementation showed to be many times faster, which indicates that there are openings for a real time feasibility, even for particle filters with an initially high computational demand. The goal was also to estimate the *radar cross section* (RCS), as well as the physical extension of a target. It is shown that it is possible to capture these characteristics, easily by modifying or extending our dynamical model, and that particle filters are a simple, yet good solution to the filtering problem arising from this. Furthermore, the estimates of the RCS and the extent of a target showed on good accuracy for high signal to noise ratio as well as for low.

7.2 Further Studies

In the efficient implementation an extensive complexity analysis of the particle filter should be done, in order to state that the investigated methods can be applied with a successful outcome in general. This can be done by using the equivalent flop measure, introduced in [17]. The quality of the estimates must be further investigated and the definition for this purpose extended. In particular the affect in detection performance. In the case where the power originating from a target was estimated, good results was achieved first when more than one bearing angle was taken into account. A study of whether this is necessary to maintain the observability of the system might contribute to this assumption. The ability to observe fluctuating characteristics provides an interesting opening that it can be combined with, for example, the manoeuvring state of a target to improve the tracking performance, or with an extended target situation. The *extended target particle filter* (ETPF)

should be applied on manoeuvring target measurements. Future work should also investigate its ability to capture features, and identify targets. In order to do this the 1D "stick" can be replaced with a target model of multidimensional extension. Moreover, all models should be further developed in a multi target context, see [7].

Appendix A

Bayes' Rule

Let x and y denote two stochastic variables with probability density functions $p(x)$ and $p(y)$ respectively. The joint density can be written as

$$p(x, y) = p(x|y)p(y), \tag{A.1}$$

or equivalent as

$$p(x, y) = p(y|x)p(x). \tag{A.2}$$

Combining (A.1) and (A.2), and solving for the conditional probability $p(x|y)$ yields

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}. \tag{A.3}$$

This expression is usually referred to as *Bayes' rule* or *Bayes' theorem*. The denominator $p(y)$ can be obtained by marginalizing out x in (A.2)

$$p(y) = \int_{\mathbb{R}^n} p(x, y)dx = \int_{\mathbb{R}^n} p(y|x)p(x)dx. \tag{A.4}$$

Bibliography

[1] S. Blackman and R. Popoli.
Design and Analysis of Modern Tracking Systems.
New York: Artech House Publishers, 1999.

[2] Y. Bar-Shalom and X.R. Li.
Estimation and Tracking Principles, Techniques and Software.
Artech House, Norwood, MA, 1993.

[3] R. Ristic, S. Arulampalam and N. Gordon.
Beyond the Kalman Filter - Particle Filters for Tracking Applications.
Boston - London: Artech House 2004.

[4] Y. Boers and J.N. Driessen.
Particle filter based detection for tracking.
In: Proceedings American Control Conference, Vol. 6, 2001, p. 4393-4397.

[5] Y. Boers H. Driessen and K. Grimmerink.
Particle filter based detection schemes.
In: Signal and Data Processing of Small Targets 2002, SPIE Vol. 4728, 2002.

[6] Y. Boers.
On the number of samples to be drawn in particle filtering.
In: Proceedings of the IEE Colloquium on Target Tracking: Algorithms and Applications, London UK, 11-12 nov. 1999, p 5/1-5/6.

[7] Boers, Y. and H. Driessen, F. Vershure, W.P.M.H. Heemels, A. Juloski.
A multi target track before detect application.
In: Proc. 2003 Workshop on Multi-Object Tracking, Madison, WI, June 2003.

[8] J.N Driessen and Y. Boers.
An Efficient Particle Filter for Jump Markov Nonlinear Systems.
In: Proceedings of *The IEE Colloquium on Target Tracking, Sussex*, UK,March 2004.

[9] A.F.M. Smith and A.E. Gelfand.
Bayesian Statistics without tears: A Sampling-Resampling Perspective.
The American Statician, vol. 46, p. 84-88, 1992.

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

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

[12] N.J. Gordon, D.J. Salmond and A.F.M. Smith.
Noval approach to nonlinear/non-Gaussian Bayesian state estimation.
In: IEE Proceedings-F, Vol. 140, No. 2, April 1993, p.107-113.

[13] N. Gordon, S. Maskell and T. Kirubarajan.
Efficient Particle Filtes for Joint Tracking and Classification.
In: Signal and Data Processing of Small Targets 2002, SPIE Vol.4728, 2002.

[14] K. Grimmerink.
Particle Filter Design and Detection for a Track Before Decect Radar Application.
University of Twente. Faculty of Applied Mathematics, Department of Systems, Signals and Control.

[15] R. Karlsson.
Simulation Based Methods for Target Tracking. Lic. Thesis No. 930.
Linköping universitet, Linköping, Sweden, 2002.

[16] R. Karlsson.
Particle Filtering for Positioning and Tracking Applications. PhD Thesis No. 924.
Linköping universitet, Linköping, Sweden, 2005.

[17] R. Karlsson, T. Schön and F. Gustafsson
Complexity Analysis of the Marginalized Particle Filter
Published: LiTH-ISY-R-2680, Feb 2005.

[18] D.J. Salmond, D. Fisher and N.J. Gordon.
Tracking in the presence of spurious objects and clutter.
In: Signal and Data Processing of Small Targets 1998, SPIE Vol. 3373, p. 460-474, 1998.

[19] M.I. Skolnik.
Introduction to radar systems.
McGraw-Hill, Inc, Singapore, 2nd ed., 1981.

[20] F. Gustavsson, L. Ljung and M. Millnert.
Signalbehandling.
Studentlitteratur, Lund, 2001

[21] J. D. Hol.
Resampling in particle filters.
Linköping University, Linköping Sweden, 2004.

[22] G. Kitagawa
Monte Carlo filters and smoothers for non-Gaussian nonlinear state space models.
Journal of Computational and Graphical Statistics, 5(1):1-25, 1996.

[23] J.S. Liu and R. Chen.
Sequential Monte Carlo methods for dynamic systems.
Journal of the American Statistical Association, 93(443):1032-1044, 1998.

[24] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp.
A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking.
IEEE Proceedings on Signal Processing, 50(2):174-188, 2002, 1998.

[25] H.V. Poor.
An introduction to signal detection and estimation.
springer-Verlag, New York NY, 1994.

[26] J. Vermaak, N. Ikoma and S.J. Godsill.
Extended Object Tracking using Particle Techniques.
Aerospace 2004