5-27-2016

Tracking in Several Real-World Scenarios

Kevin Romeo
kromeo42@gmail.com

Tracking in Several Real-World Scenarios

Kevin Romeo, Ph.D.

University of Connecticut, 2016

Tracking algorithms are used in many applications to provide estimates of states (position, velocity, etc.) of targets from noisy measurements. These estimates can be used for predicting future target states. Some possible targets that may be of interest (and that we will consider here) include aircraft, ships, and missiles. This dissertation looks at several real-world scenarios and develops new tracking algorithms to accurately and efficiently solve these problems. These algorithms are compared to the current state-of-the-art and shown to be superior in position and velocity RMSE, or in computational complexity.

We investigate three real-world tracking scenarios. First, we develop a new algorithm with low computational complexity for tracking closely spaced targets. Second, we apply a regularized particle filter to the banana and contact lens problems using a multidimensional version of the Epanechnikov kernel for state vectors, developed in the course of the research for this dissertation. Finally, we develop a generalization of the ML-PMHT and apply it to several Over-the-Horizon radar scenarios.

Tracking in Several Real-World Scenarios

Kevin Romeo

B.S., University of Connecticut, 2009

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

at the

University of Connecticut

2016

APPROVAL PAGE

Doctor of Philosophy Dissertation

Tracking in Several Real-World Scenarios

Presented by

Kevin Romeo, B.S.

Major Advisor _____
Yaakov Bar-Shalom

Major Advisor _____
Peter Willett

Associate Advisor _____
Krishna Pattipati

University of Connecticut

2016

To my family

ACKNOWLEDGEMENTS

I would like to express special appreciation and gratitude to my co-major advisors Prof. Yaakov Bar-Shalom and Prof. Peter Willett for their continuous support of my study and research. Their guidance has helped me in all aspects of my research and writing of this thesis. Thank you for encouragement; your brilliant advice, comments, and suggestions have been priceless.

I also thank my labmates for their advice and stimulating discussions.

A special thanks to my friends and family for supporting and motivating me throughout this work, and in my life in general.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Outline of the Dissertation

In Chapter 2 we look at various algorithms for approximating the target-measurement association probabilities of the Joint Probabilistic Data Association Filter (JPDAF). We consider their computational complexity and compare their performance with respect to the Mean Optimal Subpattern Assignment (MOSPA) statistic in a scenario involving closely-spaced targets.

In Chapter 3, we present a new algorithm for approximating the target-measurement association probabilities of the Joint Probabilistic Data Association Filter (JPDAF). This algorithm is designed to robustify the JPDAF against track coalescence which can greatly degrade the performance of the JPDAF and other approximate algorithms. It is based on the works of Roecker and the JPDAF* of Blom and Bloem.

In Chapter 4, we present an approach for tracking with a high-bandwidth radar in long range scenarios. We consider both 2-D and 3-D measurements, in polar and r-u-v, respectively. We show that in these scenarios the extended Kalman filter is not desirable as it suffers from major consistency problems; and most flavors of particle filter suffer from a loss of diversity among particles after resampling. This leads to sample impoverishment and the divergence of the filter. However, a regularized particle filter will be shown to avoid this diversity problem while producing consistent results. The regularization is accomplished using a multidimensional version of the Epanechnikov kernel for state vectors developed in the course of the research for this dissertation.

In Chapters 5 and 6, we present a generalization of the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker (ML-PMHT), which is a Deep Track Extractor (DTE), which is then applied to several Over-The-Horizon radar (OTHR) scenarios: a surface target, a constant altitude target, and a constant vertical acceleration target. We use both an ideal reflection model and a refraction-based ionosphere model through the application of a ray tracing algorithm. Each scan can contain multiple measurements originating from each target; each of these target-originated measurements takes one of four possible round-trip paths and any of these may fail to be detected. Also a large number of false alarms, indistinguishable from the target-originated measurements, are present due to the necessary low detection threshold for very low SNR targets.

The contributions of the dissertation are summarized in Chapter 7.

1.2 Publications

The publications directly related to this dissertation are:

Journals:

- Kevin Romeo, Yaakov Bar-Shalom, and Peter Willett, "Detecting Low SNR Tracks with OTHR Using a Refraction Model," Submitted to *IEEE Transactions on Aerospace and Electronic Systems*, 2016.

- Kevin Romeo, Yaakov Bar-Shalom, and Peter Willett, "Fusion of Multipath Data with ML-PMHT for Very Low SNR Track Detection in an OTHR," *Journal of Advances in Information Fusion*, Accepted for publication, May 2015. To appear December 2015.

- Kevin Romeo, Peter Willett, and Yaakov Bar-Shalom, "Particle Filter Tracking for the Banana and Contact Lens Problems," IEEE Transactions on Aerospace and Electronic Systems, Accepted for publication, October 2014. To appear October 2015.

Proceedings:

- Kevin Romeo, Yaakov Bar-Shalom, and Peter Willett, "Low SNR Track Detection with OTHR Based on a Refraction Model," in *Proceedings of the 2016 IEEE Radar Conference*, Philadelphia, May 2016.

- Kevin Romeo, Yaakov Bar-Shalom, and Peter Willett, "Data Fusion with ML-PMHT for Very Low SNR Track Detection in an OTHR," in *Proceedings of the 18th International Conference on Information Fusion*, Washington, D.C., July 2015.

- Kevin Romeo, Peter K. Willett , and Yaakov Bar-Shalom, "Particle Filter for Long Range Radar in RUV," in *Proceedings of SPIE Signal and Data Processing of Small Targets 2014*, vol. 9092, July 2014.

- Kevin Romeo, Peter K. Willett, and Yaakov Bar-Shalom, "Particle Filter Tracking for the Banana Problem," in *Proceedings of SPIE Signal and Data Processing of Small Targets 2013*, vol. 8857, October 2013.

- Kevin Romeo, David F. Crouse, Yaakov Bar-Shalom, and Peter Willett, "A Fast Coalescence-Avoiding JPDAF," in *Proceedings of SPIE Signal and Data Processing of Small Targets 2012*, vol. 8393, June 2012.

- Kevin Romeo, Peter Willett, and Yaakov Bar-Shalom, "Particle Filter Tracking for Long Range Radars," in *Proceedings of SPIE Signal and Data Processing of Small Targets 2012*, vol. 8393, June 2012.

- Kevin Romeo, David F. Crouse, Yaakov Bar-Shalom, and Peter Willett, "The JPDAF in Practical Systems: Approximations," in *Proceedings of SPIE Signal and Data Processing of Small Targets 2010*, vol. 7698, April 2010.

Chapter 2

The JPDAF in Practical Systems: Approximations

2.1 Introduction

Two basic methods for tracking multiple targets are the Multiple Hypothesis Tracking (MHT) and the Joint Probabilistic Data Association Filter (JPDAF). In the MHT, multiple possible track extensions from a set of feasible measurements are kept for a certain amount of time. During this time frame, additional measurements are collected, which in turn are used to eliminate some of the multiple possible track extensions that are highly improbable given these additional measurements. In the JPDAF, tracks are updated with a weighted sum of the feasible measurements. The weights are determined from the probabilities of validated measurements extending a track; all possible sets of measurement-to-track combinations and their probabilities must be calculated. [50]

Both the MHT and the JPDAF use a significant amount of processing for large numbers of targets. In the MHT, this is due to the increasing number of possible track extensions that must be kept over time and also because there is no guarantee of when possible track extensions will be eliminated. In the JPDAF, increasing the number of targets increases the number of combinations, and the processing time, accordingly. In attempts to lower the computational cost of the JPDAF, many fast algorithms, ad hoc formulations, and suboptimal calculations have been developed. These algorithms (hopefully) increase in their accuracy as their computational cost approaches that of the optimal JPDAF. [50]

2.2 JPDAF [6]

The JPDAF is an algorithm for tracking multiple targets in clutter, which assumes there is a known number of targets with initialized tracks. Measurements from a target can appear close to neighboring targets, causing persistent interference. It also assumes that a target can only produce at most one measurement, with a known probability. Additionally, a measurement could not have been produced by more than one target. The filter uses the state estimate and covariance for each target to summarize the past. The dynamic and measurement models for target t, which may differ between targets, are given as

$$x_t(k+1) = F_t(k)x_t(k) + v_t(k) \tag{2.1}$$

and

$$z_t(k) = H_t(k)x_t(k) + w_t(k) \tag{2.2}$$

respectively. At time index k for target t, $x_t(k)$ is the state vector, $z_t(k)$ is the measurement vector, $H_t(k)$ is the measurement matrix, and $F_t(k)$ is the state transition matrix. The measurement noise $w_t(k)$ and the process noise $v_t(k)$ are uncorrelated, zero mean, and normally distributed with covariances $R(k)$ and $Q(k)$, respectively. The state estimation is done separately for each target.

The most computationally expensive part of the JPDAF is determining the joint association probabilities between measurements and targets. This can be done by finding all the feasible joint association events: an event where one or no target produced each measurement and no measurement is from more than one target. The probability of the joint association event θ given the measurements up to time k is found as

$$P\{\theta(k)|Z^k\} = \frac{1}{c}\prod_j \{\lambda^{-1} f_{t_j}[z_j(k)]\}^{\tau_j(\theta)} \prod_t (P_D^t)^{\delta_t(\theta)}(1 - P_D^t)^{1-\delta_t(\theta)} \tag{2.3}$$

where

$$f_{t_j}[z_j(k)] = \mathcal{N}[z_j(k); \hat{z}_{t_j}(k|k-1), S_{t_j}(k)] \tag{2.4}$$

and $\mathcal{N}[x; \mu, P]$ denotes the Gaussian distribution function with parameter x, mean μ, and variance P. We use Z^k to denote the set of measurements up to and including time index k and c as a normalizing constant. In event θ, t_j is the index of the target that measurement j is associated with. At time k, λ is the known spatial density of clutter, $z_j(k)$ is the jth measurement, and P_D^t is the probability

of detection of target t; $\tau_j(\theta) = 1$ if measurement j is associated with any target in the event θ, and $\delta_t(\theta) = 1$ if target t is associated with any measurement (they are equal to zero otherwise); $\hat{z}_{t_j}(k|k-1)$ is the predicted measurement for target t_j given as

$$\hat{z}_{t_j}(k|k-1) = H_{t_j}(k)\hat{x}_{t_j}(k|k-1) \tag{2.5}$$

with the predicted state

$$\hat{x}_{t_j}(k|k-1) = F_{t_j}(k-1)\hat{x}_{t_j}(k-1|k-1) \tag{2.6}$$

The associated innovation covariance, $S_{t_j}(k)$, is

$$S_{t_j}(k) = H_{t_j}(k)P_{t_j}(k|k-1)H_{t_j}(k)' + R(k) \tag{2.7}$$

where $P_{t_j}(k|k-1)$ is the covariance of the predicted state from the covariance at the previous time index, $P_{t_j}(k-1|k-1)$, and is given by

$$P_{t_j}(k|k-1) = F_{t_j}(k-1)P_{t_j}(k-1|k-1)F_{t_j}(k-1)' + Q(k-1) \tag{2.8}$$

The probability β_{jt} that measurement j is associated with target t is then given by summing the joint probabilities of the joint events where this target to measurement association occurs. This summation can be written as

$$\beta_{jt}(k) = \sum_{\theta} P\{\theta(k)|Z^k\}\hat{\omega}_{jt}(\theta, k) \tag{2.9}$$

where $\hat{\omega}_{jt}(\theta, k) = 1$ if measurement j is associated with target t in event $\theta(k)$ and equal to zero otherwise.

The state estimation for target t is carried out as follows

$$\hat{x}_t(k|k) = \hat{x}_t(k|k-1) + W_t(k)\nu_t(k) \tag{2.10}$$

$$\nu_t(k) = \sum_j \beta_{jt}\nu_{jt}(k) \tag{2.11}$$

$$\nu_{jt}(k) = z_j(k) - \hat{z}_t(k|k-1) \tag{2.12}$$

$$W_t(k) = P_t(k|k-1)H_t(k)'S_t(k)^{-1} \tag{2.13}$$

where $\hat{x}_t(k|k)$ is the state estimate of target t at time k, $\nu_{jt}(k)$ is innovation, $\nu_t(k)$ is the combined innovation, and $W_t(k)$ is the filter gain for target t. The covariance update is

$$P_t(k|k) = \beta_{0t}(k)P_t(k|k-1) + [1-\beta_{0t}(k)]P_t^c(k|k) + \tilde{P}_t(k) \tag{2.14}$$

$$P_t^c(k|k) = P_t(k|k-1) - W_t(k)S_t(k)W_t(k)' \tag{2.15}$$

$$\tilde{P}_t(k) = W_t(k)\left[\sum_j \beta_{jt}(k)\nu_{jt}(k)\nu_{jt}(k)' - \nu_t(k)\nu_t(k)'\right]W_t(k)' \tag{2.16}$$

where $P_t(k|k)$ is the covariance associated with the updated state, $P_t^c(k|k)$ is the covariance of the state updated with the correct measurement, $\tilde{P}_t(k)$ is the spread of the innovations, and $\beta_{0t}(k)$ is the probability that target t is not associated with any measurement at time k.

The computational requirements of the JPDAF can be reduced somewhat by first gating the measurements. The true measurement is assumed to be normally distributed, namely,

$$p[z(k)|Z^{k-1}] = \mathcal{N}[z(k); \hat{z}(k|k-1), S(k)] \tag{2.17}$$

We can remove from consideration the measurements that are outside a threshold γ using the gating equation

$$[z_j(k) - \hat{z}_t(k|k-1)]' S_t(k)^{-1} [z_j(k) - \hat{z}_t(k|k-1)] \leq \gamma \tag{2.18}$$

This gating step can be applied to all of the approximate algorithms that follow.

2.3 Low-Complexity Approximations

2.3.1 Fitzgerald's First Algorithm: The Cheap JPDAF [26]

The cheap JPDAF is a simplified form of the JPDAF from Section 2.2. The optimal JPDAF updates tracks with a weighted average of all the measurements that can possibly be associated with that track. The weights are determined by the association probabilities between the tracks and the measurements. In a multitarget scenario these association probabilities can easily become costly for computation in a practical system. To approximate the probability of association β_{jt} between a track t and a measurement j, the cheap JPDAF employs the ad hoc formula

$$\beta_{jt} = \frac{G_{jt}}{S_{ij} + S_{rt} - G_{jt} + B} \tag{2.19}$$

with G_{jt} proportional to the Gaussian likelihood function, given as

$$G_{jt} = \frac{1}{\sqrt{\det A_{jt}}} \exp\left(-\frac{1}{2} \nu_{jt}' A_{jt}^{-1} \nu_{jt}\right) \tag{2.20}$$

where ν_{jt} is the vector of measurement residuals for measurement j and track t and A_{jt} is the covariance matrix of the residuals; S_{ij} and S_{rt} are the sum of

all G_{jt}'s for track t and the sum of all G_{jt}'s for measurement j, respectively. The value of G_{jt} represents how well track t fits with measurement j. The above formula can be used with a relatively low computational cost and it reduces to the correct form in the case of only a single track. The bias term B in the denominator should be set to a nonzero constant when the probability of detection is less than 1 or when the tracks are in the presence of clutter. In the simulation that follows we choose $B = 0$.

2.3.2 Fitzgerald's Second Algorithm: The Approximate Nearest Neighbor JPDAF [26]

The approximate nearest neighbor JPDAF forgoes the use of a weighted average to update the tracks. Instead, the association probabilities are used to match tracks to measurements one-to-one. For a given track, the measurement with the highest association probability from (2.19) is matched to that track. The track and matched measurement are removed from consideration while the procedure is repeated for the next track until all tracks have been matched. This is similar to the standard nearest neighbor filter except that instead of using distance to determine the closeness between a track and measurement, the approximate association probability is used. At a higher computational cost we can instead use the global nearest neighbor method and maximize the sum of the association probabilities to achieve a better result.

2.3.3 Quan, Hongcai, Peide and Zhou's Algorithm [48]

The difference between the cheap JPDAF in Section 2.3.1 and the modified version below is the inclusion of an amendment coefficient

$$r_{jt} = \frac{G_{jt}}{\sum_{l=1}^{m} G_{lt}} \tag{2.21}$$

which produces the new formula for approximating the association probabilities,

$$\beta_{jt} = \frac{G_{jt}}{S_{ij} + \sum_{s=1}^{m} r_{js} G_{js} - r_{jt} G_{jt} + B} \tag{2.22}$$

2.3.4 Roecker and Phillis' Algorithm [51]

This suboptimal JPDAF simplifies the form of the probability of the joint events in the optimal JPDAF by assuming that the probability of detection is close to 1. The probability of the joint event $\theta(k)$ from the optimal filter in (2.3) is now reduced to the form

$$P\{\theta(k)|Z^k\} = \frac{1}{c} \prod_j \{\lambda^{-1} f_{t_j}[z_j(k)]\}^{\tau_j(\theta)} \tag{2.23}$$

This can be reduced further by combining the normalization constant c with the term λ^{-1}, as all joint events now have the same number of clutter measurements. This results in the form

$$P\{\theta(k)|Z^k\} = \frac{1}{c} \prod_j \{f_{t_j}[z_j(k)]\}^{\tau_j(\theta)} \tag{2.24}$$

While the algorithms from sections 2.3.1, 2.3.2, and 2.3.3 replace the joint events with single events to reduce complexity, this algorithm uses partial joint events,

a subset of the joint events used by the JPDAF. The events in this subset can be chosen, and the approximate joint association probabilities calculated, using the steps that follow.

1. For each track t, create a list of all measurements j that lie within the gate of t:

   - $L_t = \{j \text{ is in the gate of } t\}$

2. For each measurement j, create a list of all tracks t that contain measurement j within their gate:

   - $L_j = \{t\text{'s gate contains measurement } j\}$

3. For each track t, create a list of all other tracks that contain at least one measurement in the gate of t:

   - $\text{LOT}_t = \left(\bigcup_{j \in L_t} L_j \right) \notin t$

4. For each measurement j and track t, calculate the joint partial event, H_{jt}, by following the steps below.

   (a) For each track t_i in LOT_t consider each measurement in L_{t_i} and find:

   - $M_{jt_i} = \max_{h \in L_{t_i},\, h \neq j} (G_{ht_i})$

   (b) If $M_{jt_i} = 0$, j is the only measurement in the gate of t_i, thus:

   - $M_{jt_i} = G_{jt_i}$

(c) If $\text{LOT}_t = \emptyset$, there are no tracks that share a gated measurement with t, thus:

- $H_{jt} = G_{jt}$

(d) If $\text{LOT}_t \neq \emptyset$:

- $H_{jt} = G_{jt} \sum_{t_i \in LOT_t} M_{jt_i}$

5. The suboptimal joint association probability for track t and measurement j is now given by:

$$\beta_{jt} = \frac{H_{jt}}{B + \sum_i H_{it}} \tag{2.25}$$

The parameter B here is used in the same way as in (2.19).

2.3.5 The Bakhtiar and Alavi Algorithm [5]

This algorithm notes that the expression for the joint association probabilities may be decomposed to

$$\beta_{jt} = G_{jt} F_{jt} \tag{2.26}$$

F_{jt} is a function of all $G_{l\tau}$ where $l \neq j$ and $\tau \neq t$. We define the likelihood matrix for m measurements and n targets to be

$$P = \begin{bmatrix} G_{01} & G_{02} & \cdots & G_{0n} \\ G_{11} & G_{12} & \cdots & G_{1n} \\ \vdots & \vdots & & \vdots \\ G_{m1} & G_{m2} & \cdots & G_{mn} \end{bmatrix} \tag{2.27}$$

To find an approximation of F_{jt} we take the product of the sum of each column, omitting row j and column t. For example,

$$F_{11} = (G_{02}+G_{22}+G_{32}+\cdots+G_{m2})(G_{03}+G_{23}+G_{33}+\cdots+G_{m3})\cdots(G_{0n}+G_{2n}+G_{3n}+\cdots+G_{mn}) \tag{2.28}$$

In the case of only two targets this approximate F_{jt} is equivalent to the actual F_{jt}. For more than two targets it differs by a small amount. The procedure for determining β_{jt} is

$$\beta_{jt} = \frac{1}{c} G_{jt} \prod_{\tau=1;\tau\neq t}^{n} \left(\sum_{l=0;l\neq j}^{m} G_{l\tau} + G_{0\tau}\delta_{0j} \right) \tag{2.29}$$

The term $G_{0\tau}\delta(j)$ is included since row 0 is not removed when calculating β_{0t}. δ_{0j} is the Kronecker delta function. In the simulation that follows we choose $G_{0j} = 0$ for all j.

2.4 Medium-Complexity Approximations

2.4.1 Roecker's Algorithm [50]

This near optimal JPDAF algorithm uses full joint association events for probability calculations while the faster and less accurate algorithms do not. Instead of calculating all possible joint events like the optimal JPDAF, this near optimal JPDAF only calculates the highly probable joint events; this is a subset of the set of events the optimal JPDAF would use. The joint events that are not calculated will not have a significant effect on the probability calculations. This

allows for improvements in speed and computational requirements without losing much accuracy.

The subset of joint association events is generated by constructing joint events from the more probable single events. Each single event association between a target t and a measurement j is scored based on the statistical distance between the predicted measurement of t and the measurement j. Once this scoring is done, the algorithm to produce the subset of joint events is as follows.

1. Initialization: Find the initial "best" assignment, θ_1; $\ell = 1$

2. For each combination of marginal events, L, in θ_1:

   (a) $\ell \leftarrow \ell + 1$

   (b) Find the "best" assignment, θ_ℓ, such that $L \nsubseteq \theta_\ell$.

At this time the algorithm can repeat the entire procedure above, but remove the two lowest scoring single events at a time instead of one. Now each combination of two events can be removed once. This will create additional joint events and will lead to a more accurate approximation of the joint association probabilities. The number of removed events can continue to be increased in this way if a more accurate result is desired. At any time the algorithm can stop producing new joint events. The probability of each event is found using (2.3), and the joint association probabilities are found using (2.9). In the simulation that follows we stop the algorithm after no more single events can be removed.

2.4.2 Oh and Sastry's Algorithm [42, 43]

Markov chain Monte Carlo (MCMC) is a general method which can be applied here to approximate the joint association probabilities needed by the JPDAF. At some joint association event ω in the set of all feasible joint association events Ω, we choose another event ω' and accept it with probability

$$A(\omega, \omega') = \min\left(1, \frac{\pi(\omega')}{\pi(\omega)}\right) \tag{2.30}$$

where

$$\pi(\omega) = \frac{1}{Z}\lambda^{N-|\omega|}P_D^{|\omega|}(1-P_D)^{K-|\omega|}\prod_j f_{t_j}[z_j(k)]^{\tau_j(\omega)} \tag{2.31}$$

In the above Z is a normalizing constant, N is the number of measurements, and K is the number of targets. Here $|\omega|$ is the number of targets that are associated with measurements in ω. The product is over all associations between targets t and measurements j in the event ω.

There are three valid transitions that can occur between ω and ω':

1. Deletion move:

   The deletion move removes an existing association in ω to create ω'.

2. Addition move:

   The addition move adds a new association in ω to create ω' if the chosen new association does not conflict with the associations in ω. A conflict occurs when adding the new association would result in one target being

associated with two measurements or two targets being associated with the same measurement.

3. Switch move:

   If the chosen new association conflicts with what exists already in ω, the conflicting old association is removed, and the new association is added.

With these transitions defined, the MCMC data association (MCMCDA) algorithm can be carried out as follows:

- Initialize $\beta_{jt} = 0$, $\forall j$, $\forall k$.
- Initialize ω by choosing a joint association event randomly from Ω, defining ω as this event.
- For a set number of runs, n_{mc}:
  - Choose any valid association event from Ω at random and apply the appropriate transition move to create ω'.
  - Make $\omega = \omega'$ with probability $A(\omega, \omega')$ from (2.30).
  - For every association event in ω, $\beta_{jt} = \beta_{jt} + 1/n_{mc}$.

The resulting β's will be approximations of the values calculated in the JPDAF. In the simulation that follows we choose $n_{mc} = 100$.

2.5 Simulations

2.5.1 The MOSPA Statistic [66]

To produce a measure of algorithm accuracy we employ the Optimal Subpattern Assignment (OSPA) statistic. This metric gives an evaluation of the performance of a tracking algorithm. It allows tracks to switch and penalizes an algorithm for finding more or fewer tracks than actually exist. In our simulations, however, the trackers always know exactly how many tracks there are. The statistic needs two parameters, c and p, where c is the cutoff parameter which limits the penalty assigned for false or lost tracks, or for true tracks that are not being tracked, and p determines how much of a penalty is assigned for estimates not close to any true target. We use the values $p = 1$ and $c = 150\,\mathrm{m}$ in the simulation that follows.

The OSPA statistic between the set of estimated state vectors $\hat{x}(k|k)$ and the set of true target states $x(k)$ is given by

$$\bar{d}^{(c)}(\hat{x}(k|k), x(k)) = \left(\frac{1}{T} \min_{\pi \in \Pi} \sum_{t=1}^{T} d^{(c)}(\hat{x}(k|k), x_{\pi(t)}(k))^p \right)^{1/p} \tag{2.32}$$

with

$$d^{(c)}(\hat{x}(k|k), x(k)) = \min\left[c, d(\hat{x}(k|k), x(k))\right] \tag{2.33}$$

where T is the number of targets and Π is the set of all permutations of the true target states. The OSPA finds the assignment between the true states and the estimated states at each time k that yields the smallest sum of errors. Over a number of Monte Carlo runs we take the average of these errors to find the mean

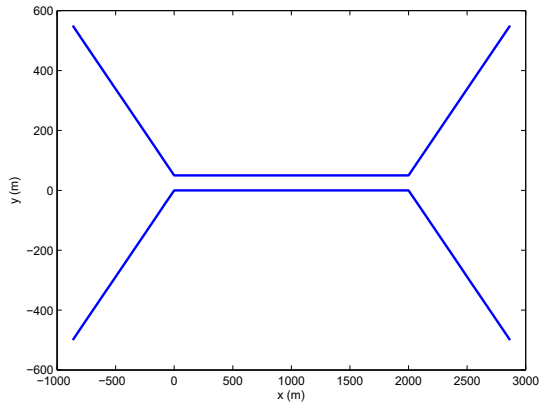


Figure 2.1: The first simulated scenario.

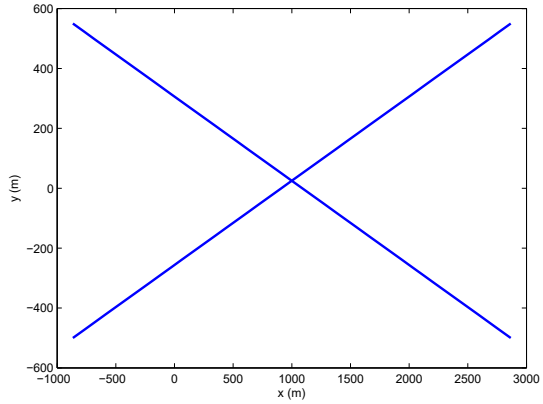


Figure 2.2: The second simulated scenario.

OSPA (MOSPA). A lower relative MOSPA statistic implies that a particular tracker performs better than another one in that given scenario.

2.5.2 Scenarios

In order to establish a comparison between the low and medium complexity approximation algorithms of the JPDAF in this paper, we simulate two simple scenarios. The first scenario, similar to that in [18], consists of two targets approaching each other, moving parallel to each other at a close distance at

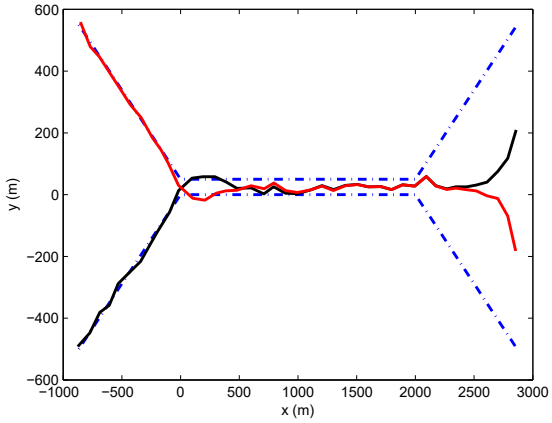


Figure 2.3: Examples of tracks made by the optimal JPDAF. The dashed lines are the true target tracks.

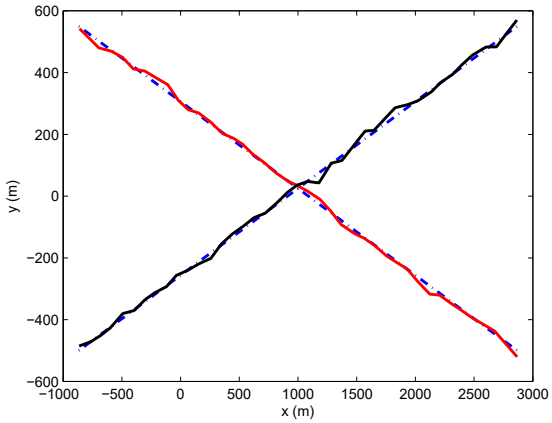


Figure 2.4: Examples of tracks made by the optimal JPDAF. The dashed lines are the true target tracks.

$k = 11$, and then separating $k = 31$. The two targets in the second scenario travel with constant velocities at a shallow angle to each other and cross paths halfway through their trajectories at $k = 21$. In both scenarios the targets have a constant speed of $v = 100\,\mathrm{m/s}$ with measurements taken at a sampling rate of $1\,\mathrm{Hz}$. The probability of detection for both targets in both scenarios is 90%. The covariance matrix of the measurements is

$$R = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \tag{2.34}$$

The standard deviations are $\sigma_x = \sigma_y = 10\,\mathrm{m}$. Using the state vector ordered as $[x\ y\ \dot{x}\ \dot{y}]'$, the measurement matrix is then

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.35}$$

and the state transition matrix for the chosen sampling period is

$$F = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.36}$$

The process noise is given by

$$Q = \begin{bmatrix} \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 1 & 0 \\ 0 & \frac{1}{2} & 0 & 1 \end{bmatrix} \tilde{q} \tag{2.37}$$

and we choose $\sqrt{\tilde{q}} = 50$ for the first scenario and $\sqrt{\tilde{q}} = 1$ for the second. The chosen values for $\sqrt{\tilde{q}}$ are based on the maximum change in velocity in a sampling period in each scenario. The simulated tracks did not have added process noise; $\sqrt{\tilde{q}}$ is used only to account for maneuvers. The number of clutter points at each time k is Poisson distributed with mean $\lambda = 10^{-6}/\mathrm{m}^2$. These points are generated uniformly in the rectangular area given by $x \in [-1370\,\mathrm{m},\ 3370\,\mathrm{m}]$ and $y \in [-1000\,\mathrm{m},\ 1050\,\mathrm{m}]$. The gate threshold is chosen to be $\gamma = 9$. Track initialization was done using two point differencing on the first two measurements from each target. The initial covariance for each track was

$$P = \begin{bmatrix} R & R \\ R & 2R \end{bmatrix} \tag{2.38}$$

The resulting MOSPA over time for each tracking algorithm is shown in Figures 2.5 and 2.6. The running times for each algorithm in scenario 1 are shown in Table 2.1.

2.6 Conclusions

From the MOSPA, we see that the algorithm [5] from Section 2.3.5 performs closest to that of the optimal JPDAF in both of the two scenarios. It is also clear that the simple addition of the amendment coefficient to the cheap JPDAF algorithm of Section 2.3.1 to create the algorithm [48] from Section 2.3.3 made a significant improvement.

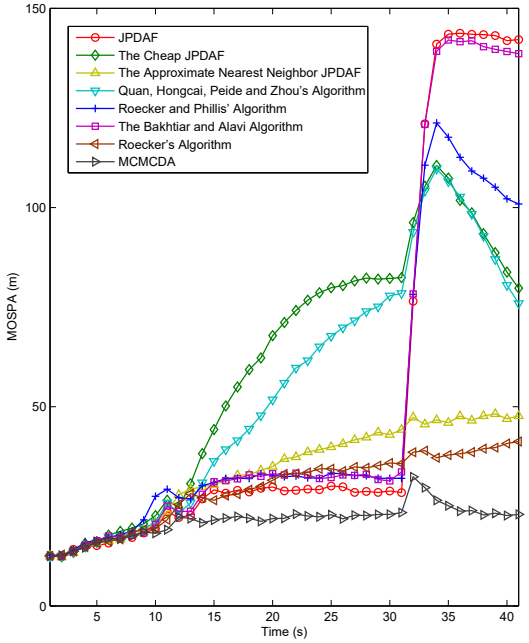


Figure 2.5: The MOSPA performance of the trackers in scenario one

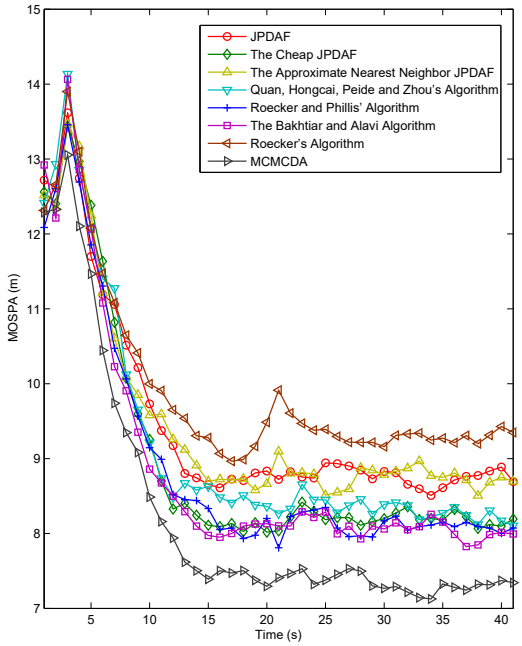


Figure 2.6: The MOSPA performance of the trackers scenario two.

The Approximate Nearest Neighbor JPDAF	2.6497
The Cheap JPDAF	2.6632
The Bakhtiar and Alavi Algorithm	2.7777
Roecker's Algorithm	2.9035
Quan, Hongcai, Peide and Zhou's Algorithm	3.0271
Roecker and Phillis' Algorithm	3.7976
JPDAF	8.9050
MCMCDA	41.8296

Table 2.1: Total computation times for scenario 1 in seconds for 50 runs

All of the algorithms, with the exception the ones from Sections 2.3.2, 2.4.1, and 2.4.2, had problems with track coalescence in the first scenario when the targets began to move apart. An example of this is seen in Figure 2.3, where the two tracks merged into one during the time the targets were close together. During the split the tracks were influenced equally by the measurements from both targets. The approximate nearest neighbor did not feel the effect of coalescence because it simply would not allow the two tracks to update using measurements from both targets at the same time.

The MCMCDA has the best performance in terms of the MOSPA statistic, but it has a computation time nearly 15 times greater than the approximate algorithms.

Chapter 3

A Fast Coalescence-Avoiding JPDAF

3.1 Introduction

In this chapter we present a new algorithm for approximating the target-measurement association probabilities of the Joint Probabilistic Data Association Filter (JPDAF). This algorithm is designed to robustify the JPDAF against track coalescence which can greatly degrade the performance of the JPDAF and other approximate algorithms. It is based on the works of Roecker and the JPDAF* of Blom and Bloem. We compare our new algorithm with the two it is based on, as well as the "cheap JPDAF" and the Set JPDAF, and show that it offers a significant improvement in computational complexity over the JPDAF* and Set JPDAF, and improvement in tracking error over the other algorithms. Their performance comparison is with respect to the Mean Optimal Subpattern Assignment (MOSPA) statistic in scenarios involving several closely-spaced targets. A consistency comparison of the various algorithms considered is also presented.

There are many different approaches to multiple target tracking, some of which are described in [8, 10]. The method explored in this paper is the Joint Probabilistic Data Association Filter (JPDAF). In the JPDAF, tracks are updated with a weighted sum of the feasible (validated) measurements from the current time.[1] The weights are determined from the probabilities of validated measurements extending a track. Advances in algorithms, for instance those discussed in [41, 37], have led to large reductions in the computational complexity of the JPDAF.

In the JPDAF, increasing the number of targets increases the number of feasible combinations of measurements and targets, and the processing time, accordingly. In attempts to lower the computational cost of the JPDAF, many fast algorithms, *ad hoc* formulations, and suboptimal calculations have been developed. We compared the performance of several of these approximate algorithms in [55]; the majority of the algorithms suffered from track coalescence. Figure 3.1 shows an example of track coalescence in a three target scenario using the conventional JPDAF.

One approximate JPDAF algorithm developed by Roecker [50] has a significantly lower computational cost for large numbers of targets when compared to the conventional JPDAF. Another algorithm, the JPDAF* [13], has nearly the

[1] This is in contrast to the Multiple Hypothesis Tracker (MHT), which carries out associations over a window of time and has a significantly higher complexity than the JPDAF. For a large number of targets, even the complexity of the JPDAF can become excessive, which motivates the need for faster versions.

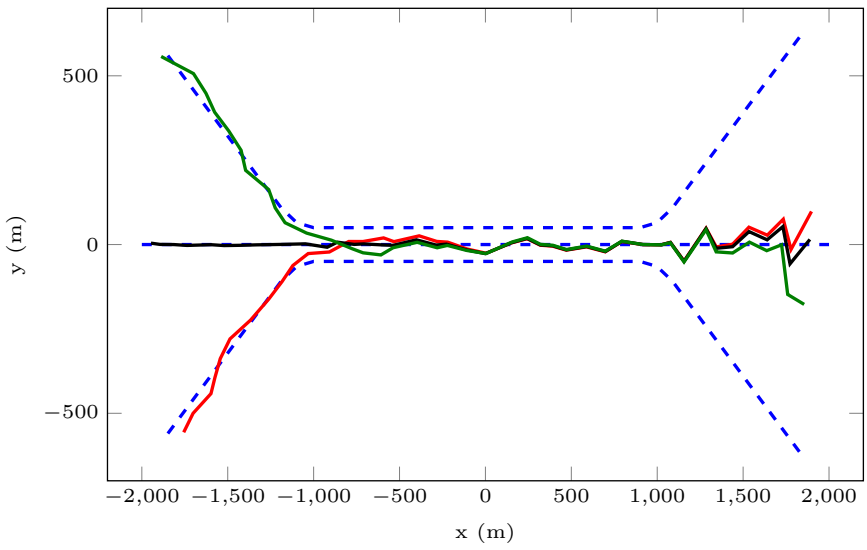


Figure 3.1: An example of track coalescence in the conventional JPDAF (dashed lines: true trajectories; solid lines: tracks).

same computational complexity as the conventional JPDAF, but through an ingenious modification, it greatly reduces track coalescence. The speed-up ideas used in the algorithms in [41, 37] cannot be used in the JPDAF*. Our new algorithm, the JPDAF†, which we have discussed in [56], combines the speed-up ideas from [50] and the JPDAF*, resulting in a fast coalescence-avoiding approximation.

Section 3.2 describes the model used and the conventional JPDAF algorithm. The "cheap JPDAF", Set JPDAF, and JPDAF* are described briefly in Sections 3.3.1, 3.3.2, and 3.3.3, respectively. Section 3.3.4 details the Roecker algorithm. We introduce our new JPDAF† algorithm in Section 3.3.5. We use the Mean Optimal Subpattern Assignment (MOSPA) statistic described in Section 3.4.1 as

well as the normalized estimation error squared (NEES) to evaluate the performance of the algorithms in the scenarios presented in Section 3.4.2. The scenarios consist of three to nine targets.

3.2 The JPDAF

The JPDAF [8] is an algorithm for tracking multiple targets in clutter, which assumes there is a known number of targets with initialized tracks. Measurements from one target can appear close to neighboring targets, causing persistent interference. While unresolved (merged) measurements, discussed in [21, 12], may arise in a real scenario involving closely spaced targets, we will not include them here. The targets' separations considered here yield resolved measurements. The track coalescence problem can occur in the JPDAF even when the targets are resolved.

The JPDAF assumes that a target can only produce at most one measurement, with a known detection probability. Additionally, a measurement could not have been produced by more than one target. The filter uses the state estimate and covariance for each target to summarize the past. The dynamic and measurement models for target t, which may differ between targets, are given as

$$x_t(k+1) = F_t(k)x_t(k) + v_t(k) \tag{3.1}$$

$$z_t(k) = H_t(k)x_t(k) + w_t(k) \tag{3.2}$$

respectively. At time index k for target t, $x_t(k)$ is the state vector, $z_t(k)$ is the measurement vector[2] , $H_t(k)$ is the measurement matrix, and $F_t(k)$ is the state transition matrix. The measurement noise $w_t(k)$ and the process noise $v_t(k)$ are mutually uncorrelated (and across targets), zero mean, and normally distributed with covariances $R(k)$ and $Q(k)$, respectively. The state estimation is done separately for each target.

The most computationally expensive part of the JPDAF is determining the joint association probabilities between measurements and targets. This can be done by finding all the feasible joint association events. The feasibility of an event requires that one or no target produces each measurement and none is from more than one target. The probability of a joint association event θ given the measurements up to time k is [8]

$$P\{\theta(k)|Z^k\} = \frac{1}{c}\prod_j \{\lambda^{-1} f_{t_j}[z_j(k)]\}^{\tau_j(\theta)} \prod_t (P_D^t)^{\delta_t(\theta)}(1-P_D^t)^{1-\delta_t(\theta)} \tag{3.3}$$

where

$$f_{t_j}[z_j(k)] = \mathcal{N}[z_j(k); \hat{z}_{t_j}(k|k-1), S_{t_j}(k)] \tag{3.4}$$

and $\mathcal{N}[x; \mu, P]$ denotes the Gaussian pdf with argument x, mean μ, and variance P. We use Z^k to denote the set of measurements up to and including time index k and c is a normalizing constant. In event θ, t_j is the index of the target that measurement j is associated with. At time index k, λ is the known spatial density of the (Poisson distributed) clutter, $z_j(k)$ is the jth measurement, and P_D^t is the

[2]The tracker does not know which measurement corresponds to which target. This is made clear by the double indexing used in the sequel.

probability of detection of target t; $\tau_j(\theta) = 1$ if measurement j is associated with any target in the event θ, and $\delta_t(\theta) = 1$ if target t is associated with any measurement (they are equal to zero otherwise); $\hat{z}_{t_j}(k|k-1)$ is the predicted measurement for target t_j given as

$$\hat{z}_{t_j}(k|k-1) = H_{t_j}(k)\hat{x}_{t_j}(k|k-1) \tag{3.5}$$

with the predicted state

$$\hat{x}_{t_j}(k|k-1) = F_{t_j}(k-1)\hat{x}_{t_j}(k-1|k-1) \tag{3.6}$$

The associated innovation covariance, $S_{t_j}(k)$, is

$$S_{t_j}(k) = H_{t_j}(k)P_{t_j}(k|k-1)H_{t_j}(k)' + R(k) \tag{3.7}$$

where $P_{t_j}(k|k-1)$ is the covariance of the predicted state from the covariance at the previous time, $P_{t_j}(k-1|k-1)$, and is given by

$$P_{t_j}(k|k-1) = F_{t_j}(k-1)P_{t_j}(k-1|k-1)F_{t_j}(k-1)' + Q(k-1) \tag{3.8}$$

The (marginal) probability β_{jt} that measurement j is associated with target t is then given by summing the probabilities of the joint events where this target to measurement association occurs. This summation can be written as

$$\beta_{jt}(k) = \sum_{\theta} P\{\theta(k)|Z^k\}\hat{\omega}_{jt}(\theta, k) \tag{3.9}$$

where $\hat{\omega}_{jt}(\theta, k) = 1$ if measurement j is associated with target t in event $\theta(k)$ and equal to zero otherwise.

The state estimation for target t is carried out as follows.

$$\hat{x}_t(k|k) = \hat{x}_t(k|k-1) + W_t(k)\nu_t(k) \tag{3.10}$$

$$\nu_t(k) = \sum_j \beta_{jt}\nu_{jt}(k) \tag{3.11}$$

$$\nu_{jt}(k) = z_j(k) - \hat{z}_t(k|k-1) \tag{3.12}$$

$$W_t(k) = P_t(k|k-1)H_t(k)'S_t(k)^{-1} \tag{3.13}$$

where $\hat{x}_t(k|k)$ is the state estimate of target t at time k, $\nu_{jt}(k)$ is the innovation, $\nu_t(k)$ is the combined innovation, and $W_t(k)$ is the filter gain for target t. The covariance update is

$$P_t(k|k) = \beta_{0t}(k)P_t(k|k-1) + [1-\beta_{0t}(k)]P_t^c(k|k) + \tilde{P}_t(k) \tag{3.14}$$

$$P_t^c(k|k) = P_t(k|k-1) - W_t(k)S_t(k)W_t(k)' \tag{3.15}$$

$$\tilde{P}_t(k) = W_t(k)\left[\sum_j \beta_{jt}(k)\nu_{jt}(k)\nu_{jt}(k)' - \nu_t(k)\nu_t(k)'\right]W_t(k)' \tag{3.16}$$

where $P_t(k|k)$ is the covariance associated with the updated state, $P_t^c(k|k)$ is the covariance of the state that would be obtained in the absence of measurement origin uncertainty, $\tilde{P}_t(k)$ is the spread of the innovations, and $\beta_{0t}(k)$ is the probability that target t is not associated with any measurement at time index k.

The computational requirements of the JPDAF can be reduced somewhat by first gating the measurements. The true measurement is assumed to be normally distributed with mean (3.5) and covariance (3.7)

$$p[z(k)|Z^{k-1}] = \mathcal{N}[z(k); \hat{z}(k|k-1), S(k)] \tag{3.17}$$

We can remove from consideration the measurements that are outside a gating threshold γ using the equation

$$[z_j(k) - \hat{z}_t(k|k-1)]' S_t(k)^{-1} [z_j(k) - \hat{z}_t(k|k-1)] \leq \gamma \tag{3.18}$$

How to choose an appropriate value of γ is discussed in [8].

All the approximations of the JPDAF that follow have the same assumptions as described in this section and use gating. The difference between the conventional JPDAF and these algorithms is how β in (3.9) is calculated.

3.3 Approximations to the JPDAF

3.3.1 The "Cheap JPDAF"

For comparison, we include the "cheap JPDAF" of Fitzgerald [26] in our simulations. To approximate the probability of association β_{jt} between a track t and a measurement j, this algorithm employs the *ad hoc* formula

$$\beta_{jt} = \frac{G_{jt}}{C_j + D_t - G_{jt} + B} \tag{3.19}$$

$$C_j = \sum_i G_{ji} \qquad D_t = \sum_r G_{rt} \tag{3.20}$$

with G_{jt} proportional to the Gaussian likelihood function that z_j originated from target t, given as

$$G_{jt} = |2\pi S_t|^{-1/2} \exp\left(-\frac{1}{2} \nu_{jt}' S_t^{-1} \nu_{jt}\right) \tag{3.21}$$

where ν_{jt} is the vector of measurement residuals for measurement j and track t and S_t is the covariance matrix of the residuals. The value of G_{jt} represents how

well the predicted state from track t fits with measurement j. The above formula can be used with a relatively low computational cost and it reduces to the PDA form in the case of only a single track. The term B in the denominator can be set to a nonzero constant when the probability of detection is less than 1 or when the tracks are in the presence of clutter. In the simulations that follow we choose $B = 0$.

3.3.2 The Set JPDAF

We also compare the Set JPDAF [67] in our simulations. This algorithm is designed for scenarios where it is not important to retain the identities of targets over time. One way of managing identity is discussed in [19]. The Set JPDAF is derived using finite set statistics, and has been shown to outperform the conventional JPDAF when utilizing the MOSPA statistic defined in 3.4.1. The reader is referred to [67] and [17] for details of the derivation and implementation of this algorithm, respectively.

3.3.3 The JPDAF*

The JPDAF* [13] is the same as the JPDAF, but with one key modification: for each set of targets that are detected and set of measurements, only the best joint association event is chosen to be used in the calculation of the values of the association probabilities β. The other events that consist of the same sets of targets and measurements, but with a different assignment between targets

and measurements, are discarded. Removing these permuted assignments helps to alleviate track coalescence. Similar pruning techniques have also been applied to hypothesis generation in MHTs [44].

3.3.4 The Roecker Algorithm

This JPDAF version, introduced in [50], uses a certain number of joint association events for probability calculations. Faster and less accurate algorithms, such as the ones we previously evaluated in [55], use a more limited number of events. Instead of calculating all possible joint events like the conventional JPDAF, this algorithm only calculates the highly probable joint events; this is a subset of the set of events the conventional JPDAF would use. These joint events are constructed from the more probable marginal events.

This algorithm can be compared to Murty's m-best 2D assignment algorithm [10]. The Roecker algorithm can be thought of as a simplified version of Murty's algorithm. Both algorithms start from an initial "best" assignment, θ_1; Murty's algorithm starts from the optimal assignment, but the Roecker algorithm can start from a suboptimal assignment. Murty's algorithm partitions the problem into subproblems, and each subproblem can then yield further subproblems. From these, the m-best assignments can be found. The Roecker algorithm produces one set of subproblems based on the initial assignment and finds a number of assignments, not necessarily the m-best (or unique), from these subproblems. The Roecker algorithm produces the set of subproblems by removing sets of marginal

events from the initial problem. These sets of marginal events are taken from the initial assignment. Each combination of marginal events can be removed from the initial problem to produce a subproblem.

Roecker recommends using a greedy algorithm when finding assignments because it results in fewer duplicate events. The algorithm proceeds as follows.

1. Initialization: Find the initial "best" assignment, θ_1; $\ell = 1$
2. For each combination of marginal events, L, in θ_1:
   (a) $\ell \leftarrow \ell + 1$
   (b) Find the "best" assignment, θ_ℓ, such that $L \nsubseteq \theta_\ell$.

The probabilities of the resulting unique joint events θ_ℓ from the above iterations are evaluated. From these, the marginal association event probabilities are calculated and used in the state update. The algorithm does not require that it be run in its entirety. At any point it may be halted and the current set of θ_ℓ used.

3.3.5 The JPDAF†

In this new algorithm, we modify the Roecker algorithm of Section 3.3.4 to incorporate the idea presented in the JPDAF*. We remove joint association events that are permutations of other, more likely, events. The modified Roecker algorithm proceeds as follows:

1. Generate a new joint association event in the same manner as the Roecker algorithm of section 3.3.4.

2. Compare the new event against all previously generated events that contain the same number of assignments. If the new event is a permutation of an old event then discard it. Otherwise keep it to use in the calculation of β.

3. Repeat from step 1 until the algorithm finishes generating events.

The probability of each joint event is found using (3.3), and the marginal association probabilities are found using (3.9). By eliminating permutations of assignments, the JPDAF† lessens the amount of track coalescence and it does not suffer from the large computational complexity that the JPDAF* encounters.

3.4 Simulations

3.4.1 The MOSPA Statistic

To obtain a measure of performance of the various algorithms we employ the Optimal Subpattern Assignment (OSPA) statistic [66]. This metric gives an evaluation of the performance of a tracking algorithm without considering the targets' identities. It allows tracks to switch, but penalizes an algorithm for finding more or fewer tracks than actually exist. While in our simulations the trackers always know exactly how many targets there are, some of the tracks can "veer off" and not represent any target or they might coalesce. The OSPA statistic needs two parameters, c and p, where c is the cutoff parameter which

limits the penalty assigned for false or lost tracks, or for true targets that are not being tracked, and p determines how much of a penalty is assigned for estimates not close to any true target. We use the values $p = 1$ and $c^2 = 20000\,\mathrm{m}^2$ in the simulation that follows.

The OSPA statistic between the set of estimated states $\hat{X}(k|k)$ and the set of true target states $X(k)$ is given by

$$\bar{d}_p^{(c)}(\hat{X}(k|k), X(k)) = \left(\frac{1}{T} \min_{\pi \in \Pi} \sum_{t=1}^{T} d^{(c)}[\hat{x}_t(k|k), x_{\pi(t)}(k)]^p \right)^{1/p} \tag{3.22}$$

with

$$d^{(c)}[\hat{x}_t(k|k), x_{\pi(t)}(k)] = \min \left[c, d[\hat{x}_t(k|k), x_{\pi(t)}(k)] \right] \tag{3.23}$$

where T is the number of targets, $\hat{x}_t(k|k)$ denotes the t^{th} state estimate contained in $\hat{X}(k|k)$,

$d[\hat{x}_t(k|k), x_{\pi(t)}(k)]$ is the distance between $\hat{x}_t(k|k)$ and $x_{\pi(t)}(k)$, and Π is the set of all permutations of the true target states. We use only the position components of the state in the calculation of the OSPA. The OSPA finds the assignment between the true states and the estimated states at each time index k that yields the smallest sum of errors. Over a number of Monte Carlo runs we take the average of these errors to find the mean OSPA (MOSPA). The goal is to have the MOSPA as low as possible.

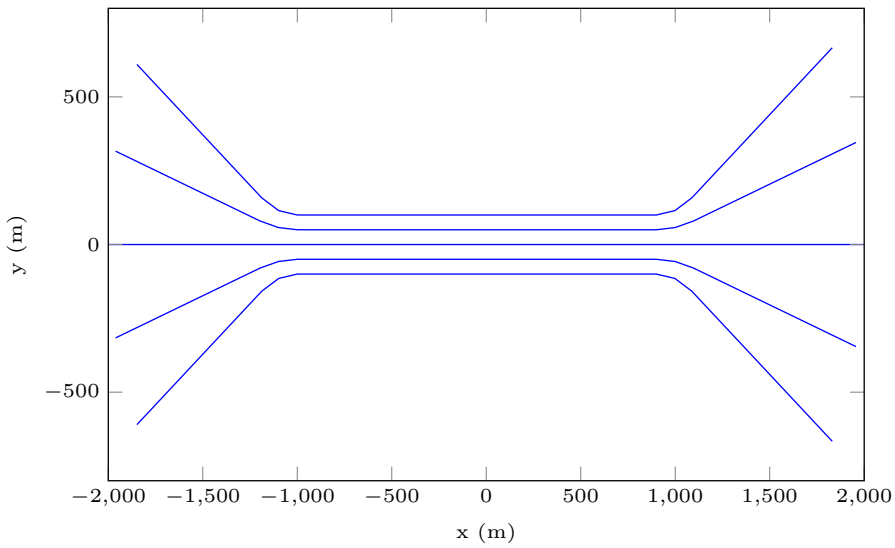


Figure 3.2: The simulated 5 target scenario.

3.4.2 Scenarios

In order to establish a comparison between the five algorithms considered in this paper, we simulate a few simple scenarios similar to those in [17] and [18]. The four scenarios considered all consist of a number of targets (3, 5, 7, and 9) approaching each other, moving parallel to each other at a close distance starting at time index $k = 11$, and then separating at $k = 31$. An example of a scenario consisting of 5 targets is shown in Figure 3.2.

We use the discretized continuous white noise acceleration (CWNA) model in our simulations [7]. The turning rate of targets ranges between $\pm.3\,\text{rad/s}$. In all scenarios the targets have a constant speed of $V = 100\,\text{m/s}$ and are separated by $50\,\text{m}$ during the time they are traveling parallel. Measurements are taken with a sampling interval of $T = 1\,\text{s}$. The probability of detection for all targets is 85%. The measurements are taken in polar coordinates from a sensor located at ($0\,\text{km}$,

-40 km) and converted to Cartesian. The standard deviations are $\sigma_r = 2\,\mathrm{m}$ and $\sigma_\theta = 10^{-3}\,\mathrm{rad}$. Using the state vector $x = [\xi\ \eta\ \dot{\xi}\ \dot{\eta}]'$, the measurement matrix, state transition matrix for the chosen sampling period and the process noise are given by

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.24}$$

$$F = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.25}$$

$$Q = \begin{bmatrix} \frac{1}{3}T^3 & 0 & \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{3}T^3 & 0 & \frac{1}{2}T^2 \\ \frac{1}{2}T^2 & 0 & T & 0 \\ 0 & \frac{1}{2}T^2 & 0 & T \end{bmatrix} \tilde{q} \tag{3.26}$$

We choose $\tilde{q} = 300\,\mathrm{m}^2/\mathrm{s}^3$ for all scenarios. The chosen value for $\sqrt{\tilde{q}}$ is based on the maximum change in velocity in a sampling period ($\sqrt{\tilde{q}T} \approx 17\,\mathrm{m/s}$) in the given scenarios. The simulated tracks did not have added process noise; $\sqrt{\tilde{q}}$ is used only to account for the (unknown to the tracker) maneuvers. The number of clutter points at each time index k is Poisson distributed with mean $\lambda = 10^{-6}/\mathrm{m}^2$. These points are generated uniformly in a rectangular region whose borders are at least 400 m from the closest point in any track. The gate threshold is chosen

to be $\gamma = 16.22$. The average number of clutter points in a validation gate was close to 1. Track initialization was done using an Information Filter with the first two correctly associated measurements from each target [7].

From the MOSPA plots in Figures 3.3-3.6, it is clear that the JPDAF† significantly outperforms the Roecker algorithm of Section 3.3.4, especially during the time the targets begin to split apart and track coalescence is the most prevalent. The JPDAF† has a small, roughly 10%, increase in error when compared to the JPDAF*. The "cheap JPDAF" does not perform well in any of these scenarios.

Table 3.1 shows the run times for each algorithm in each of the four simulated scenarios. While the JPDAF† runs in a similar time to the JPDAF* and the Roecker algorithm for 3 targets, it runs significantly faster than the JPDAF* in the 5, 7, and 9 target scenarios. In the 7 target scenario the JPDAF† ran approximately 65 times faster than the JPDAF*, and 1000 times faster in the 9 target scenario, while only showing a small increase in the MOSPA. The "cheap JPDAF" ran in the least amount of time, but its poor performance makes it undesirable in these scenarios.

We evaluated the average normalized estimation error squared (NEES), $\bar{\epsilon}(k)$, over N Monte Carlo runs at each time k to measure the consistency of the algorithms using the equation

$$\bar{\epsilon}(k) = \frac{1}{N} \sum_{i=1}^{N} \epsilon^i(k) \tag{3.27}$$

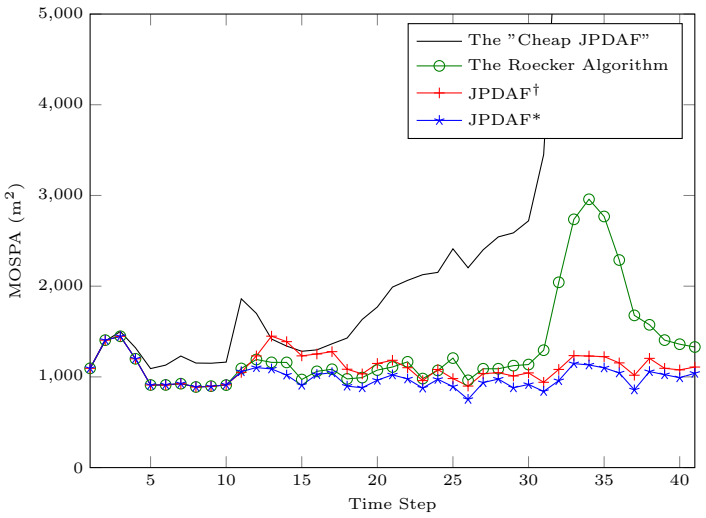


Figure 3.3: A comparison of the MOSPA performance of the "cheap JPDAF," the Roecker algorithm, the JPDAF†, and the JPDAF* from 100 Monte Carlo runs in the 3 target simulated scenario.

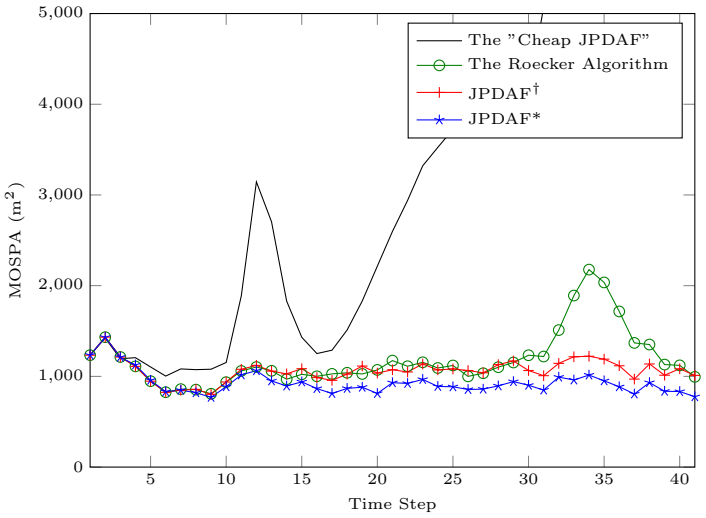


Figure 3.4: A comparison of the MOSPA performance of the "cheap JPDAF," the Roecker algorithm, the JPDAF†, and the JPDAF* from 100 Monte Carlo runs in the 5 target simulated scenario.

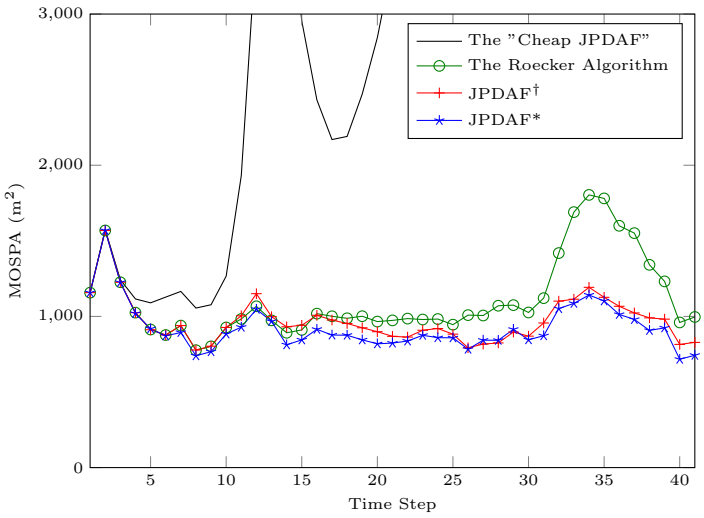


Figure 3.5: A comparison of the MOSPA performance of the "cheap JPDAF," the Roecker algorithm, the JPDAF*, and the JPDAF† from 100 Monte Carlo runs in the 7 target simulated scenario.

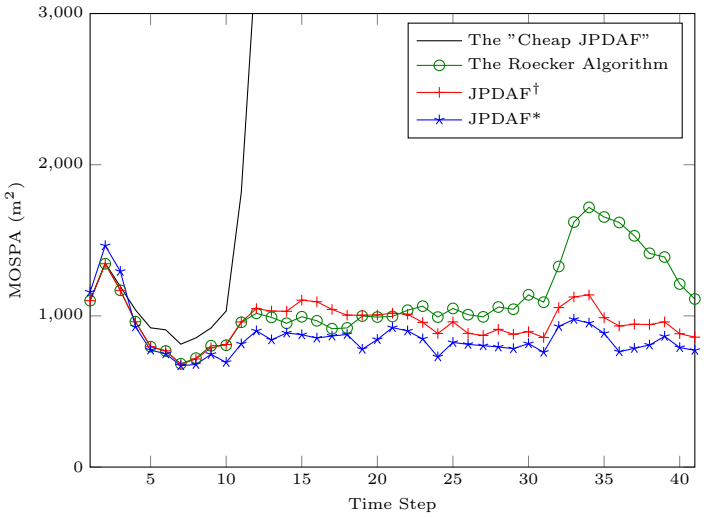


Figure 3.6: A comparison of the MOSPA performance of the "cheap JPDAF," the Roecker algorithm, and the JPDAF† from 100 Monte Carlo runs in the 9 target simulated scenario. The JPDAF* is also compared here from 51 runs.

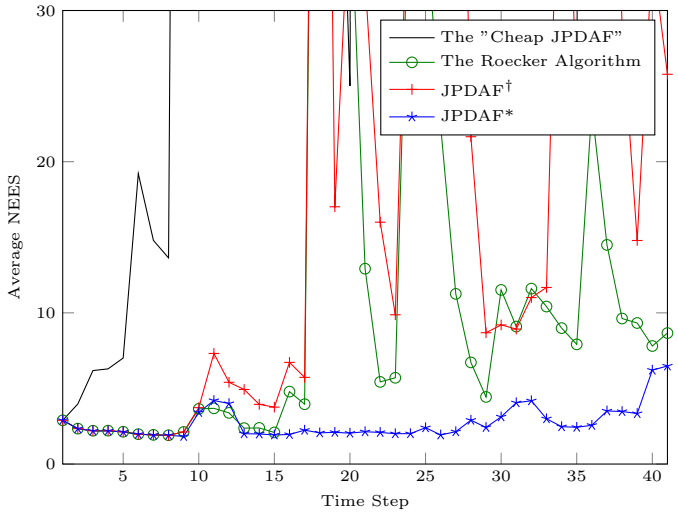


Figure 3.7: A comparison of the NEES of the "cheap JPDAF," the Roecker algorithm, the JPDAF†, and the JPDAF* from 100 Monte Carlo runs in the 3 target simulated scenario.

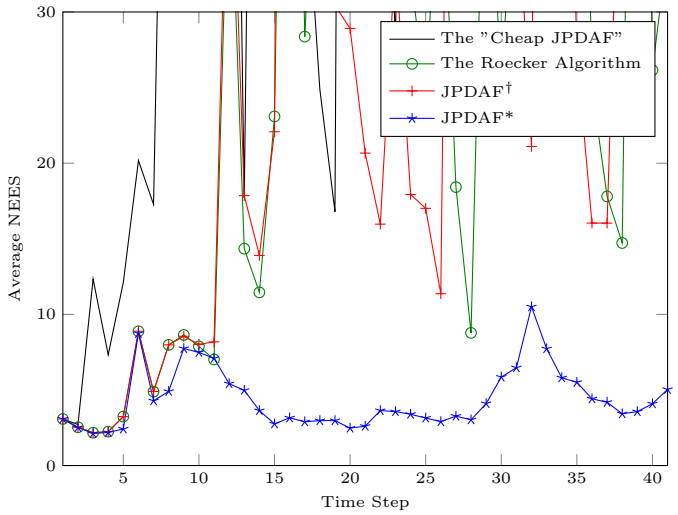


Figure 3.8: A comparison of the NEES of the "cheap JPDAF," the Roecker algorithm, the JPDAF†, and the JPDAF* from 100 Monte Carlo runs in the 5 target simulated scenario.

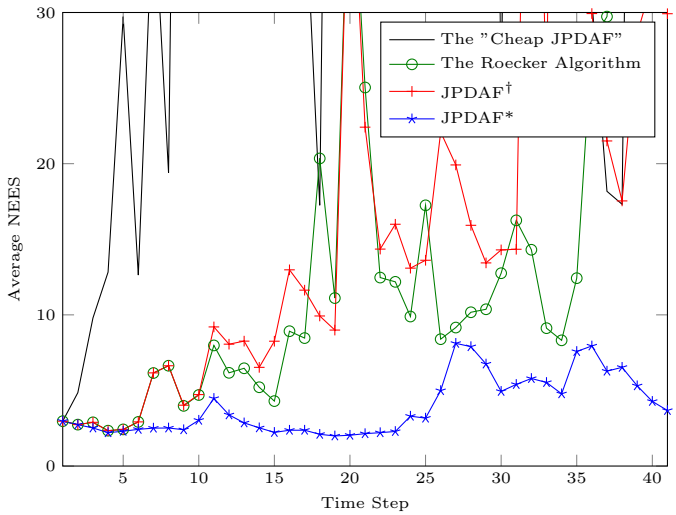


Figure 3.9: A comparison of the NEES of the "cheap JPDAF," the Roecker algorithm, the JPDAF*, and the JPDAF† from 100 Monte Carlo runs in the 7 target simulated scenario.

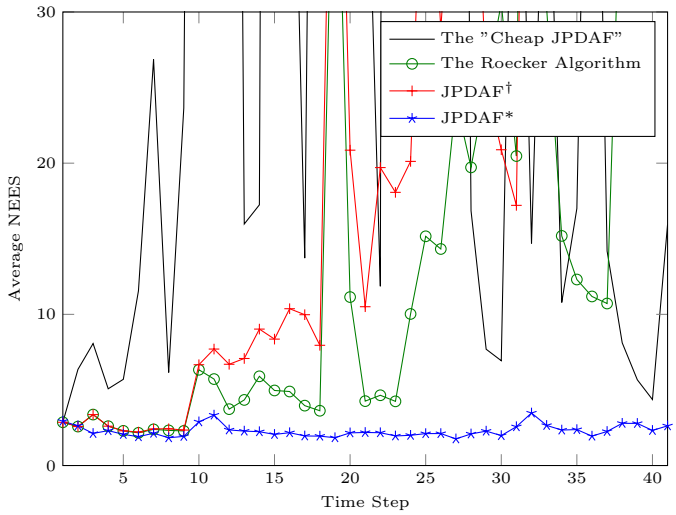


Figure 3.10: A comparison of the NEES of the "cheap JPDAF," the Roecker algorithm, and the JPDAF† from 100 Monte Carlo runs in the 9 target simulated scenario. The JPDAF* is also compared here from 51 runs.

	3 Targets	5 Targets	7 Targets	9 Targets
The "Cheap JPDAF"	14	27	43	63
JPDAF†	14	37	107	387
The Roecker Algorithm	17	62	331	2557
JPDAF*	20	231	6544	198110 (for only 51 runs)
Set JPDAF	4078	-	-	-

Table 3.1: Total running time, in MATLAB, for 100 Monte Carlo runs in s on an Intel Core 2 Duo E8600 3.33GHz. The time taken for the evaluation of the MOSPA is not part of the running time. In our configuration MATLAB did not have enough memory to run the Set JPDAF in the 5, 7, or 9 target scenarios. The JPDAF* was run for about 1 week in the 9 target scenario before it was stopped. It was able to complete 51 Monte Carlo runs in this time and appeared to be stuck in run 52.

where

$$\epsilon^i(k) = \frac{1}{T} \min_{\pi \in \Pi} \sum_{t=1}^{T} (x_{\pi(t)}(k) - \hat{x}_t(k|k))' P_t(k|k)^{-1} (x_{\pi(t)}(k) - \hat{x}_t(k|k)) \tag{3.28}$$

Figures 3.7-3.10 show that all the algorithms suffer from some consistency problems, however the JPDAF* does perform the best. It is not surprising that the JPDAF† is less consistent than the JPDAF* as it is a modification of the Roecker algorithm which is very inconsistent in these scenarios.

3.5 Conclusions

The JPDAF† offers a new approximation to the JPDAF that is suitable for a large number of closely-spaced targets where the JPDAF* becomes prohibitively expensive. The JPDAF† incorporates the approximation technique of the Roecker algorithm and the coalescence-avoiding technique used in the JPDAF*. The simulation results demonstrate that combining the approaches of these two algorithms

provides a significant decrease in computational complexity while approaching the accuracy of the JPDAF* in these scenarios.

Chapter 4

Particle Filter Tracking for the Banana and Contact Lens Problems

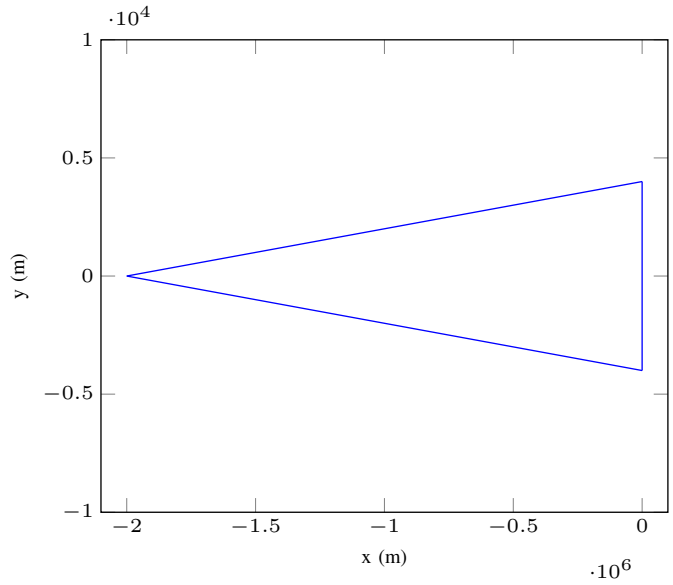
4.1 Introduction

The long range tracking scenario with an active sensor presents an interesting challenge known as the banana (or crescent) problem in two dimensions, or contact lens problem in three dimensions. Its name refers to the thin and curved shape of the measurement uncertainty region when viewed in Cartesian coordinates; a notional example of the banana shape is shown in Figure 4.1. This problem arises when the measurements are very accurate in range and inaccurate in cross range. As the range to a target becomes very large, even decent angular standard deviations can translate to severe inaccuracies in the cross range. This can result in degraded track accuracy and inconsistency (actual errors not commensurate with the filter-calculated covariances [7]) in various filters.

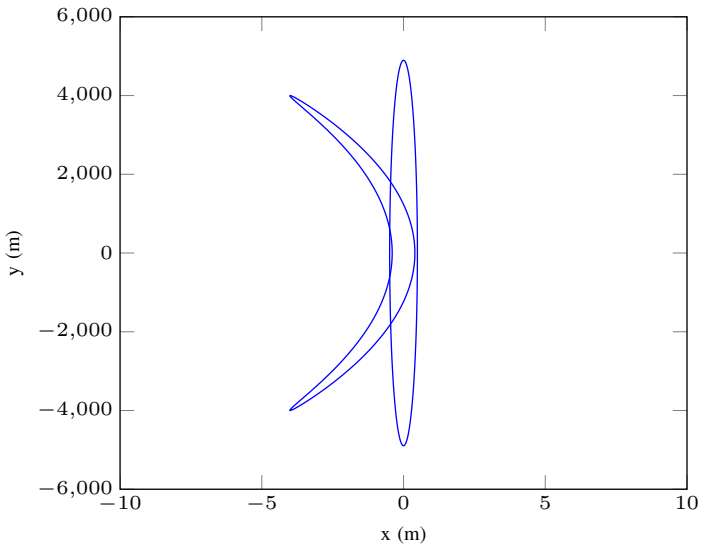
Two state of the art algorithms, the Measurement Covariance Adaptive Extended Kalman Filter

(MCAEKF) [70] and the Consistency based Gaussian Mixture Filter (CbGMF) [69], are able to produce consistent results, but do so by sacrificing accuracy in the range direction during early stages of filtering. The former is a modification of the standard Extended Kalman Filter (EKF), and the latter uses a consistent track splitting idea, which amounts to a Gaussian mixture approach similar to the techniques used in [29, 30, 68]. However, it is desirable to have a filter that does not make this sacrifice in range accuracy. Another approach is the log-homotopy particle flow filter and its versions [22], which is an emerging technology that may, when it is fully mature, provide alternative solutions to this and many difficult nonlinear filtering problems.

One possible alternative approach to this problem is the particle filter. Previous work, such as [23] and [28], has applied a particle filter to real world small-target tracking problems. In [23], a particle filter was shown to have slightly better performance than the EKF, but the particle filter occasionally diverged. In [28] many real world applications in which a particle filter may be useful were presented, such as underwater and surface positioning. In [38], a method is presented to design a proposal density based on the scenario considered. However, the scenarios explored in [38], which are similar to the banana problem, have a process noise approximately five orders of magnitude larger than the scenarios considered in this paper; the very low process noise in our scenarios precludes



(a) The notional sensor beam.



(b) The banana shaped region resulting from the sensor in (a) plotted exactly and with the linearized covariance ellipse. This shape can only be seen when the abscissa scale is magnified.

Figure 4.1: An example of a banana shaped measurement uncertainty region in two dimensions. The sensor here is located at (–2000 km, 0 km), and the range and angular accuracies are, respectively, 0.2 m and 10^{-3} rad.

the use of a proposal density as in [38]. We implemented the algorithm in [38] and discuss its results in Section 4.6. The assumption of zero process noise, as in the method of [36], is also not applicable. The process noise, albeit small, does allow for a significant change in the target's range relative to the high range accuracy of the scenario. This effect of the process noise needs to be accounted for. Additionally, in [36], a time-varying proposal density must be defined with decreasing covariance to accommodate the decreasing covariance of the posterior; our method uses the time-varying covariance of the particles which makes the kernel self-adaptive. We show that, for the real-world problem considered, the regularized version of the particle filter developed in this paper is the only one which is consistent and its range estimate accuracy is below the single range measurement accuracy; this latter feature is particularly difficult to achieve for a sensor that has very high range measurement accuracy.

A particle filter represents the posterior distribution of the state by a set of weighted samples, or particles (the pdf is approximated by a sum of delta functions, i.e., point masses). The estimate of the state and its associated covariance are easily calculated from this set of samples. This estimate approaches (in theory) the optimal one as the number of samples is increased. However, as we describe in Section 4.2, the basic particle filter encounters two significant problems in this scenario: particle degeneracy and sample impoverishment. Many particle filter versions exist that attempt to alleviate these problems, such as auxiliary particle filter [3], the regularized particle filter (RPF) [3, 40], the resample-move

particle filter [24] and the Gaussian Mixture Sigma-Point Particle Filter (GM-SPPF) [74]. Some of these versions were examined in [57]. We also evaluated a particle filter with proposal density linked to the EKF [73], but it was unable to run in the scenarios explored in this paper due to the aforementioned problems (degeneracy and impoverishment) which occur with very low process noise. We will only focus on the RPF here as it has superior performance when compared to the other particle filter versions in the scenarios considered. The regularization amounts to replacing each particle (point mass) by a pdf with finite support. This is accomplished using a generalized version of the Epanechnikov kernel, suitable for state vectors whose components have different physical dimensions (since we deal with a real-world problem).

The RPF is described in Section 4.3. The necessary modification to the Epanechnikov kernel is also discussed in that Section. Sections 4.4.1 and 4.4.2 briefly describe two covariance-adaptive algorithms [69, 70] which have had success in these scenarios. All of the algorithms are then simulated in Section 4.5 and compared in Section 4.6. The consistency of the filters is discussed in Section 4.6.2.

4.2 Particle Filter

A particle filter approximates a Bayesian filter's posterior density by a set of weighted samples (point masses). The state estimate and its associated covariance can then be calculated using this set of weighted samples called particles. In a

simple version (bootstrap, or sequential importance resampling) of a particle filter [3, 8] this is accomplished by first sampling N particles from the prior density

$$x_i(k) \sim p[x(k)|x_i(k-1)] \tag{4.1}$$

where $x_i(k)$ is the ith particle at time k. The weights of the particles are are determined by finding the likelihood of each

$$\tilde{w}_i(k) = p[z(k)|x_i(k)] \tag{4.2}$$

where $z(k)$ is the measurement at time k. The weights are then normalized

$$w_i(k) = \frac{\tilde{w}_i(k)}{\sum_{i=1}^{N} \tilde{w}_i(k)} \tag{4.3}$$

It is necessary to perform resampling of the particles to avoid the problem of particle degeneracy. This degeneracy occurs when the weights of many of the particles approach zero. Particles with nearly zero weight will not significantly change the approximation of the posterior density $p(x(k)|Z^k)$, where Z^k is used to denote all measurements up to and including time k and $x(k)$ is the true target state. These low weight particles contribute little, so little is lost when the resampling step effectively trades off their removal for duplication of high weight particles. This can be accomplished by sampling with replacement from the current set of particles. The probability of sampling each particle is given by its weight. The calculation of the state estimate should be done before any resampling, as resampling can only make the estimate worse.

It is also possible to sample from a proposal density, $q[x(k)|x_i(k-1), z(k)]$, instead of the prior density. Sampling in this manner requires that we use the

importance weight

$$\tilde{w}_i(k) = \frac{p[z(k)|x_i(k)]p[x_i(k)|x_i(k-1)]}{q[x_i(k)|x_i(k-1), z(k)]} \tag{4.4}$$

After initializing a set of particles, e.g., through the use of two point differencing, the particle filter algorithm would then proceed as follows during each iteration:

1. Sample N particles from the proposal density

$$\tilde{x}_i(k) \sim q[x(k)|x_i(k-1), z(k)] \tag{4.5}$$

2. Calculate the weights of the particles

$$\tilde{w}_i(k) = \frac{p[z(k)|\tilde{x}_i(k)]p[\tilde{x}_i(k)|x_i(k-1)]}{q[\tilde{x}_i(k)|x_i(k-1), z(k)]} \tag{4.6}$$

3. Normalize the weights

$$w_i(k) = \frac{\tilde{w}_i(k)}{\sum_{i=1}^{N} \tilde{w}_i(k)} \tag{4.7}$$

4. Resample N particles, $x_i(k)$, from $\{\tilde{x}_i(k)\}_{i=1}^{N}$ using $\{w_i(k)\}_{i=1}^{N}$ as the pmf.

The state estimate, $\hat{x}(k|k)$, and its associated covariance, $P(k|k)$, can be found by taking a weighted sum of the particles using

$$\hat{x}(k|k) = \sum_{i=1}^{N} w_i(k)x_i(k) \tag{4.8}$$

$$P(k|k) = \sum_{i=1}^{N} w_i(k)[x_i(k) - \hat{x}(k|k)][x_i(k) - \hat{x}(k|k)]' \tag{4.9}$$

In our particular scenario, which we detail below, we have a process noise that – compared to the measurement uncertainty – is very small and limits the choice

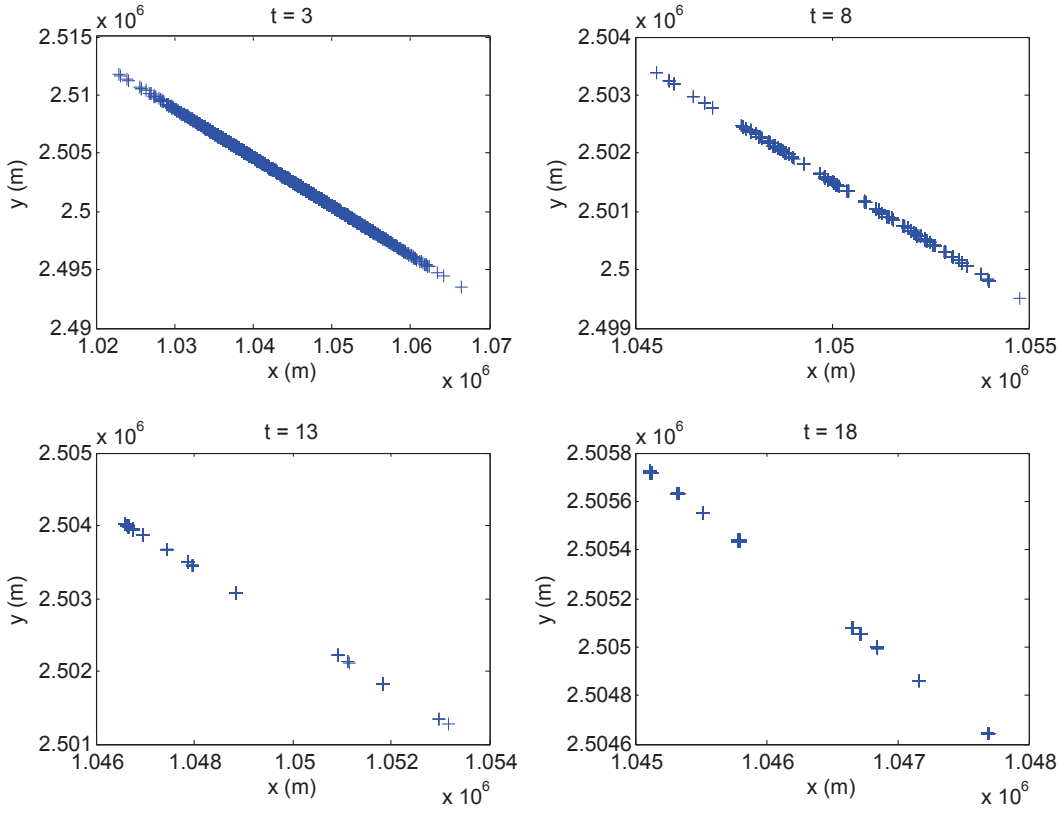


Figure 4.2: A particle filter experiencing sample impoverishment in a long range tracking scenario. Each plot shows the current set of particles at a specific time (3 s, 8 s, 13 s, and 18 s). As time proceeds, the particles become increasingly clustered.

of proposal density. If a proposal density is chosen that places particles outside of the small region allowed by the prior (prediction) density, $p[x_i(k)|x_i(k-1)]$, the calculated weights (see (4.6)) will be vanishingly small. This problem can be avoided by simply choosing to use the prior density as the proposal. However, the small process noise creates another problem, namely, sample impoverishment, or a loss of diversity among particles. The resampling step creates identical copies of particles which will then separate during the sampling step of the next iteration, but with a very small process noise the separation achieved can be so small that it is negligible when compared to the size of the space the particles should fill. An example of this is shown in Figure 4.2. There are 10000 particles shown in each plot, and the particles eventually collapse into small and distinct groups during resampling as the filter iterates – the small process noise prevents the particles from diffusing to cover the region they ought. That is, ultimately one can be too aggressive with one's importance density and end up losing a high proportion of one's particles; or one can be too timid and end up with an unsatisfactorily freckled particle countenance.

Figure 4.3 shows a simplified depiction of what occurs in a very low process noise scenario. Two particles at time k are used to create two prior (prediction) densities at time $k+1$ from which two new particles are sampled. The new lower particle happens to fall far away from the measurement so during resampling it is removed, and the upper particle is duplicated. Two new particles are then sampled at time $k+2$ from the only surviving prediction (prior) density. As

depicted, the covariance ellipse associated with this prior density is very small relative to the ellipse associated with the measurement. This will force the two particles to remain very close to each other in subsequent iterations of the filter, and result in the clustering seen in Figure 4.2.

The low process noise problem is the dual of the (better-known) low measurement uncertainty problem, as shown in Figure 4.4, that particle filters also might encounter. The prior densities have covariances much larger than the covariance associated with the measurement. Now particles can easily be sampled far enough away from the measurement to be assigned very small weights. If all of the particles move away from the area of high likelihood, as shown in this Figure at time $k+2$, then the filter will "die of impoverishment".

Since we are restricted to using the prior density (or a proposal of similar spread) due to the very low process noise, we must look to a different variation of the particle filter to solve this problem.

4.3 Regularized Particle Filter

The regularized particle filter (RPF) [3, 40] attempts to avoid the problem of sample impoverishment by changing how the resampling step is done. The resampling step in Section 4.2 samples particles from a discrete distribution which creates exact duplicate particles, leading to the impoverishment problem. If samples are instead taken from a continuous approximation of the posterior density,

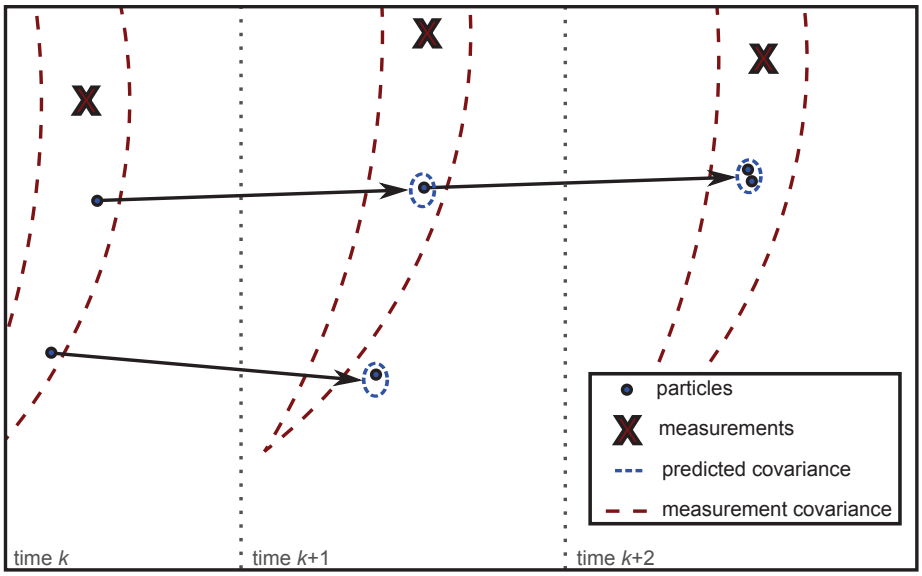


Figure 4.3: The effect of very low process noise on a basic particle filter.

no such duplication of particles will occur and this problem should be prevented. This is accomplished as follows.

We first define a kernel $K(u)$ of a vector u (with physically dimensionless components) such that

$$K(u) \geq 0 \qquad u \in \mathbb{R}^{n_x} \tag{4.10a}$$

$$\int K(u)du = 1 \tag{4.10b}$$

$$\int uK(u)du = 0 \tag{4.10c}$$

$$\int \|u\|_2^2 K(u)du < \infty \tag{4.10d}$$

A rescaled kernel is then defined as

$$K_h(u) = \frac{1}{h^{n_x}} K\left(\frac{u}{h}\right) \tag{4.11}$$

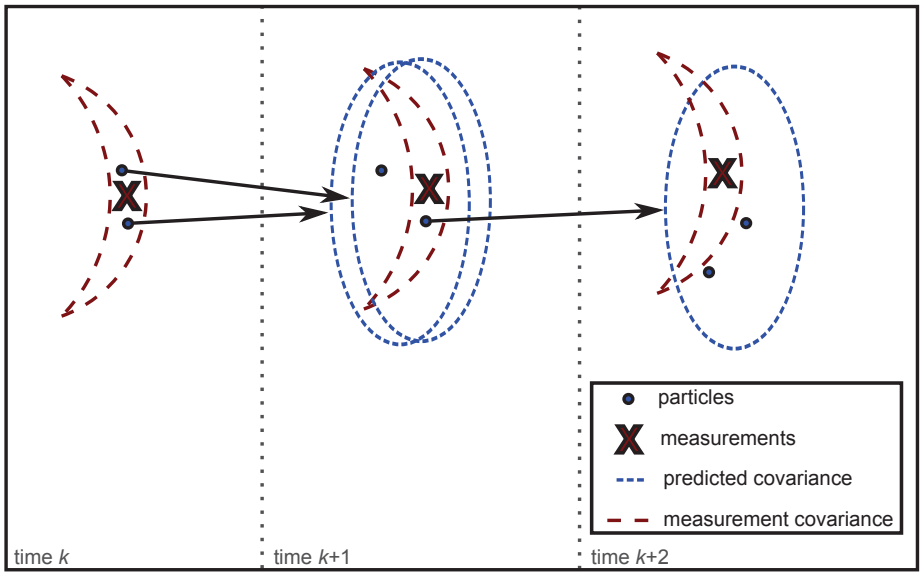


Figure 4.4: The effect of very low measurement uncertainty on a basic particle filter.

where $h > 0$ is the kernel bandwidth and n_x is the dimension of x.

When all the particles have equal weights (which is accomplished simply by resampling as in Section 4.2), and the vector u has physically dimensionless components (i.e., its norm is also physically dimensionless and can be compared to a scalar threshold, which is unity in this case) the optimal kernel in the mean integrated square error sense is the Epanechnikov kernel [3, 40]

$$K_{\text{opt}}(u) = \begin{cases} \frac{n_x+2}{2c_{n_x}}(1 - \|u\|_2^2) & \text{for } \|u\|_2 < 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

where c_{n_x} is the volume of the unit hypersphere in $\mathbb{R}^{n_x}$.

We use a kernel suitable for use with the state vector x (whose components — position, velocity, etc. — have different physical dimensions) by using a norm with respect to its inverse covariance. This is a multidimensional generalization

of what was done in [31] for the scalar case. The resulting norm is physically dimensionless and it can be compared to a number as required in (4.12). This results in the generalized Epanechnikov kernel given by

$$K_h(x) = \begin{cases} c(1 - \frac{1}{h^2}\|x\|^2_{S^{-1}}) & \text{for } \frac{1}{h}\|x\|_{S^{-1}} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.13}$$

where c is a normalizing constant.[1] The continuous approximation of the posterior density is then

$$p(x(k)|Z^k) \approx \sum_{i=1}^{N} w_i(k) K_h[x(k) - x_i(k)] \tag{4.14}$$

We can then augment the standard resampling used in Section 4.2 to create following procedure to sample from equation (4.14) using $K_h(x)$ from equation (4.13):

1. Find the sample covariance matrix of the set of weighted particles, $S(k)$.

2. Find a square root matrix $D(k)$ such that $D(k)D(k)' = S(k)$.

3. Resample N particles, $x_i(k)$, from $\{\tilde{x}_i(k)\}_{i=1}^N$ using $\{w_i(k)\}_{i=1}^N$ as the pmf.

4. For each particle, sample ϵ_i from $K_{\text{opt}}(u)$, the Epanechnikov kernel, and do

$$x_i(k) \leftarrow x_i(k) + hD(k)\epsilon_i \tag{4.15}$$

[1]Note that equation (3.50) in [49], the Epanechnikov kernel definition, requires $\mathbf{x}$ to be a physically dimensionless vector (it is not possible to take the norm of a state vector whose components are of different physical dimension, nor is it possible to compare the norm of a vector with components that have any physical dimension to unity). Therefore a modification is required in (4.13) when $\mathbf{x}$ is a state vector with physical dimensions (as it would be in a real world problem).

In the above, ϵ_i is a vector with physically dimensionless components and, when multiplied by $D(k)$, its components will have the same physical dimensions as the corresponding components of x. The kernel depends on $S(k)$, which varies with time and since the filtered state covariance is time-varying. Figure 4.6 shows a simplified depiction of the resampling process in the RPF. Each particle (point mass) is replaced by a continuous pdf with finite support. This support is in a shape similar to that of the sample covariance and its size is scaled by the bandwidth, h. The choice of h is discussed in [76].

In the RPF in our 2-D simulations we choose $h = 0.5$ ($h = 2$ for 3-D); this value, which amounts to "half sigma" in (4.15), is large enough to avoid sample impoverishment without being so large that particles spread far away from the area of high likelihood. The covariance of the particles increases by approximately 10% after resampling with this bandwidth[2] . We do not use any proposal density for the reasons described in Section 4.2. The likelihoods in equation (4.2) are calculated in measurement (polar or r-u-v) coordinates, as is the position estimate. The position estimate is converted back to Cartesian coordinates when calculating the position error. An overview of the filtering algorithm with the conversions between measurement and Cartesian coordinates highlighted is shown in Figure 4.5.

[2]Note that a 10% increase is very small; even a $\$10^8$ radar cannot specify an accuracy within 10-15%. A reference to this and to all things can be found in [20].

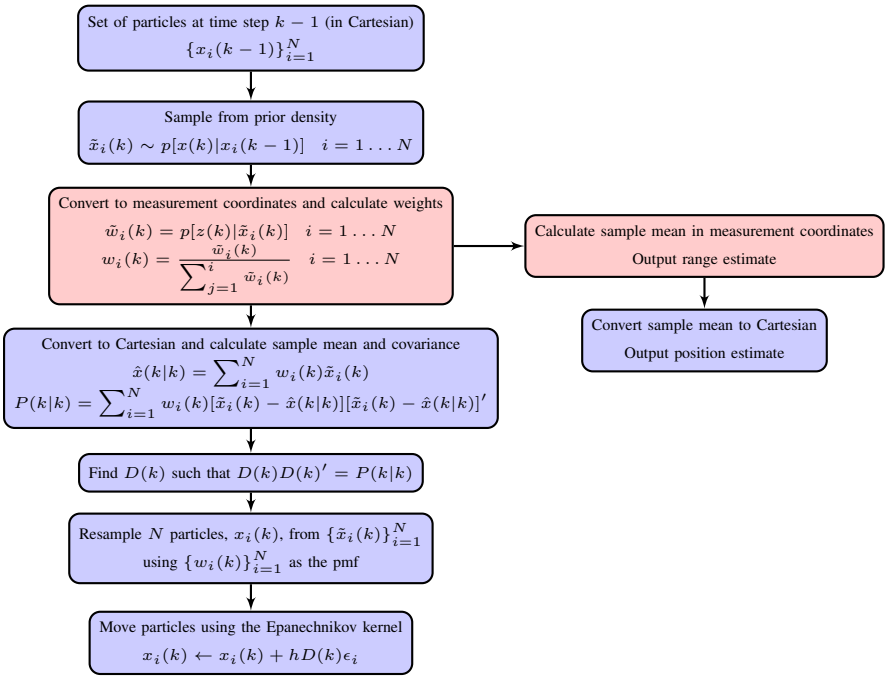


Figure 4.5: A flowchart of the RPF algorithm. Blue blocks indicate particles in Cartesian, and red blocks indicate particles in measurement coordinates.

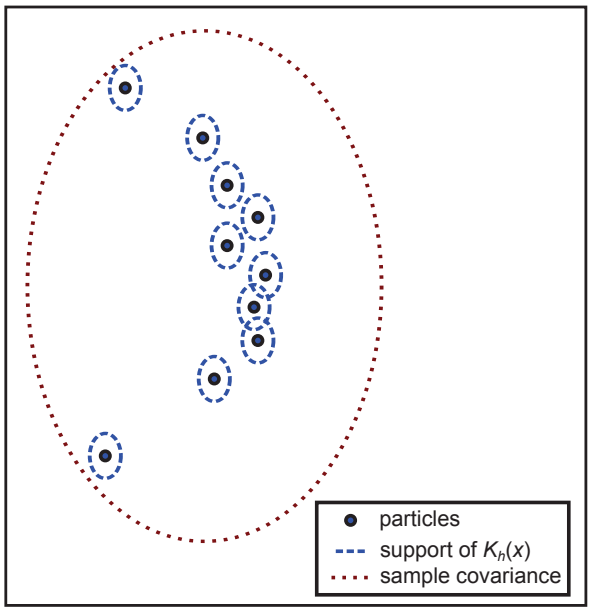


Figure 4.6: The resampling process in the regularized particle filter.

4.4 Other Algorithms

We investigated the application of several other algorithms to the long range tracking problem. We simulate the EKF and unscented Kalman filter (UKF) and show their results to be undesirable in these scenarios in Section 4.6.

4.4.1 Measurement Covariance Adaptive Extended Kalman Filter

We also include the MCAEKF [70] in our simulations for comparison because it performs well in this type of long range scenario. The MCAEKF is a modification of a standard EKF that sacrifices accuracy in the range direction during the early stages of filtering to guarantee overall filter consistency. When track accuracy is low during the early stages of filtering, the MCAEKF uses a minimum value for the standard deviation of the range measurement (larger, however, than the very small actual value) to retain consistency. Once track accuracy is sufficiently high, no modification is required and the MCAEFK is identical to the EKF.

As shown earlier, the measurement uncertainty region takes the shape of a banana. The predicted target position uncertainty in the EKF is represented by a Gaussian ellipse. The intersection of these regions is the significant region of the measurement uncertainty. Artificially increasing the standard deviation of the measurement in the range direction allows this significant region to be well covered by a first order approximation. If this region is not well covered the EKF

will become inconsistent. For details on the implementation of the MCAEKF, see [70].

4.4.2 Consistency Based Gaussian Mixture Filter

The CbGMF [69] is designed to solve the problems caused when the concentration of the posterior distribution of the target states is curved like a banana. Since this occurs in the long range tracking scenarios we are examining, we have included it for comparison. The filter adaptively divides the target track into a dynamic set of subtracks, which amounts to a Gaussian mixture model (GMM), and this guarantees filtering consistency. The CbGMF includes the same consistent filtering rule used in the MCAEKF of [70]. It also includes techniques to limit the complexity of the algorithm.

The consistent track splitting component of the algorithm was developed to work with the consistent filtering rule. It divides a single Gaussian distribution into a set of Gaussians with equal covariances, and as long as the subtracks are accurate enough to satisfy the consistent filtering rule and there is at least one subtrack that covers the true target state, the consistency of the whole set of subtracks will be guaranteed [69].

To control complexity, the subtracks that are far away from the true target state are removed, and an upper limit is set for the total number of subtracks. When an iteration occurs where the number of subtracks is going to violate this limit, all subtracks in the previous time step are recombined through a weighted

sum into a single Gaussian. Then the algorithm proceeds as normal, splitting the track according to the consistent track splitting algorithm. The details of the implementation of this algorithm are not discussed here; they can be found in [69].

4.4.3 Other Sequential Monte-Carlo Methods

Various Sequential Monte-Carlo (SMC) methods can also be applied to the very long range tracking problem. Indeed, in [38], a method is presented to design a proposal density based on the scenario considered. However, there are some considerable differences between the scenarios explored in [38] and those in this paper. The scenarios from [38], while similar to the banana or contact lens problem, have a process noise approximately five orders of magnitude larger. It is the very low process noise in our scenarios that precludes the use of a proposal density as in [38].

To contrast this, there are other SMC methods, such as [36], where there is an assumption of zero process noise. This too is also not applicable to the problem considered here. The process noise present in the banana problem, albeit small, does allow for a significant change in the target's range relative to the high range accuracy of the scenario (25 times the range standard deviation over the course of the simulation of the scenarios discussed later in this paper). This effect of the process noise needs to be accounted for.

In the wider family of SMC methods there may exist solutions appropriate for a small, but non-zero, process noise, which would correctly model the long range radar scenario considered here. To the knowledge of the authors, no specific work has been done exploiting SMC methods to satisfactorily address the banana or contact lens scenarios. Future work may include extending existing SMC methods to deal with the specific challenges of the long range radar problem.

4.5 Scenarios for Simulations

We evaluated the performance of the RPF in scenarios in two and three dimensions, and compared it to the MCAEKF, the CbGMF, as well as the EKF and UKF. We also compare the performance to the BCRLB[3] (pCRLB) [72].

4.5.1 2-D Scenarios

The scenarios we simulate are real-world problems for point targets. The scenarios consist of a single target moving in a straight line trajectory. We use the discretized continuous white noise acceleration (CWNA) model [7] and order the state vector components as

$$x = [\xi \ \dot{\xi} \ \eta \ \dot{\eta}]' \tag{4.16}$$

The dynamic model is given by

$$x(k+1) = Fx(k) + v(k) \tag{4.17}$$

[3]Bayesian CRLB (BCRLB) according to [75] which was called posterior CRLB (pCRLB) in [72].

where F is the state transition matrix, and $v(k)$ the white process noise which is Gaussian with zero mean and covariance Q. The measurements are taken in polar coordinates and ordered as

$$z(k) = [z_r(k)\ z_\theta(k)]' \tag{4.18}$$

with

$$z_r(k) = \sqrt{\eta(k)^2 + \xi(k)^2} + w_r(k) \tag{4.19}$$

$$z_\theta(k) = \tan^{-1}\left(\frac{\xi(k)}{\eta(k)}\right) + w_\theta(k) \tag{4.20}$$

and with $w_r(k)$ and $w_\theta(k)$ assumed to be zero mean independent white Gaussian noise with standard deviations σ_r and σ_θ.

We set the standard deviations of the noise in range and angle to be $\sigma_r = 0.2\,\mathrm{m}$ and $\sigma_\theta = 10^{-3}\,\mathrm{rad}$, respectively. The sampling interval between the measurements is $T = 1\,\mathrm{s}$. We choose the power spectral density of the process noise to be $\tilde{q} = 10^{-3}\,\mathrm{m}^2/\mathrm{s}^3$. The value of $\sqrt{\tilde{q}}$ is based on the expected maximum acceleration of the target in a sampling period. The simulated tracks do not have added process noise; this value is used only to account for possible maneuvers. The state transition matrix and the process noise covariance matrix are then given by

$$F = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.21}$$

$$Q = \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} & 0 & 0 \\ \frac{T^2}{2} & T & 0 & 0 \\ 0 & 0 & \frac{T^3}{3} & \frac{T^2}{2} \\ 0 & 0 & \frac{T^2}{2} & T \end{bmatrix} \tilde{q} \tag{4.22}$$

We simulate a 2-D scenarios, with the starting state for the target $x = [1050\,\text{km},\ -200\,\text{m/s},\ 2500\,\text{km},\ 300\,\text{m/s}]'$, which has a range of 3000 km. The shape of the measurement uncertainty region is shown in Figure 4.7. In our simulations we use $N = 10000$ particles in the RPF in the 2-D scenario. Track initialization is done by sampling particles in polar coordinates around the first two measurements using the covariance of the noise, converting the particles into Cartesian coordinates, and then performing two point differencing on pairs of particles to calculate velocities.[4] An example of this initialization for a target in the long range scenario is shown in Figure 4.8. Initializing in this way creates a set of particles with ranges close to the true range. Particles generated from the Gaussian approximation resulting from two point differencing using the first two measurements directly would result in a substantially lower initial range accuracy. Two-point differencing using the unbiased measurement conversion from polar to Cartesian coordinates was used in the initialization of the other filters [7].

[4]This is in contrast to other filters which use a batch consisting of a very large number of frames of data in their initialization [20].

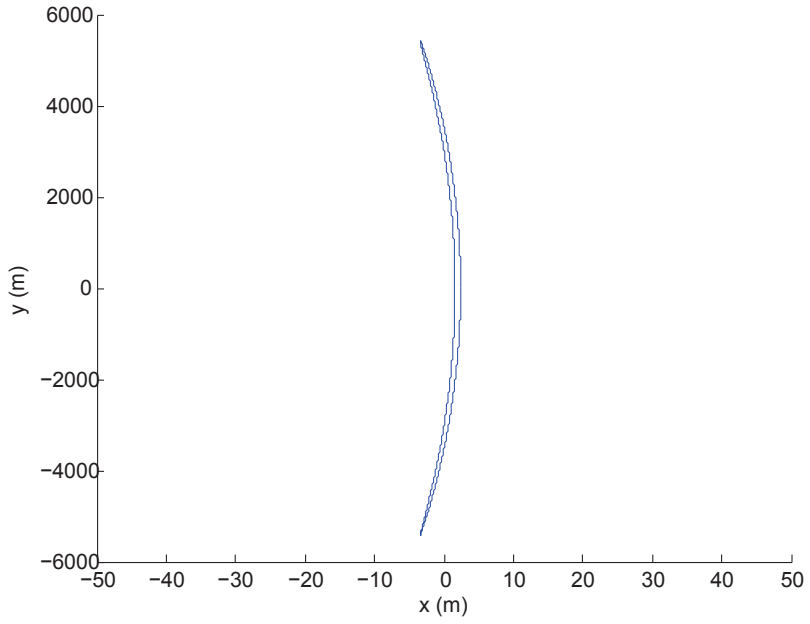


Figure 4.7: The banana shaped measurement uncertainty region, in two dimensions, using the range and accuracies of the simulated 2-D scenario. The target is shifted to the x-axis so that the shape of the bananas is easily seen (the abscissa scale is magnified to this purpose).

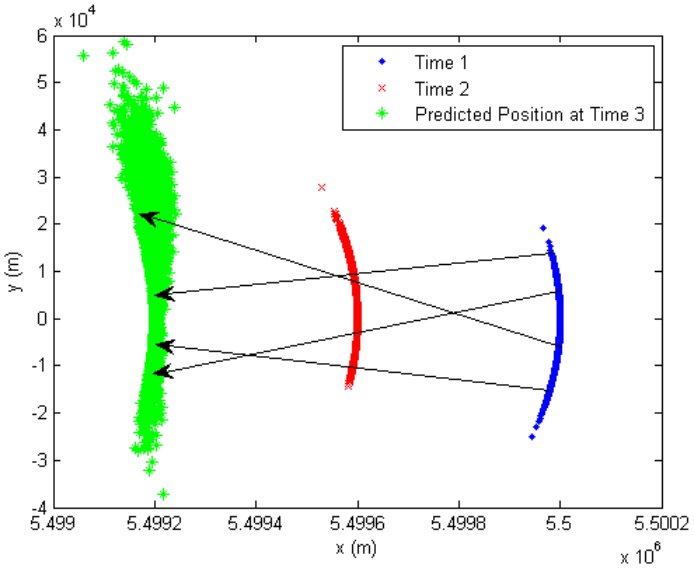


Figure 4.8: The initial set of particles used by the RPF (in 2-D), the leftmost group (one step prediction), is produced by performing two-point differencing on pairs of particles generated around the first and second measurements (the rightmost and center groups, respectively). The target is shifted on the x-axis so that the banana shapes are more easily seen. The arrows represent four examples of two-point differencing between pairs of particles.

4.5.2 3-D Scenario

We again use the discretized CWNA model, now with the state vector as

$$x = [\xi \ \dot{\xi} \ \eta \ \dot{\eta} \ \zeta \ \dot{\zeta}]' \tag{4.23}$$

The starting state for the long range target simulated is $x = [2520\,\text{km}, -200\,\text{m/s}, 1320\,\text{km}, 300\,\text{m/s}, 960\,\text{km}, -100\,\text{m/s}]'$, which has a range of 3000 km. The measurements are taken in r-u-v coordinates (range and direction cosines[5]) and ordered as

$$z(k) = [z_r(k) \ z_u(k) \ z_v(k)]' \tag{4.24}$$

with

$$z_r(k) = \sqrt{\xi(k)^2 + \eta(k)^2 + \zeta(k)^2} + w_r(k) \tag{4.25}$$

$$z_u(k) = \frac{\xi(k)}{\sqrt{\xi(k)^2 + \eta(k)^2 + \zeta(k)^2}} + w_u(k) \tag{4.26}$$

$$z_v(k) = \frac{\eta(k)}{\sqrt{\xi(k)^2 + \eta(k)^2 + \zeta(k)^2}} + w_v(k) \tag{4.27}$$

and with $w_r(k)$, $w_u(k)$, and $w_v(k)$ assumed to be zero mean independent white Gaussian noise with standard deviations σ_r, σ_u, and σ_v. In our simulations we set the standard deviations of the noise in range and angle to be $\sigma_r = 0.2\,\text{m}$ and $\sigma_u = \sigma_v = 10^{-3}\,\text{sin}$, respectively. We use $N = 300,000$ particles. Initialization was done with a method similar to the 2-D case. The running time of the RPF with this N, in MATLAB, was approximately 0.9 s per sampling time of the filter, i.e., faster than real time.

[5]While most of the literature dealing with tracking in 3-D assumes measurements in spherical coordinates, we use the real world radar coordinates (r-u-v), which do not lend themselves to debiasing like spherical coordinates ([8], Section 1.7.4).

4.6 Simulation Results

4.6.1 Accuracy

Figures 4.9 and 4.10 both show that the EKF diverges. The UKF has a very high position error (relative to the others) in the 2-D scenario and is not included at all in the 3-D simulation. The algorithm from [38] ran successfully on this scenario, but only with inflated process noise (the work [38] used it only in situations with high process noise). However, when using the low process noise in the scenario considered here, this filter diverged due to particle degeneracy.

In these scenarios the RPF performs the best in both position and range. The position RMSE of the RPF, shown in Figure 4.9a, only has a small improvement over the CbGMF, but the range RMSE in Figure 4.9b clearly demonstrates that the RPF offers a substantial improvement in range accuracy during the first half of the tracking period. This improvement is magnified in the 3-D scenario in Figures 4.10a and 4.10b.

We compare the performance of the RPF (in 2-D) for different values of the bandwidth h in Figure 4.11. It is clear from these results that the best value of h is approximately 0.5 as discussed following (4.15). All other simulated values produce worse position estimates. The chosen value had a minimal effect on the range estimates, but the smaller values made the filter inconsistent.

In these scenarios the BCRLB should not be attainable as the posterior is not Gaussian, and indeed the MCAEKF is relatively close to the bound in position

RMSE, but is far above it in range. However, we also see that the RPF outperforms the BCRLB in the 3-D scenario. This is a result of initializing the BCRLB with the sample covariance of the initial set of particles used in the RPF. This covariance does not adequately represent the initial set of particles.

The high range accuracy of the RPF can be partly attributed to the conversion of the particles between Cartesian and measurement coordinates in the algorithm. Specifically, calculating the position estimate in measurement coordinates takes full advantage of the shape the particles take (the banana or contact lens), which is very thin and elongated; there is a small deviation in range among particles. The two covariance-adaptive filters, besides approximating the banana shaped covariance regions with Gaussian ellipses, are designed to sacrifice some range accuracy for the sake of consistency. They are unable to meet the range accuracy of the RPF for a significant amount of time in the simulations.

4.6.2 Consistency

The normalized estimation error squared (NEES) can be used to determine if the size of a filter's state error is appropriate given the filter's state covariance. It is given by

$$\epsilon(k) = [x(k) - \hat{x}(k|k)]' P(k|k)^{-1} [x(k) - \hat{x}(k|k)] \tag{4.28}$$

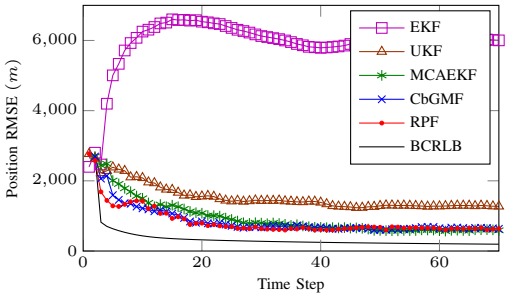
Letting $\bar{\epsilon}(k)$ denote the average NEES over N_{MC} Monte Carlo runs, $N_{\mathrm{MC}}\bar{\epsilon}(k)$ will have a chi-square density with $N_{\mathrm{MC}}n_x$ degrees of freedom if the filter is consistent and the estimation error is Gaussian, where n_x is the dimension of the state x.

As seen in Figures 4.9c and 4.10c, the EKF was not consistent in either of the scenarios[6] , and the UKF was not consistent in the 2-D scenario; their NEES was far larger than the upper bound so these filters are not desirable. The MCAEKF and the CbGMF are largely consistent. The NEES for the RPF is tends to be too small; the covariance yielded by the filter is too large. Figure 4.12 shows the position of the particles and their weights at one time step in a 2-D long range scenario with the target located on the x-axis; it depicts why the NEES would not follow the chi-square mean or bounds. The shape that the particles form in Cartesian coordinates, the banana, is clearly not Gaussian, and the extreme ends of the banana result in the calculated covariance (graphically represented by an ellipse) becoming too large along the range direction. We can instead calculate the NEES after converting the particles to measurement coordinates as shown in Figure 4.13. Here we can see that the ellipse corresponding to the covariance does a much better job covering the relevant area. In Figures 4.9d and 4.10d, the resulting NEES is much closer to the appropriate bounds. Note that in Cartesian coordinates we calculate the NEES using the full state (position and velocity), while in measurement coordinates we use position only.

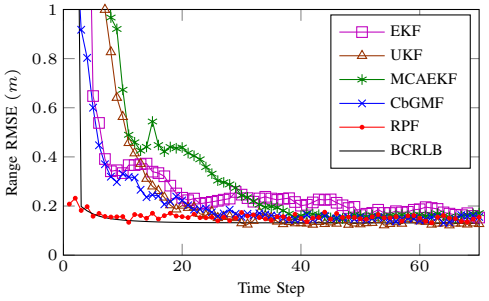
4.7 Conclusions

The regularized particle filter (RPF) has been shown to be the only viable filter for tracking in long range scenarios with polar or r-u-v measurements where

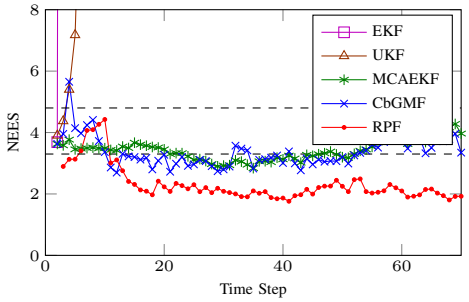
[6]Note that, as pointed out by a reviewer, when given perfect initialization the EKF is consistent in these scenarios, however, perfect initialization is not available in the real world.



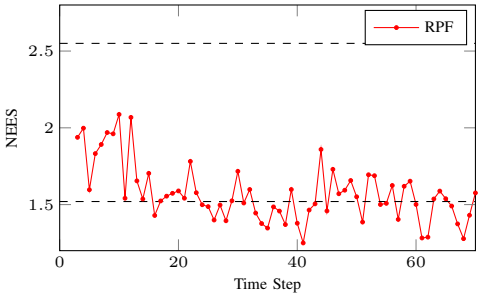
(a) Position RMSE



(b) Range RMSE

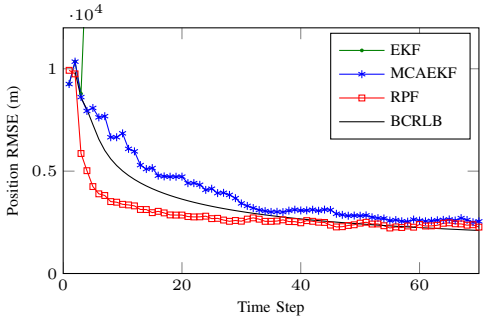


(c) NEES and 99% probability regions

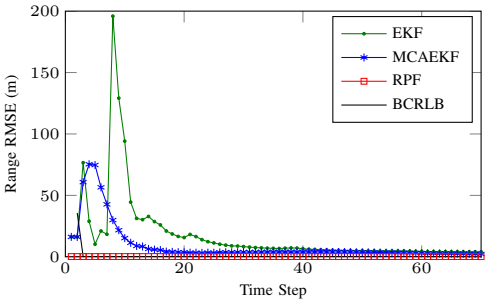


(d) NEES calculated in polar coordinates
and 99% probability regions

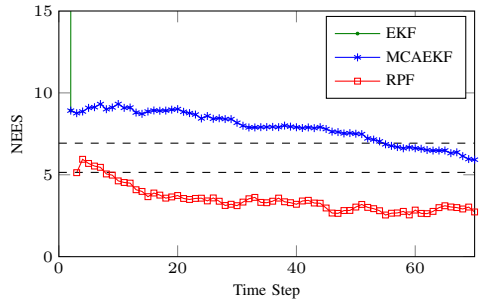
Figure 4.9: Results from the 2-D long range scenario from 100 Monte Carlo runs.



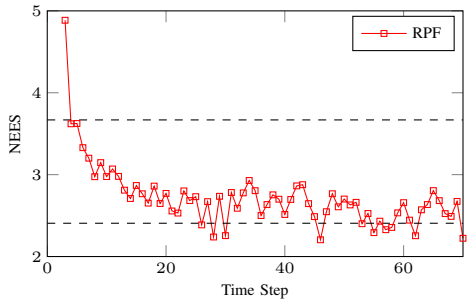
(a) Position RMSE



(b) Range RMSE

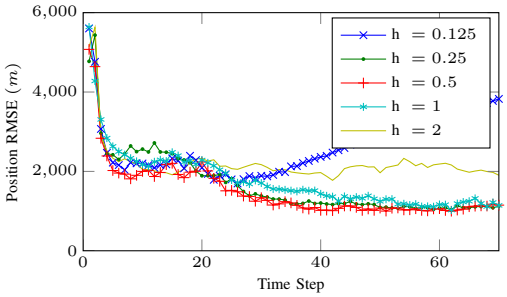


(c) NEES and 99% probability regions

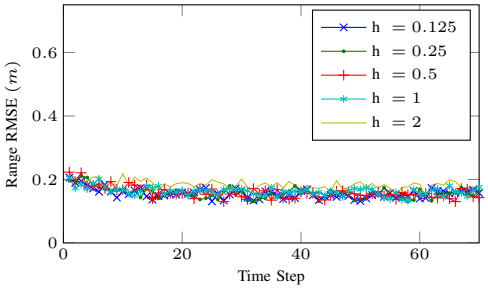


(d) NEES calculated in r-u-v coordinates and 99% probability regions

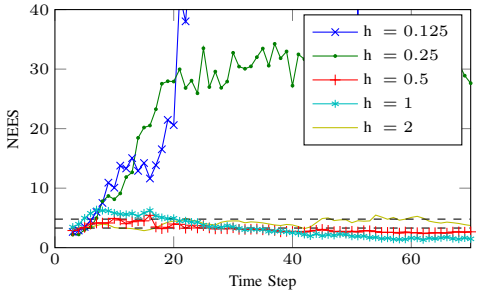
Figure 4.10: Results from the 3-D long range scenario from 100 Monte Carlo runs. Where not shown, the EKF is out of the bounds of the axes.



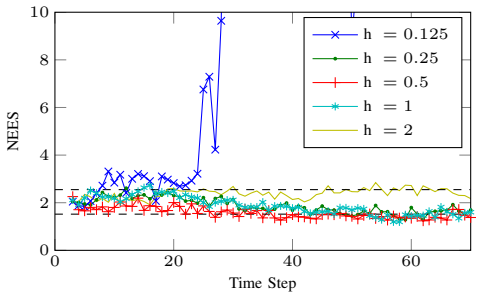
(a) Position RMSE



(b) Range RMSE



(c) NEES and 99% probability regions



(d) NEES calculated in polar coordinates and 99% probability regions

Figure 4.11: Performance of the RPF in the 2-D long range scenario for different values of the Epanechnikov kernel bandwidth h.

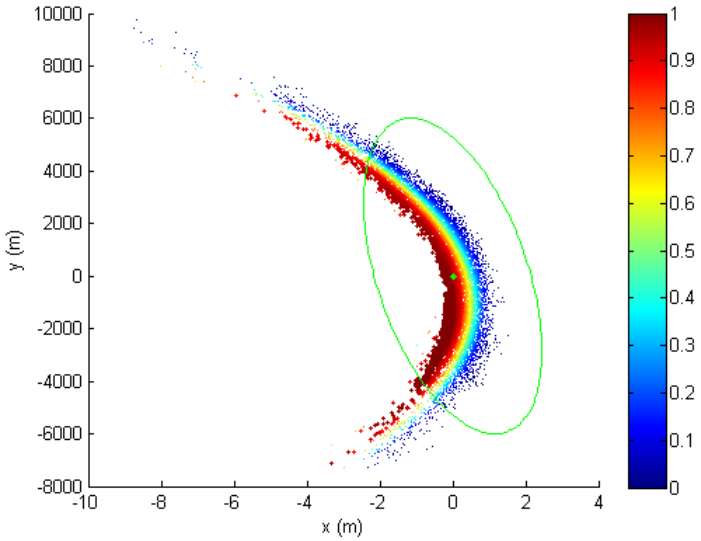


Figure 4.12: Particles colored and sized by their weight with their sample mean and covariance in Cartesian coordinates (2-D long range scenario). Note the highly non-elliptical spread of the particles.

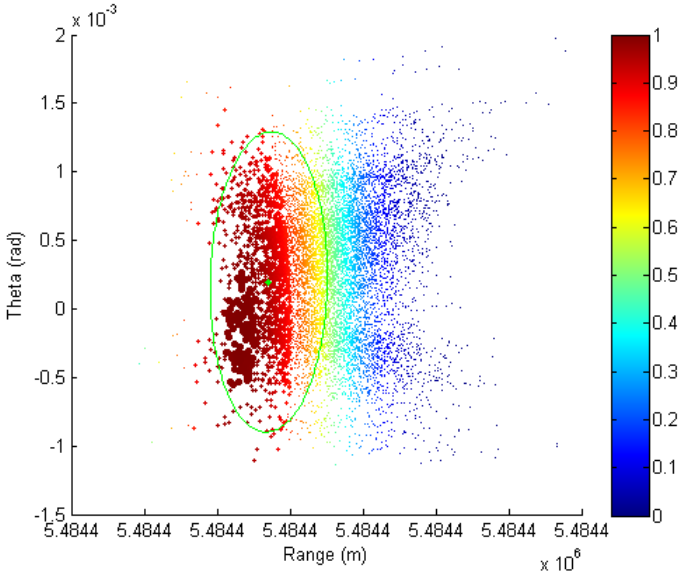


Figure 4.13: Particles colored and sized by their weight with their sample mean and covariance in polar coordiantes (2-D long range scenario). Note that the particles outside the ellipse have very low weight — the ellipse covers most of the probability mass.

the range accuracy is high, and the cross-range accuracy is low. The algorithm presented overcomes the two great difficulties that other particle filters face in these scenarios, namely particle degeneracy and sample impoverishment. The regularization was accomplished using an appropriately modified version of the Epanechnikov kernel suitable for state vectors.

Long-range high-bandwidth sensor measurements offer an environment in which it is very challenging either to track at all (the unadorned EKF does not) or track efficiently (like the UKF [69, 70]) making proper use of the high range accuracy. The Measurement Covariance Adaptive Extended Kalman Filter (MCAEKF) offers worthy competition; but we can do better. Specifically, the MCAEKF must inflate its assumed range accuracy during its initiation phases, leading to degraded estimation performance over the first portion of the track versus our RPF. The MCAEKF also appears to be efficient in terms of the BCRLB; but this is in the Cartesian space where the MCAEKF operates, and if projected into range its accuracy is always much poorer than that of the RPF – considerably worse even than the accuracy of the radar range measurements themselves. In summary: The RPF presented here exceeds significantly the performance of the MCAEKF in these scenarios.

Chapter 5

Fusion of Multipath Data with ML-PMHT for Very Low SNR Track Detection in an OTHR

5.1 Introduction

Over-the-horizon radar (OTHR) relies on signal refraction through the ionosphere to detect targets beyond the horizon. Due to the nature of the ionosphere, the signal from the radar may propagate via multiple paths, resulting in several target-originated detections. There is an ambiguity between detections and paths; the path corresponding to each target detection is not known. There are also measurements from false detections.

There are a wide range of approaches to the OTHR problem, varying in how detection, tracking, and association are handled. The multiple detection multiple hypothesis tracker (MD-MHT) [61] is formulated to solve the data association problem between measurements and measurement paths using an extended

multiframe assignment technique. Alternatively, a multihypothesis fusion algorithm, presented in [45, 58–60], is a measurement-level fusion algorithm using only measurements already associated with targets by another filter to calculate the probabilities of association hypotheses. In [1, 2] a method is proposed for joint multiple target ground track estimation and slant track association. Additionally they assume unknown ionospheric conditions. Their method shows an improvement in accuracy and the number of correct track and path assignments. The Signal Inversion for Target Extraction and Registration (SIFTER) signal processing algorithm developed in [27] provides a better detection of low SNR targets in clutter by solving for the scattering surface that reproduces the radar's measurements and has been demonstrated effectively on real OTHR data.

Other approaches include applying the probabilistic data association filter (PDAF) [14–16], the multipath probabilistic data association algorithm (MPDA) [47], and the Probabilistic Multi-Hypothesis Tracker (PMHT) [16] to OTHR data. An extension of the PDAF called the Multiple Model Unified PDAF (MM-UPDAF) is developed in [14]. The MM-UPDAF is designed to handle multiple nonuniform clutter regions. The SNR in [14] is unavailable as the parameters used to determine the performance of the MM-UPDAF are proprietary. The lowest SNR available from [47] and [61] is around 10 dB, with an ionosphere model similar to what we use in our simulations. We show that our algorithm with a VLO target SNR of 4 dB yields a high track detection probability (95%) and a very low false track rate (less than one per day) for the scenario considered.

A multipath Expectation Maximization algorithm is developed and applied to an OTHR scenario in [35]. Similar to the PMHT, it treats data association as missing data. It also treats propagation paths as missing data. The (single path) Maximum Likelihood Probabilistic Multi-Hypothesis Tracker (ML-PMHT) uses the log-likelihood function based on the PMHT model. The ML-PMHT has previously been formulated for single and multitarget [64, 65] scenarios. It has been shown to perform well even with very low target SNR.

In this paper we extend the ML-PMHT formulation from [64, 65]. We present a generalized form of the ML-PMHT that accounts for multiple possible propagation paths. We apply this algorithm to an OTHR scenario. Unlike the MD-MHT [61], no data association is required. The ML-PMHT considers simultaneously all the measurements without knowing their origins or propagation paths and, remarkably, has *linear complexity in the number of measurements*. The ML-PMHT performs fusion of the multipath data in the presence of false measurements.

Section 5.2 briefly describes the ML-PMHT for a single target case and the extension to allow for multiple paths. Section 5.3 describes the multiple path extension to the ML-PDA. Section 5.4 presents the OTHR model used for simulation. Section 5.5 discusses the performance of the ML-PMHT from Monte Carlo testing. Section 5.6 develops the Cramér-Rao Lower Bound for the multipath ML-PMHT.

5.2 ML-PMHT

5.2.1 Single Target ML-PMHT

The ML-PMHT log-likelihood ratio (LLR) for the motion parameter of a single target is developed in [64]. This LLR is given by

$$\begin{aligned}\Lambda(\mathbf{x}; Z) &\triangleq \ln\left\{\frac{p(Z|\mathbf{x})}{p(Z|\text{all false})}\right\} \\ &= \sum_{i=1}^{N_w}\sum_{j=1}^{m_i} \ln\left\{\pi_0 + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}(i)]\rho_j(i)\right\}\end{aligned} \tag{5.1}$$

with

$$Z \triangleq \{\{\mathbf{z}_j(i)\}_{j=1}^{m_i}\}_{i=1}^{N_w} \tag{5.2}$$

Here N_w is the number of scans in the batch (the window length), and m_i is the number of measurements in the i^{th} scan (frame). The parameter $\mathbf{x}$ determines the target state $\mathbf{x}(i)$ in a deterministic way (we use a constant velocity model[1] i.e., $\mathbf{x} = [s_t(1), \dot{s}_t]'$, where $s_t(1)$ and $\dot{s}_t$ are the initial position and velocity of the target, respectively). The prior probabilities that a measurement occurred due to clutter or due to a target are given by π_0 and π_1, respectively. These values are related to the probability of detection, P_D, and the probability of false alarm in a resolution cell, P_{FA}. The volume of the search region is V and a measurement, which didn't occur due to a target, has a uniform pdf in V. The j^{th} measurement in the i^{th} scan is $\mathbf{z}_j(i)$ and its associated amplitude likelihood ratio is $\rho_j(i)$. Finally, $p[\mathbf{z}_j(i)|\mathbf{x}(i)]$ is a Gaussian with mean determined by the

[1] Any arbitrary deterministic motion model can be used, such as deterministic motion in a known gravitational field [6].

target state parametrization $\mathbf{x}(i)$, and with the measurement noise covariance matrix. The amplitude likelihood ratio serves as a feature discriminant between the target originated measurements and the false ones due to spurious detections.

The pdfs $p[Z(i)|\mathbf{x}(i)]$ (likelihood of the target present hypothesis) and $p[Z(i)|\text{all false}]$ (likelihood of the target absent hypothesis) are derived using the ML-PMHT assumptions [64]:

- There is a single target with known probability of detection.
- Any number of measurements in a scan can be assigned to the target.
- The motion of the target is deterministic.
- False detections are uniformly distributed.
- The number of false detections is Poisson distributed with known density.
- Amplitudes of target and false detections are Rayleigh distributed with known distribution.
- Target measurements are corrupted with zero-mean Gaussian noise.
- Measurements at different times, conditioned on the parameterized state, are independent.

These likelihoods are then given by

$$p[Z(i)|\mathbf{x}(i)] = \prod_{j=1}^{m_i} \left\{ \frac{\pi_0}{V} p_0^{\tau}[a_j(i)] + \pi_1 p[\mathbf{z}_j(i)|\mathbf{x}(i)] p_1^{\tau}[a_j(i)] \right\} \tag{5.3}$$

$$p[Z(i)|\text{all false}] = \prod_{j=1}^{m_i} \frac{1}{V} p_0^{\tau}[a_j(i)] \tag{5.4}$$

where $p_0^{\tau}[a_j(i)]$ and $p_1^{\tau}[a_j(i)]$ are the pdfs of a false alarm and target measurement amplitude conditioned on exceeding the threshold τ, respectively.

5.2.2 The Multipath ML-PMHT Log-Likelihood Ratio for OTHR

The LLR of the generalized ML-PMHT that allows multiple propagation paths is given by

$$\Lambda(\mathbf{x}; Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ \pi_0 + \pi_1 V \rho_j(i) \sum_{\ell=1}^{n_p} p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell] P[\ell] \right\} \tag{5.5}$$

where ℓ is used to denote which path the signal took, $P[\ell]$ is the probability of path ℓ being taken, and n_p is the total number of possible paths. The mean of the Gaussian $p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell]$ is $f_\ell(\mathbf{x}(i))$, where f_ℓ is the function that transforms the target state $\mathbf{x}(i)$ into the measurement space via path ℓ. The covariance matrix for this Gaussian is the measurement noise covariance for a measurement from path ℓ. Note that, for simplicity, we have assumed that $\rho_j(i)$ is the same for each path ℓ (a path dependent LLR can be used if available).

5.3 ML-PDA

We can extend the single-path ML-PDA likelihood presented in [64] to allow for multiple paths by applying the total probability theorem. For a single scan

this results in

$$\Lambda(\mathbf{x}; Z) = \sum_{n_d=0}^{n_p} p(\mathbf{z}|\mathbf{x}, n_d) P(n_d) \tag{5.6}$$

$$P(n_d) = P_D^{n_d}(1 - P_D)^{n_p - n_d} \binom{n_p}{n_d} \tag{5.7}$$

$$p(\mathbf{z}|\mathbf{x}, n_d) = \frac{1}{\binom{m}{n_d}\binom{n_p}{n_d} n_d!} \cdot \sum_{M \in \mathcal{M}_{n_d}} \sum_{A \in \mathcal{A}_{n_d}} p[\{\mathbf{z}_k\}_{k \notin M} | \text{“clutter”}] \prod_{j=1}^{n_d} p[\mathbf{z}_{M(j)} | \mathbf{x}, A(j)] \tag{5.8}$$

where n_d is the number of detections, and P_D is the probability of detection. $\mathcal{M}_{n_d}$ is the set of all unordered n_d-tuples of measurement indices. It contains $\binom{m}{n_d}$ n_d-tuples. $\mathcal{A}_{n_d}$ is the set of all ordered n_d-tuples of path indices. This set contains $\binom{n_p}{n_d} n_d!$ n_d-tuples. The ML-PDA gives similar results to the ML-PMHT in very low clutter scenarios, but is significantly more complex. For scenarios with a large amount of clutter (like the ones we are exploring in this paper) the ML-PDA becomes intractable due to the number of terms in the double summation in equation (5.8). Its CRLB is also complicated to determine since it requires extensive Monte Carlo simulations [11]. We choose the ML-PMHT for its simplicity and effectiveness. In a single scan i the ML-PMHT has $m_i n_p$ terms — linear complexity — while the ML-PDA has $\sum_{n_d=0}^{n_p} \binom{m_i}{n_d}\binom{n_p}{n_d} n_d!\, n_d = m_i n_p \sum_{n_d=0}^{n_p} (n_d - 1)! \binom{m_i - 1}{n_d - 1}\binom{n_p - 1}{n_d - 1}$ terms and therefore suffers from a combinatorial explosion with increasing m_i.

5.4 OTHR Model

We investigate two two-dimensional OTHR scenarios which assume the target to be in a great circle plane on the earth's surface as shown in Figure 5.1, and a three-dimensional scenario where the target is on the surface of a sphere. We use a two-layer reflection model (spherical mirror model) for the ionosphere.[2] In this model the signal may reflect from either layer of the ionosphere resulting in multiple (up to four) round-trip propagation paths. In the 2-D and 3-D scenarios the radar measures slant range, slant range rate, and amplitude. In the 3-D scenario it also measures azimuth.

5.4.1 Measurement Amplitudes

We model the amplitudes of the measurements according to a Swerling I model [6]. The amplitude is Rayleigh distributed with pdfs

$$p_0(a) = ae^{-\frac{a^2}{2}} \qquad a \geq 0 \tag{5.9}$$

$$p_1(a) = \frac{a}{1+d}e^{-\frac{a^2}{2(1+d)}} \qquad a \geq 0 \tag{5.10}$$

for the noise only and target, respectively. Here d is the expected SNR of the target in a resolution cell. For a chosen threshold τ we have

$$P_D = \int_\tau^\infty p_1(a)da \tag{5.11}$$

$$P_{FA} = \int_\tau^\infty p_0(a)da \tag{5.12}$$

[2]The actual paths are subject to refraction, which requires numerical algorithms for ray tracing. The reflection model used here is a simplified one, which, however, captures the essence of the OTHR.

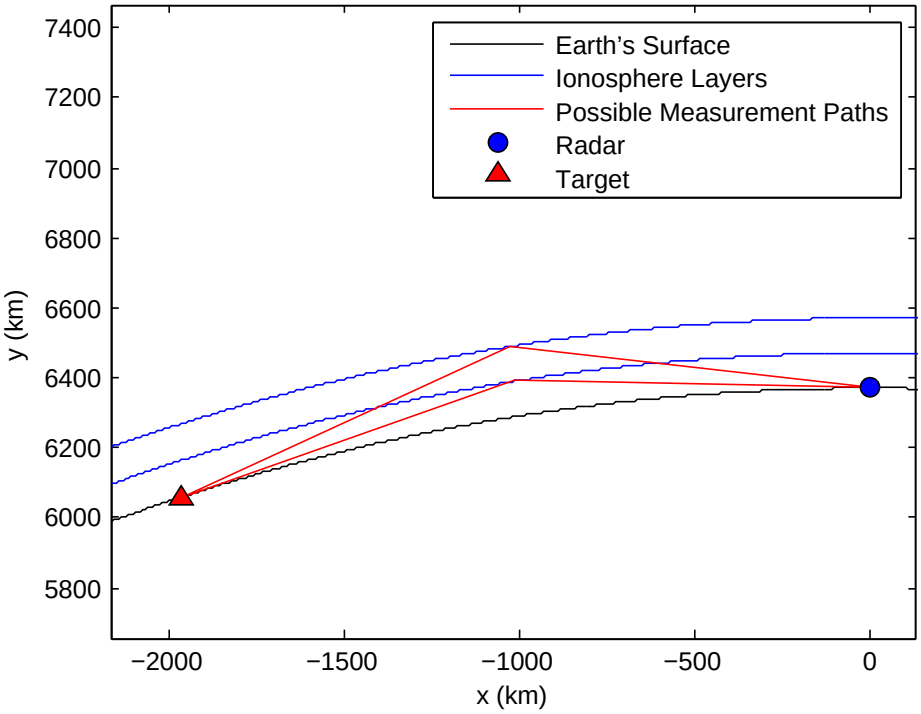


Figure 5.1: The 2-D OTHR scenario with a reflection ionosphere model (spherical mirror model).

The pdfs of the amplitude of a measurement given that it has exceeded the threshold τ are

$$p_0^\tau(a) = \frac{1}{P_{FA}} p_0(a) \qquad a \geq \tau \tag{5.13}$$

$$p_1^\tau(a) = \frac{1}{P_D} p_1(a) \qquad a \geq \tau \tag{5.14}$$

and the amplitude likelihood ratio is then

$$\rho_j(i) = \frac{p_1^\tau[a_j(i)]}{p_0^\tau[a_j(i)]} \tag{5.15}$$

5.4.2 Measurements

The OTHR measures both position and velocity of the target via slant range and slant range rate measurements. The equations of the measurements, given the signal reflected off the lower layer in both directions, are given below.[3] Defining

$$r_1 \triangleq 4\sqrt{h_1^2 - 2R_\oplus(h_1 + R_\oplus)\cos\left(\frac{s_r - s_t}{2R_\oplus}\right) + 2h_1R_\oplus + 2R_\oplus^2} \tag{5.16}$$

$$\dot{r}_1 \triangleq \frac{\partial r_1}{\partial s_t}\frac{\partial s_t}{\partial t} = -\frac{2\sin\left(\frac{s_r - s_t}{2R_\oplus}\right)(R_\oplus + h_1)}{\sqrt{2R_\oplus h_1 + 2R_\oplus^2 + h_1^2 - 2R_\oplus\cos\left(\frac{s_r - s_t}{2R_\oplus}\right)(R_\oplus + h_1)}}\dot{s}_t \tag{5.17}$$

[3]The signal propagates forward and is reflected in the plane of the great circle defined by the radar and the target. We assume that the antenna beam illuminating the target is in this plane. This beam corresponds to the measured azimuth of the reflection from the target.

one has

$$z_{r_1} = r_1 + w_{r_1} \tag{5.18}$$

$$z_{\dot{r}_1} = \dot{r}_1 + w_{\dot{r}_1} \tag{5.19}$$

Here h_1 is the height of the lower ionosphere layer. The radius of the earth is $R_\oplus$. The locations of the radar and target on the surface of the earth (on the great circle connecting them) are given by s_r and s_t, respectively. An illustration of the geometry of this problem is shown in Fig. 5.2. The velocity of the target along the great circle is $\dot{s}_t$. The noise terms, w_{r_1} and $w_{\dot{r}_1}$, are zero-mean, Gaussian, independent of each other, and with variances σ_r and $\sigma_{\dot{r}}$, respectively. We assume, for simplicity, the same noise variances on the other paths.

Given that the signal reflected off the upper layer only (with height h_2), we can find similar equations for r_2, $\dot{r}_2$, z_{r_2}, and $z_{\dot{r}_2}$, with noises w_{r_2} and $w_{\dot{r}_2}$. The equations for the measurements resulting from the remaining two paths, where the signal reflects off of alternate layers, can then be expressed as

$$z_{r_3} = \frac{1}{2}(r_1 + r_2) + w_{r_3} \tag{5.20}$$

$$z_{\dot{r}_3} = \frac{1}{2}(\dot{r}_1 + \dot{r}_2) + w_{\dot{r}_3} \tag{5.21}$$

$$z_{r_4} = \frac{1}{2}(r_1 + r_2) + w_{r_4} \tag{5.22}$$

$$z_{\dot{r}_4} = \frac{1}{2}(\dot{r}_1 + \dot{r}_2) + w_{\dot{r}_4} \tag{5.23}$$

The azimuth measurement (used only in the 3-D scenario) is given by

$$z_\theta = \theta + w_\theta \tag{5.24}$$

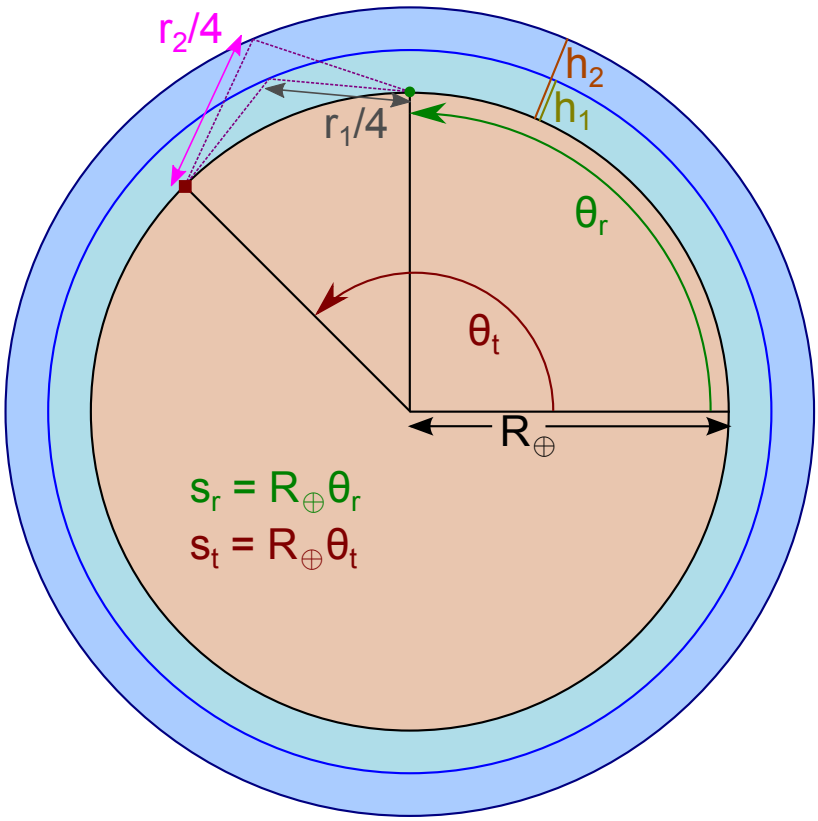


Figure 5.2: Illustration of the geometry used to derive the equations for the measurements. Here θ_r and θ_t are the angles in polar coordinates of the radar and the target, respectively.

where θ is the true azimuth of the target, and the noise term w_θ is zero-mean, Gaussian, and has variance σ_θ.

5.4.3 2-D Simulation Parameters

We simulated a target with an initial position 2000 km away from the radar, moving with a constant speed of 10 m/s towards it. The other values used in the 2-D simulations are given in Tables 5.1 and 5.2. Figures 5.3 and 5.4 show the measurements used (after amplitude thresholding) in one run of the tracker from scenario 1. False measurements are generated uniformly in the measurement space. Note that, due to the very low SNR in a cell, P_D is a meager 0.34 and the high P_{FA} leads to 60 false measurements per scan. Also note that there are usually zero to three target originated measurements in each scan (rarely all four) and the overwhelming number of false measurements, which, however, can be successfully handled by the multipath ML-PMHT track detector.

5.4.4 3-D Simulation Parameters

We also simulated a target starting at 2000 km away from the radar and 0 azimuth. It is moving with a constant speed of 10 m/s with an initial course of 5°. The target follows the great circle starting from these initial conditions. The other values used in the 3-D simulation are given in Tables 5.1 and 5.2. The very low SNR in a cell now leads to 72 false measurements per scan.

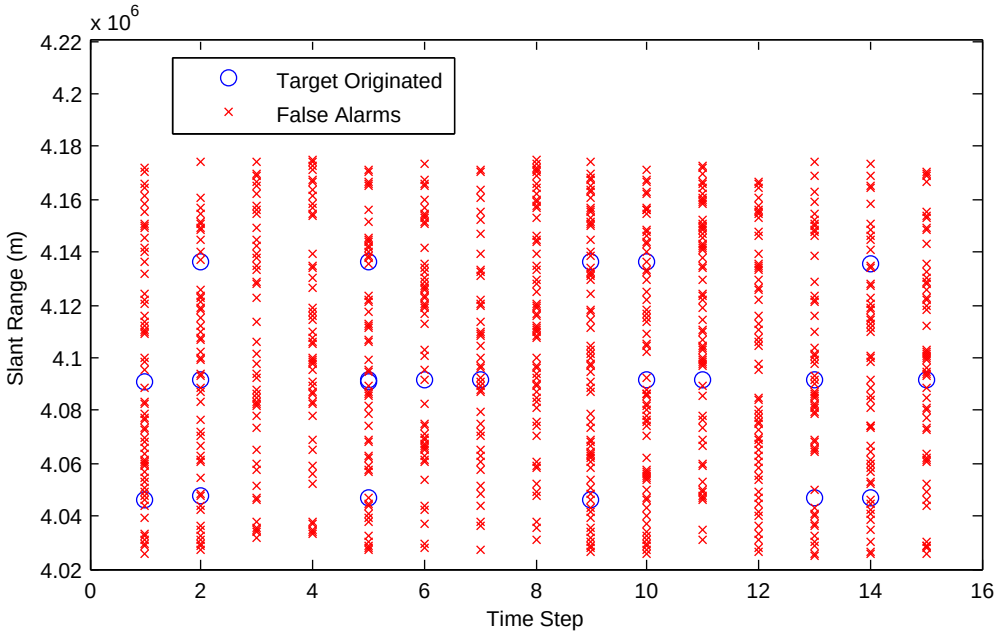


Figure 5.3: Slant range measurements in one batch in 2-D scenario 1 (4dB post-signal processing SNR).

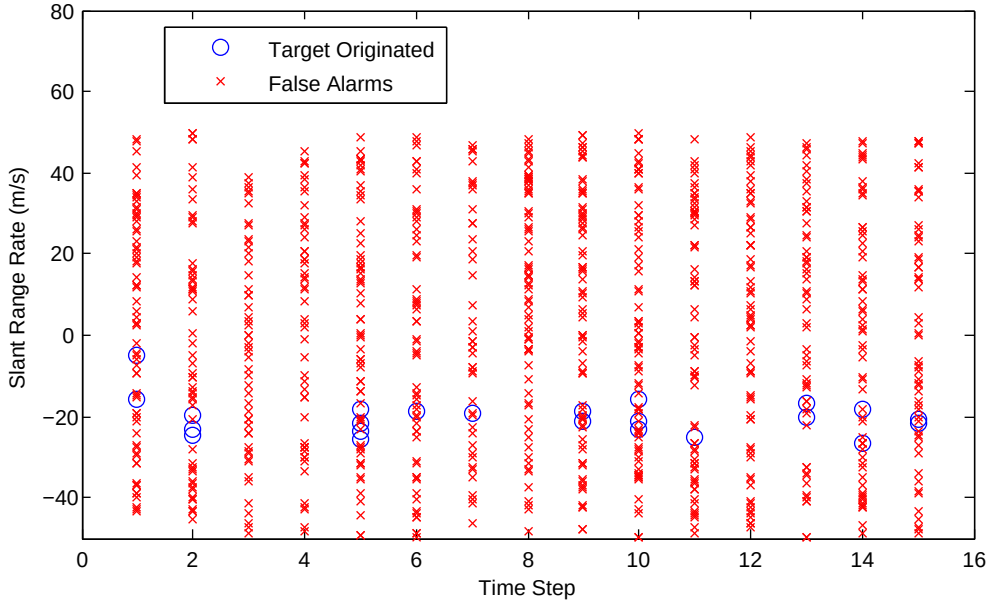


Figure 5.4: Slant range rate measurements in one batch in 2-D scenario 1 (4dB post-signal processing SNR).

N_w	15
Time between scans	1 s
σ_r	300 m
$\sigma_{\dot{r}}$	5 m/s
SNR in a cell	**2.5 = 4 dB**
$R_\oplus$	6371 km
Ionosphere lower layer height	100 km
Ionosphere upper layer height	200 km
$P[\ell]$ for all ℓ	0.25

Table 5.1: Scenario parameters used in the both the 2-D and 3-D simulations.

	2-D Scenario 1	2-D Scenario 2	3-D Scenario
Resolution cell size	600 m x 10 m/s	15000 m x 10 m/s	1200 m x 20 m/s x 1.2°
Search region size	150 km x 100 m/s	150 km x 100 m/s	150 km x 100 m/s x 90°
Number of cells	2500	100	46875
V	$1.5 \cdot 10^7$ m^2/s	$1.5 \cdot 10^7$ m^2/s	$2.4 \cdot 10^7$ m^2/s x rad
σ_θ	N/A	N/A	0.3°
Amplitude detection threshold τ	2.7	1.7	3.6
P_D for each path	0.34	0.66	0.16
P_{FA} in a cell	0.024	0.24	0.0015
Expected number of false alarms per scan	60	24	72
π_0	0.9776	0.8991	0.9913

Table 5.2: Scenario parameters used in the 2-D and 3-D simulations.

5.5 Performance of the Track Detector

5.5.1 2-D Results

The LLR of the ML-PMHT for a single run is shown in Figures 5.5 and 5.6 for the first 2-D scenario. The plot is centered at the true target location. There are five peaks resulting from path ambiguity. The central peak (the correct one), however, is easily distinguishable from the side peaks. It is also much higher than any peak occurring due to clutter.

We use a simple grid search with 1 km spacing in range, and 20 m/s spacing in velocity to get into the neighborhood of the global maximum of (5.5). For simplicity, no target feature was used. We then run a local optimization routine from MATLAB using an interior-point algorithm on the highest valued point

from the grid search to produce the final state estimate. In the first 2-D scenario this takes approximately 8 seconds per run in MATLAB (faster than real time), and from 10000 Monte Carlo runs the root mean square (RMS) errors for position and velocity at the end of the batch were 40.7 m and 0.7 m/s, respectively. There were no false tracks or missed tracks.

We also applied the MD-MHT [61] to the first 2-D scenario. Using perfect initialization and a sliding window of size 2, the RMS errors from 100 Monte Carlo runs for position and velocity at the end of the run were 83 m and 5.8 m/s, respectively, i.e., significantly larger than the ML-PMHT. An extended Kalman filter was used to update the track with the measurement-path combinations chosen by the algorithm. In the MD-MHT, increasing the window size N_w rapidly increases the computational requirements of the algorithm. The number of hypotheses for a single target scenario depends on the number of paths and measurements and is approximately $(N_{\text{paths}} N_{\text{meas}})^{N_w}$, which quickly becomes intractable.

Using the same grid search method for the second scenario 2, the RMS errors from the ML-PMHT for position and velocity at the end of the batch were 27.2 m and 0.4 m/s, respectively, from 100 Monte Carlo runs, also with no false tracks.

We also ran the first 2-D scenario with different values for the SNR and threshold τ. We chose τ such that P_D remained fixed at 0.34. These results are shown in Table 5.3. In the lowest SNR case (4 dB) the track was detected in each of the 10^4 runs. The algorithm was demonstrated to yield a *track detection probability*, P_{DT}, higher than 95%. Also no false tracks were detected by the

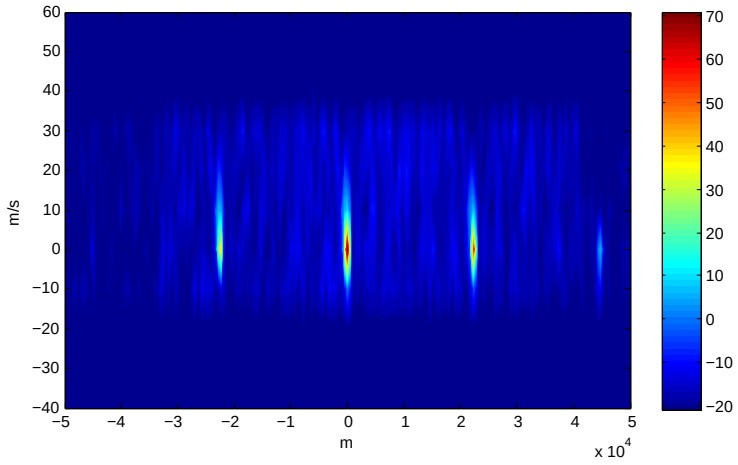


Figure 5.5: The log-likelihood ratio centered on the true target state from 2-D scenario 1.

algorithm in these 10^4 runs, thus the *probability of false track*, P_{FT}, is at most 10^{-4} for the 15 s time interval. Based on this, the *false track rate* (over 24 hours) is 0.6/day.

5.5.2 3-D Results

Figure 5.7 shows the LLR surface using the true values for azimuth and course. Similarly, Figure 5.8 shows the LLR surface using the true values for range and speed. We use MATLAB's GlobalSearch algorithm to perform the optimization. From 100 Monte Carlo runs the RMSE values were 3.3 km in position, 54 m in range, and 21 m/s in velocity (in the range direction 0.86 m/s while in the crossrange direction 21 m/s; the latter is due to the fact that the crossrange rate is based on the 0.3° azimuth measurement, which maps to 5 mrad x 2000 km = 10 km crossrange errors, i.e., extremely large).

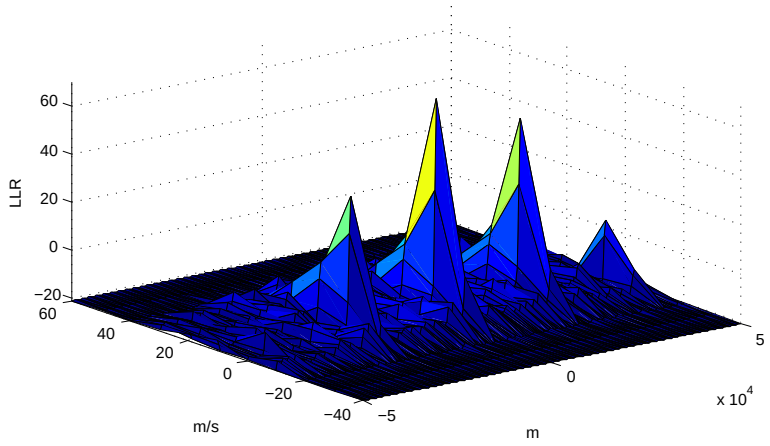


Figure 5.6: The log-likelihood ratio centered on the true target state from 2-D scenario 1.

The algorithm's running time in the 3-D scenario was approximately 2 minutes per run (on 15 s of data) in MATLAB. Therefore, this algorithm can run at least one order of magnitude faster, i.e., it is real time capable if it is implemented in a faster programming language, such as C.

5.6 Multipath Fusion ML-PMHT Cramér-Rao Lower Bound

We develop the Cramér-Rao Lower Bound (CRLB) [7] for the multipath fusion ML-PMHT and show that it is statistically efficient in the first 2-D scenario. We can assume all scans to be independent and also assume the measurements in each scan to be independent. The Fisher Information Matrix (FIM) $\mathbf{J}$ will then be the sum of the FIM's $\mathbf{J}_{i,j}$ of each measurement,

$$\mathbf{J} = \mathrm{E}\{(\nabla_{\mathbf{x}} \ln p[\mathbf{Z}|\mathbf{x}])(\nabla_{\mathbf{x}} \ln p[\mathbf{Z}|\mathbf{x}])^T\}|_{\mathbf{x}=\mathbf{x_0}} = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \mathbf{J}_{i,j} \tag{5.25}$$

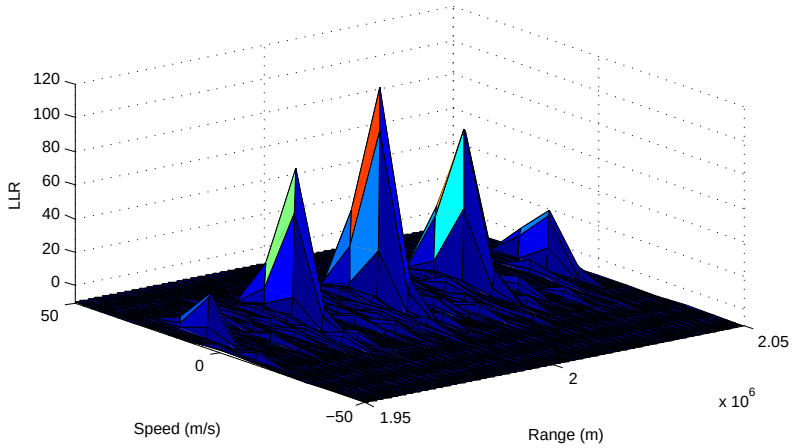


Figure 5.7: The log-likelihood ratio at the true values for azimuth and course from the 3-D scenario.

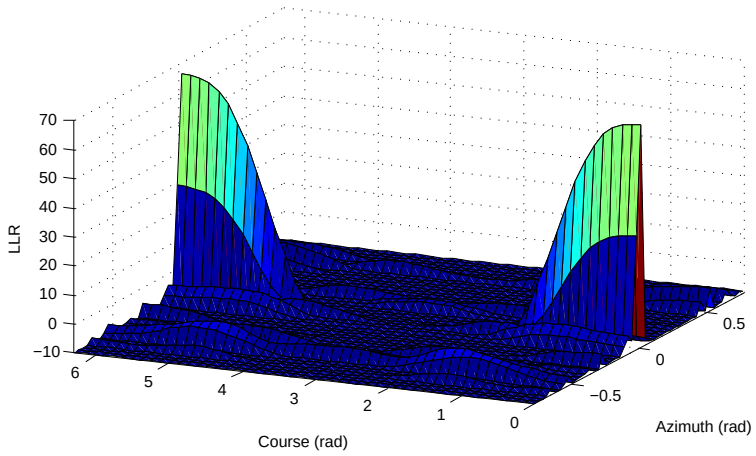


Figure 5.8: The log-likelihood ratio at the true values for range and speed from the 3-D scenario.

where

$$\mathbf{J}_{i,j} = \mathrm{E}\{(\nabla_{\mathbf{x}(i)} \ln p[\mathbf{z}_j(i)|\mathbf{x}(i)])(\nabla_{\mathbf{x}(i)} \ln p[\mathbf{z}_j(i)|\mathbf{x}(i)])^T\}|_{\mathbf{x}(i)=\mathbf{x_0}(i)} \tag{5.26}$$

The state vector $\mathbf{x}(i)$ is given by

$$\mathbf{x}(i) = [s_t(i),\ \dot{s}_t]' \tag{5.27}$$

where $s_t(i)$ is the target's position at time i, and $\dot{s}_t$ is the target's velocity. The function that transforms $\mathbf{x}(i)$ into the measurement space via the path that reflects both ways off the lower layer only is expressed as (5.28) with derivatives given by (5.29) and (5.30). The functions $f_\ell(\mathbf{x}(i))$ for the other paths (and their derivatives) can be found similarly.

$$f_1(\mathbf{x}(i)) = \begin{bmatrix} 4\sqrt{h_1^2 - 2R_\oplus(h_1+R_\oplus)\cos\left(\frac{s_r - s_t(i)}{2R_\oplus}\right) + 2h_1R_\oplus + 2R_\oplus^2} \\ -\frac{2\sin\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}{\sqrt{2R_\oplus h_1 + 2R_\oplus^2 + h_1^2 - 2R_\oplus\cos\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}}\dot{s}_t(i) \end{bmatrix} \tag{5.28}$$

$$\frac{\partial}{\partial s_t(i)} f_1(\mathbf{x}(i)) = \begin{bmatrix} -\frac{2\sin\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}{\sqrt{2R_\oplus h_1 + 2R_\oplus^2 + h_1^2 - 2R_\oplus\cos\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}} \\ \frac{\cos\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)\dot{s}_t(i)}{R_\oplus\sqrt{2R_\oplus h_1 + 2R_\oplus^2 + h_1^2 - 2R_\oplus\cos\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}} - \frac{\sin\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)^2(R_\oplus+h_1)^2\dot{s}_t(i)}{[2R_\oplus h_1 + 2R_\oplus^2 + h_1^2 - 2R_\oplus\cos\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)]^{3/2}} \end{bmatrix} \tag{5.29}$$

$$\frac{\partial}{\partial \dot{s}_t(i)} f_1(\mathbf{x}(i)) = \begin{bmatrix} 0 \\ -\frac{2\sin\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}{\sqrt{2R_\oplus h_1 + 2R_\oplus^2 + h_1^2 - 2R_\oplus\cos\left(\frac{s_r-s_t(i)}{2R_\oplus}\right)(R_\oplus+h_1)}} \end{bmatrix} \tag{5.30}$$

$$p[\mathbf{z}_j(i)|\mathbf{x}(i)] = \frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell] P(\ell) \tag{5.31}$$

$$\nabla_{\mathbf{x}(i)} \ln p(\mathbf{z}_j(i)|\mathbf{x}(i)) = \frac{\pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} P(\ell)|2\pi\mathbf{R}|^{-\frac{1}{2}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]} \mathbf{D}_\ell^T(i)\mathbf{R}^{-1}[\mathbf{z}_j(i) - f_\ell(\mathbf{x}(i))]}{\frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} P(\ell)|2\pi\mathbf{R}|^{-\frac{1}{2}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]}} \tag{5.32}$$

$$\mathbf{J}_{i,j} = \int_\tau^\infty \iint_V \frac{\frac{(\pi_1 p_1^\tau[a_j(i)])^2}{|2\pi\mathbf{R}|} \sum_{\ell=1}^{n_p} \mathbf{A}_\ell(i) \sum_{\ell=1}^{n_p} \mathbf{A}_\ell^T(i)}{\frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} P(\ell)|2\pi\mathbf{R}|^{-\frac{1}{2}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]}} d\mathbf{z}_j(i) da_j(i) \tag{5.33}$$

$$\mathbf{A}_\ell(i) = P(\ell) e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]} \mathbf{D}_\ell^T(i)\mathbf{R}^{-1}[\mathbf{z}_j(i) - f_\ell(\mathbf{x}(i))] \tag{5.34}$$

The multipath ML-PMHT likelihood for a single measurement is given by (5.31). The gradient of the logarithm of this likelihood gives (5.32), where $\mathbf{D}_\ell(i)$ is the Jacobian of $f_\ell(\mathbf{x}(i))$. Finally, combining equations (5.26) and (5.32) gives us the FIM of one measurement, which has to be evaluated numerically, as (5.33).

Using the parameters given in Table 5.1, the CRLB is 39.57 m and 0.6595 m/s for the position and velocity, respectively. From 10000 Monte Carlo runs for the lowest SNR = 4 dB the standard error of the sample variance is 0.5596 m for position, and 0.009327 m/s for velocity [7]. This gives the 95% (2-sigma) intervals of [38.45 m, 40.69 m] and [0.6408 m/s, 0.6782 m/s] for position and velocity, respectively. Since the RMSE values from the *multipath fusion ML-PMHT* (which were 40.67 m and 0.6760 m/s) are within these intervals, it *is a statistically efficient estimator*. We also include the CRLB for different values of the SNR and τ in Table 5.3.

	Cell P_{FA}	Expected number of false alarms (false measurements) per scan	Position RMSE (m)	Velocity RMSE (m/s)	Position CRLB (m)	Velocity CRLB (m/s)
SNR = 10 dB, $\tau = 4.84$	$7 \cdot 10^{-6}$	0.02	32.90	0.5449	33.11	0.5519
SNR = 7 dB, $\tau = 3.6$	0.0015	4	35.30	0.5647	34.36	0.5727
SNR = 6 dB, $\tau = 3.3$	0.0043	11	37.10	0.6037	35.57	0.5929
SNR = 4 dB, $\tau = 2.73$	0.024	60	40.67	0.6760	39.57	0.6595

Table 5.3: Results for various SNR values in the first 2-D scenario from 1000 Monte Carlo runs (results for SNR = 4 dB are from 10000 Monte Carlo runs). The measurement detection threshold τ is chosen such that the single-measurement P_D is fixed at 0.34.

5.7 False Track and Target Track Detection Probabilities

We use the methods in [62, 63] to determine a threshold for the probability of false track, P_{FT}, and then calculate the probability of track detection, P_{DT}, for the first 2-D scenario with 4 dB SNR presented in Section 5.4.3.

5.7.1 Probability of False Track

We begin with the multipath LLR for a single measurement, $\mathbf{z}_j(i)$, and its corresponding amplitude LLR, $\rho_j(i)$,

$$\Lambda_{i,j}[\mathbf{z}_j(i)] = \ln\left\{\pi_0 + \pi_1 V \rho_j(i) \sum_{\ell=1}^{n_p} p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell] P[\ell]\right\} \tag{5.35}$$

and treat $\mathbf{z}_j(i) \in \mathbb{R}^2$ and $\rho_j(i) \in \mathbb{R}^+$ as random variables. Equation (5.35) is a function that transforms these random variables into a new random variable w,

$$w = \Lambda_{i,j}[\mathbf{z}_j(i)], \quad w \in \mathbb{R} \tag{5.36}$$

While in [62] it was possible to get a closed-form expression for the pdf of w when using the LLR for a single path ML-PMHT, here we cannot. The sum of exponentials that arises from the multiple paths prevents us from inverting

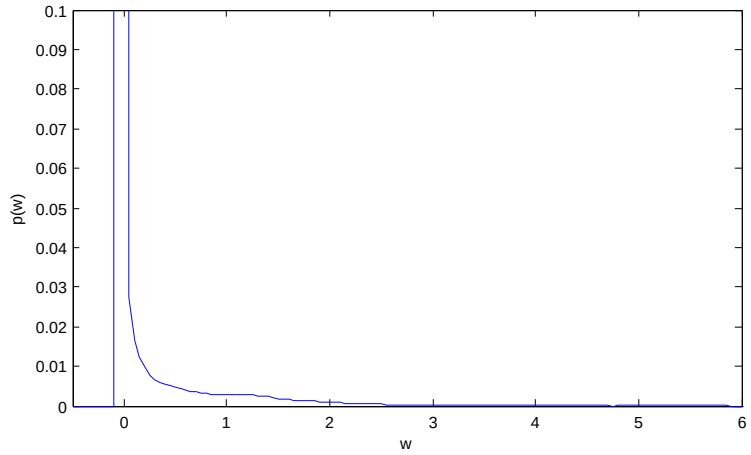


Figure 5.9: The pdf of w (a single *clutter* measurement transformed by the multipath LLR).

equation (5.35). We must instead rely on a numerical or empirical approximation of the pdf of w. This empirical pdf of w is shown in Figure 5.9.

We take the pdf of w and convolve it with itself $N-1$ times to find the pdf for a batch of N measurements from clutter. We refer to this resulting pdf as the "batch" pdf; it is the LLR pdf for a batch of measurements. We use $N = 900$, the expected number of measurements from clutter in one batch of measurements in our scenario. Again, following the methodology of [62], we must use the batch pdf to determine the "peak" pdf; this is the pdf of the maximum sample value from M samples from the batch pdf. This peak pdf is determined from extreme value theory. The determination of M is discussed in [62]; we use $M = 10^7$. The batch and peak pdfs, along with thresholds for several values of P_{FT} are shown in Figure 5.10.

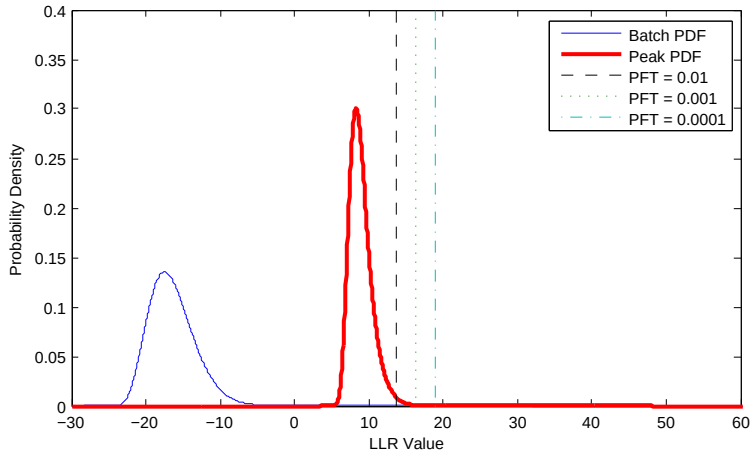


Figure 5.10: The batch and peak pdfs (from *clutter*) along with thresholds for several values of P_{FT}.

5.7.2 Probability of Target Track Detection

Now that we have calculated thresholds using the desired values for P_{FT}, we can use a similar procedure to evaluate P_{DT} for these thresholds using the methods in [63]. We again begin with the multipath LLR for a single measurement given by equation (5.35), but with $\mathbf{z}_j(i)$ as a Gaussian mixture (for the four paths) random variable (originating from the target) instead of a uniformly distributed random variable (originating from clutter). The approximation of the pdf of a single target measurement transformed by the multipath LLR pdf is shown in Figure 5.11.

We convolve this pdf with itself $N-1$ times to find the pdf for a batch of N target originated measurements. We use $N = 20$, the expected number of target originated measurements in one batch of $N_w = 15$ scans in our scenario. We do not need to find a peak pdf from this batch pdf; the batch pdf is the peak pdf in

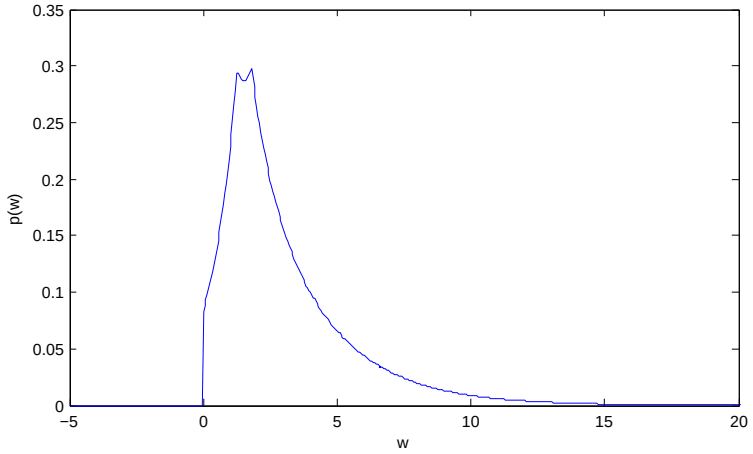


Figure 5.11: The pdf of w (a single *target* measurement transformed by the multipath LLR).

this case. The batch pdf and the thresholds calculated in Section 5.7.1 for several values of P_{FT} are shown in Figure 5.12. A P_{FT} of 10^{-4} yields $(1-P_{DT}) = 6\cdot 10^{-9}$.

5.8 Conclusions

We have developed an extension to the single target ML-PMHT to allow for the fusion of data from multiple signal propagation paths. We applied this algorithm to an OTHR scenario. We showed that, with low target SNR even down to 4 dB post-signal processing, the fusion ML-PMHT has excellent track detection and accuracy in such a scenario and is statistically efficient. Consequently, the ML-PMHT holds great promise in increasing the sensitivity and robustness of the next generation OTHR.

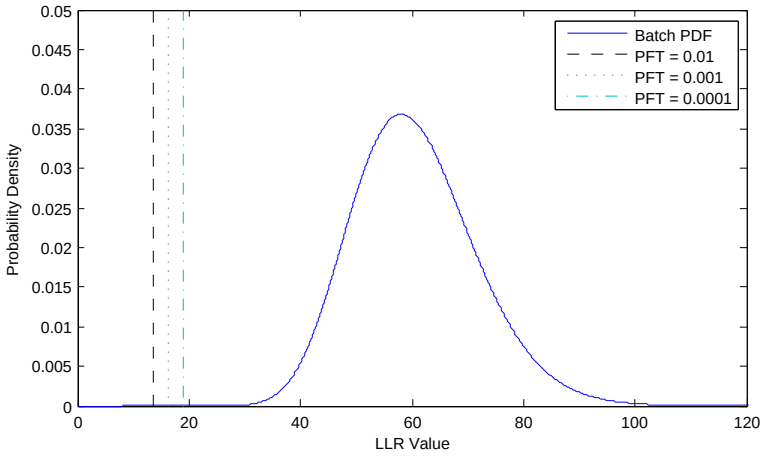


Figure 5.12: The batch and peak pdfs (from the *target*) along with thresholds for several values of P_{FT}.

The results indicate that the ML-PMHT can yield very high P_{DT} (probability of track detection) and very low P_{FT} (probability of false track). Future work would include using a more accurate ionosphere model.

Chapter 6

Detecting Low SNR Tracks with OTHR Using a Refraction Model

6.1 Introduction

Over-the-horizon radar (OTHR) [25] depends on signal refraction in the ionosphere to detect targets past the earth's curve where no direct line of sight is possible. Due to the properties of the ionosphere, the signal from the radar propagates via multiple paths, resulting in several target-originated detections, causing an ambiguity between detections and paths. The path that corresponds to each target detection is not known and there are also measurements from false detections.

There are many approaches to the OTHR problem, varying in how detection, tracking, and association are handled. These approaches include applying the probabilistic data association filter (PDAF) [14–16], the multipath probabilistic

data association algorithm (MPDA) [47], and the Probabilistic Multi-Hypothesis Tracker (PMHT) [16] to OTHR data. An extension of the PDAF called the Multiple Model Unified PDAF (MM-UPDAF) is developed in [14]. The MM-UPDAF is designed to handle multiple nonuniform clutter regions.

The multiple detection multiple hypothesis tracker (MD-MHT) [61] is formulated to solve the data association problem between measurements and measurement paths using an extended multiframe assignment technique. Alternatively, a multihypothesis fusion algorithm, presented in [45, 58–60], is a measurement-level fusion algorithm using only measurements already associated with targets by another filter to calculate the probabilities of association hypotheses. In [1, 2] a method is proposed for joint multiple target ground track estimation and slant track association. In the ML-PMHT data association is implicit and exact, and does not suffer from a combinatorial explosion with increasing scans or paths.

A multipath Expectation Maximization (EM) algorithm is developed and applied to an OTHR scenario in [35]. It is similar to the PMHT in that it treats data association as missing data. However, the multipath EM algorithm also treats the propagation paths as missing data. The (single path) ML-PMHT uses the log-likelihood function based on the PMHT model.

The lowest SNR in [47] and [61] is around 10 dB, with an ideal reflection ionosphere model. We show that the Maximum Likelihood Probabilistic Multi-Hypothesis Tracker (ML-PMHT), with a low target SNR of 4 dB and a refraction

model, yields a high track detection probability and a very low false track rate for the scenarios considered.

The ML-PMHT has previously been formulated for single and multitarget [64, 65] scenarios. It has been shown to perform well even with very low target SNR. We present the generalized form of the ML-PMHT that accounts for multiple possible propagation paths. We apply this algorithm to several OTHR scenarios. The multipath ML-PMHT considers simultaneously all the measurements without knowing their origins or propagation paths and, remarkably, has *linear complexity in the number of measurements*. The multipath ML-PMHT performs fusion of the multipath data in the presence of false measurements. Previously in [52, 53] we showed that the ML-PMHT has excellent track detection and accuracy in OTHR scenarios using an ideal reflection model for the ionosphere. As in [54], here we use a realistic refraction model relying on three-dimensional ray tracing. Further, we develop the Cramér-Rao lower bound and track detection probabilities using this model.

Section 6.2 briefly describes the ML-PMHT for a single target case and the generalization to allow for multiple paths. Section 6.3 presents the OTHR model used for simulation. Section 6.4 describes the three scenarios we consider: a surface target, a constant altitude target, and a constant vertical acceleration target. Section 6.5 discusses the performance of the multipath ML-PMHT from Monte Carlo testing. Section 6.6 presents the Cramér-Rao lower bound derivation. In

Section 6.7 we determine the track detection probability, P_{DT}, for values of the probability of false track, P_{FT}.

6.2 ML-PMHT

6.2.1 Single Target ML-PMHT

The ML-PMHT log-likelihood ratio (LLR) for the motion parameters of a single target is developed in [64]. This LLR is given by

$$\Lambda(\mathbf{x}; Z) \triangleq \ln \left\{ \frac{p(Z|\mathbf{x})}{p(Z|\text{all false})} \right\} = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \{\pi_0 + \pi_1 V p[\mathbf{z}_j(i)|\mathbf{x}(i)]\rho_j(i)\} \quad (6.1)$$

with

$$Z \triangleq \{\{\mathbf{z}_j(i)\}_{j=1}^{m_i}\}_{i=1}^{N_w} \quad (6.2)$$

Here N_w is the number of scans in the batch (the window length), and m_i is the number of measurements in the i^{th} scan (frame). The parameter vector $\mathbf{x}$ determines the target state $\mathbf{x}(i)$ in a deterministic way, i.e., a constant velocity or constant acceleration model. The prior probabilities that a measurement occurred due to clutter (false alarm) or due to a target are given by π_0 and π_1, respectively. These values are related to the probability of detection, P_D, and the probability of false alarm in a resolution cell, P_{FA}. The volume of the search region is V and a measurement, which did not occur due to a target, has a uniform pdf in V. The j^{th} measurement in the i^{th} scan is $\mathbf{z}_j(i)$ and its associated amplitude likelihood ratio is $\rho_j(i)$ given in 6.12. Finally, $p[\mathbf{z}_j(i)|\mathbf{x}(i)]$ is a Gaussian with mean determined by the target state parametrization extrapolated to time i,

$\mathbf{x}(i)$, and incorporating naturally the measurement noise covariance matrix. The amplitude likelihood ratio serves as a feature discriminant between the target originated measurements and the false ones due to spurious detections.

The pdfs $p[Z(i)|\mathbf{x}(i)]$ (likelihood of the target present hypothesis) and $p[Z(i)|\text{all false}]$ (likelihood of the target absent hypothesis) are derived using the ML-PMHT assumptions [64]:

- There is a single target with known probability of detection. (This is easy to relax under PMHT assumption; the algorithm is linear in the number of targets.)
- Any number of measurements in a scan can be assigned to the target.
- The motion of the target is deterministic.
- False detections are uniformly distributed.
- The number of false detections is Poisson distributed with known spatial density.
- Amplitudes of target and false detections are Rayleigh distributed with known distribution.
- Target measurements are corrupted with zero-mean Gaussian noise.
- Measurements at different times, conditioned on the parameterized state, are independent.

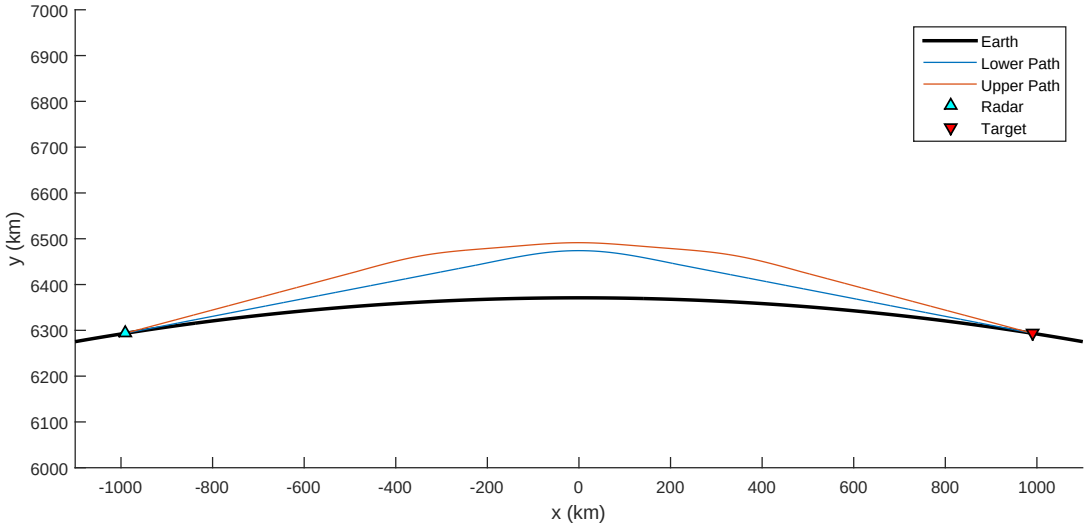


Figure 6.1: The two possible rays between the radar and the target drawn in the plane of the great circle connecting them.

These likelihoods are then given by

$$p[Z(i)|\mathbf{x}(i)] = \prod_{j=1}^{m_i} \left\{ \frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 p[\mathbf{z}_j(i)|\mathbf{x}(i)] p_1^\tau[a_j(i)] \right\} \tag{6.3}$$

$$p[Z(i)|\text{all false}] = \prod_{j=1}^{m_i} \frac{1}{V} p_0^\tau[a_j(i)] \tag{6.4}$$

where $p_0^\tau[a_j(i)]$ and $p_1^\tau[a_j(i)]$ are the pdfs of a false alarm and target measurement amplitude conditioned on exceeding the threshold τ, respectively.

6.2.2 The Multipath ML-PMHT Log-Likelihood Ratio for OTHR

The LLR of the generalized ML-PMHT that allows multiple propagation paths is given by

$$\Lambda(\mathbf{x}; Z) = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \ln \left\{ \pi_0 + \pi_1 V \rho_j(i) \sum_{\ell=1}^{n_p} p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell] P[\ell] \right\} \tag{6.5}$$

where ℓ is used to denote which path the signal took, $P[\ell]$ is the probability of path ℓ being taken, and n_p is the total number of possible paths. The mean of the Gaussian $p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell]$ is $f_\ell(\mathbf{x}(i))$, where f_ℓ is the function that transforms the target state $\mathbf{x}(i)$ into the measurement space via path ℓ. The covariance matrix for this Gaussian is the measurement noise covariance for a measurement from path ℓ. Note that, for simplicity, we have assumed that $\rho_j(i)$ is the same for each path ℓ (a path dependent LLR can be used if available).

6.3 OTHR Model

We investigate several three-dimensional OTHR scenarios using a spherical earth model. We use a refraction model (ray tracing) for the ionosphere. In this model the signal may propagate through the ionosphere in multiple (up to four) round-trip propagation paths. The radar measures slant range, slant range rate, amplitude, and azimuth. We assume additive zero-mean Gaussian noise for the slant range, slant range rate, and azimuth measurements with variances σ_r, $\sigma_{\dot{r}}$, and σ_θ, respectively.

6.3.1 Measurement Amplitudes

We model the amplitudes of the measurements according to a Swerling I model [6]. The amplitude is Rayleigh distributed with pdfs

$$p_0(a) = ae^{-\frac{a^2}{2}} \qquad a \geq 0 \tag{6.6}$$

$$p_1(a) = \frac{a}{1+d}e^{-\frac{a^2}{2(1+d)}} \qquad a \geq 0 \tag{6.7}$$

for the noise only and target, respectively. Here d is the expected SNR of the target in a resolution cell. For a chosen threshold τ we have

$$P_D = \int_\tau^\infty p_1(a)da \tag{6.8}$$

$$P_{FA} = \int_\tau^\infty p_0(a)da \tag{6.9}$$

The pdfs of the amplitude of a measurement given that it has exceeded the threshold τ are

$$p_0^\tau(a) = \frac{1}{P_{FA}}p_0(a) \qquad a \geq \tau \tag{6.10}$$

$$p_1^\tau(a) = \frac{1}{P_D}p_1(a) \qquad a \geq \tau \tag{6.11}$$

and the amplitude likelihood ratio is then

$$\rho_j(i) = \frac{p_1^\tau[a_j(i)]}{p_0^\tau[a_j(i)]} \tag{6.12}$$

6.3.2 Ray Tracing Algorithm

We use a openly available tool called IONORT [4] to perform the three-dimensional ray tracing in our simulations. It is based on the algorithm by Jones

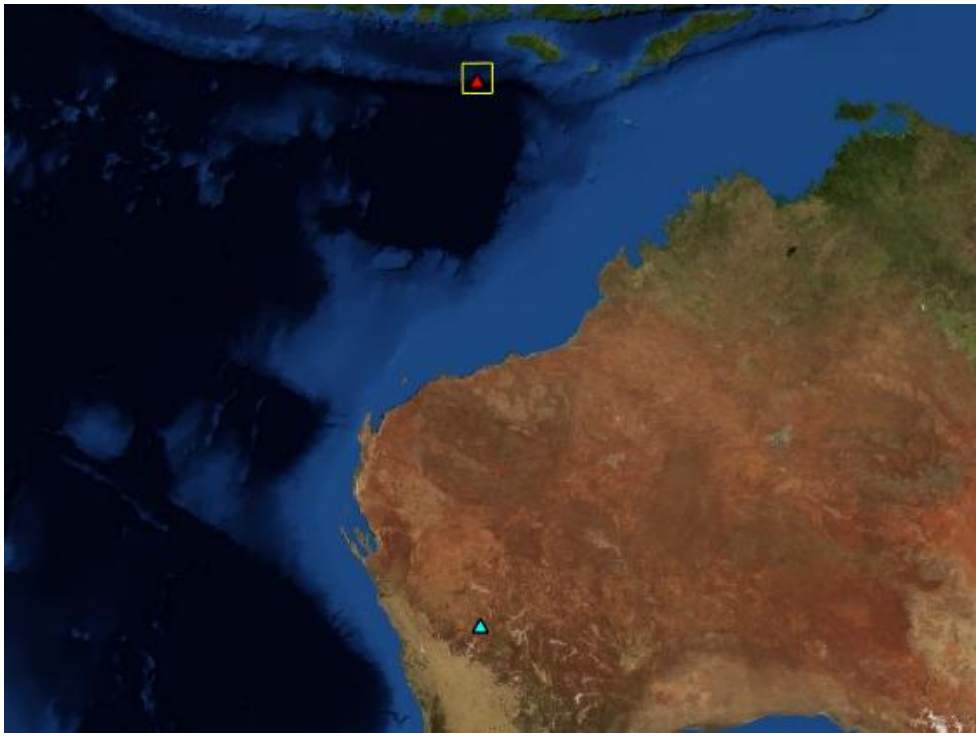


Figure 6.2: A notional satellite view of the location of the radar (blue), target (red), and the region we search for the target (yellow box, 1° wide in each of latitude and longitude).

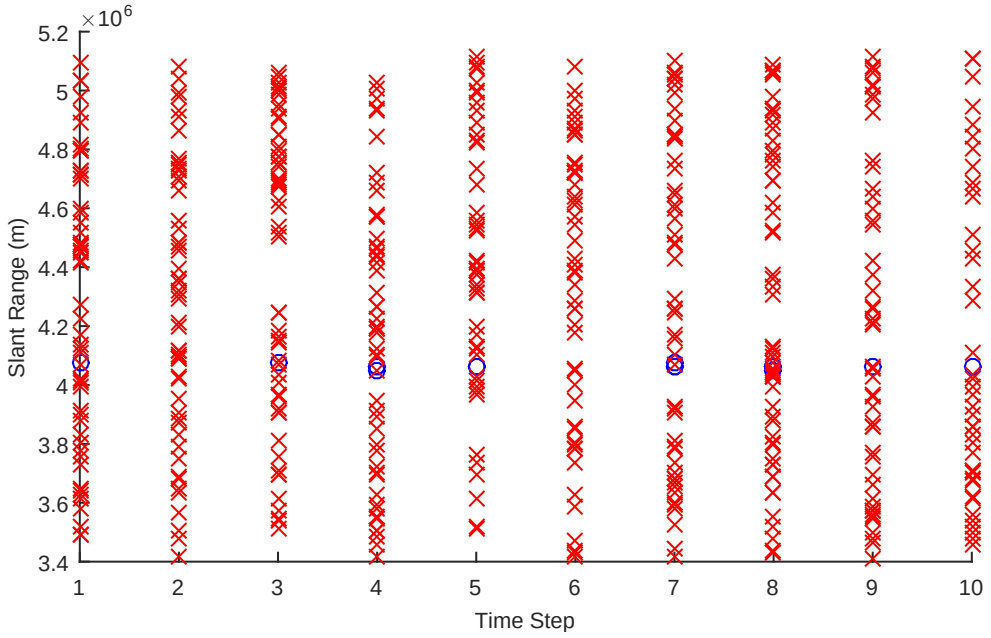


Figure 6.3: Slant range measurements in one batch (4dB post-signal processing SNR). Blue circles are target originated measurements, while red x's are false alarms.

and Stephenson [33], and uses a 3-D ionospheric regional model derived from the International Reference Ionosphere [9]. For all our simulations we choose an operating frequency of 15 MHz. An example of the two possible ray paths using this algorithm is shown in Figure 6.1.

6.4 Simulated Scenarios

6.4.1 Surface Target

We simulated a surface target with an initial position 2000 km away from the radar (see Figure 6.2), moving with a constant speed of 20 m/s and an initial course of 180°. The target follows the great circle starting from this initial state. The other values used in the simulations are given in Table 6.1. Figure 6.3 shows

Time between scans	2 s
σ_r	50 m
$\sigma_{\dot{r}}$	1 m/s
σ_θ	1°
SNR in a cell	**2.5 = 4 dB**
Amplitude detection threshold τ	2.5
Earth's radius	6371 km
$P[\ell]$ for all ℓ	0.25
Number of cells	1095
Search region	Lat: [−11.35°, −10.35°], Lon: [117.5°, 118.5°]
P_D for each path	0.4
P_{FA} in a cell	0.044
Expected number of false alarms per scan	48
π_0	0.9671

Table 6.1: Scenario parameters used in all simulations.

the measurements used (after amplitude thresholding) in one run of the tracker. False measurements are generated uniformly in the measurement space. Note that, due to the very low SNR in a cell, P_D is 0.4 and the high P_{FA} leads to 48 false measurements per scan as seen in Figures 6.4-6.6. Also note that there are usually zero to three target originated measurements in each scan (rarely all four) and the overwhelming number of false measurements, which, however, can be successfully handled by the multipath ML-PMHT track detector.

6.4.2 Constant Altitude Target

We simulated a target with an initial position 2000 km away from the radar, moving with a constant speed of 200 m/s, an initial course of 180°, and with a constant altitude of 6 km. The other values used in the simulations are given in Table 6.1. The measurements look similar to those in Figures 6.4-6.6.

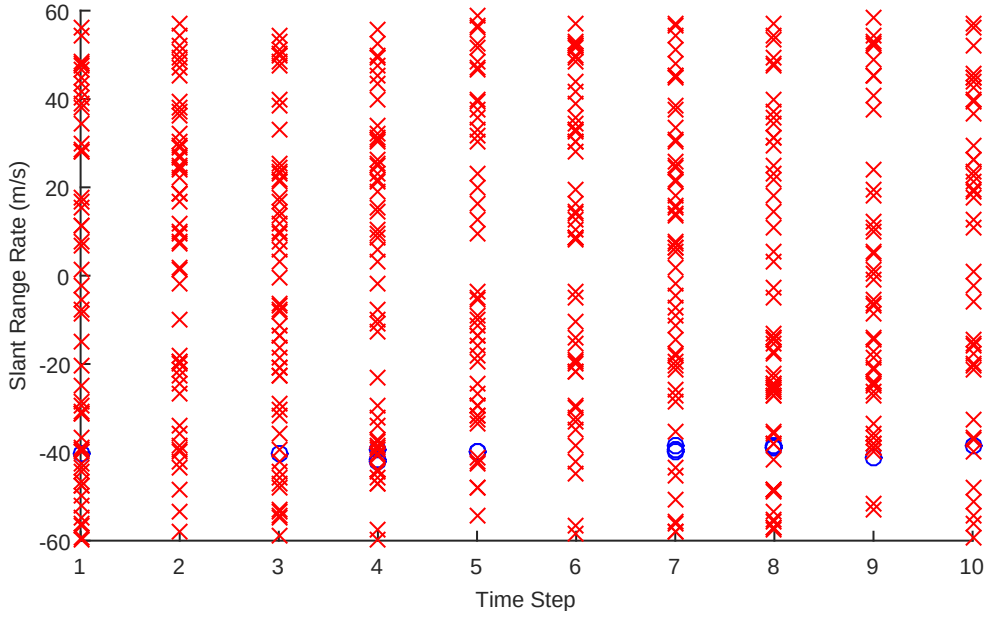


Figure 6.4: Slant range rate measurements for the surface target scenario in one batch (4dB post-signal processing SNR). Blue circles are target originated measurements, while red x's are false alarms.

6.4.3 Constant Vertical Acceleration Target

We simulated a target with an initial position 2000 km away from the radar, with an initial velocity and altitude of 0, and a constant vertical acceleration of $20\,\mathrm{m/s^2}$. The other values used in the simulations are given in Table 6.1. The measurements look similar to those shown in Figures 6.4-6.6.

6.5 Performance of the Track Detector

The LLR of the multipath ML-PMHT for a single run of the surface target scenario is shown in Figures 6.7-6.10. Figure 6.7 shows there are five peaks resulting from path ambiguity. The central peak (the correct one), however,

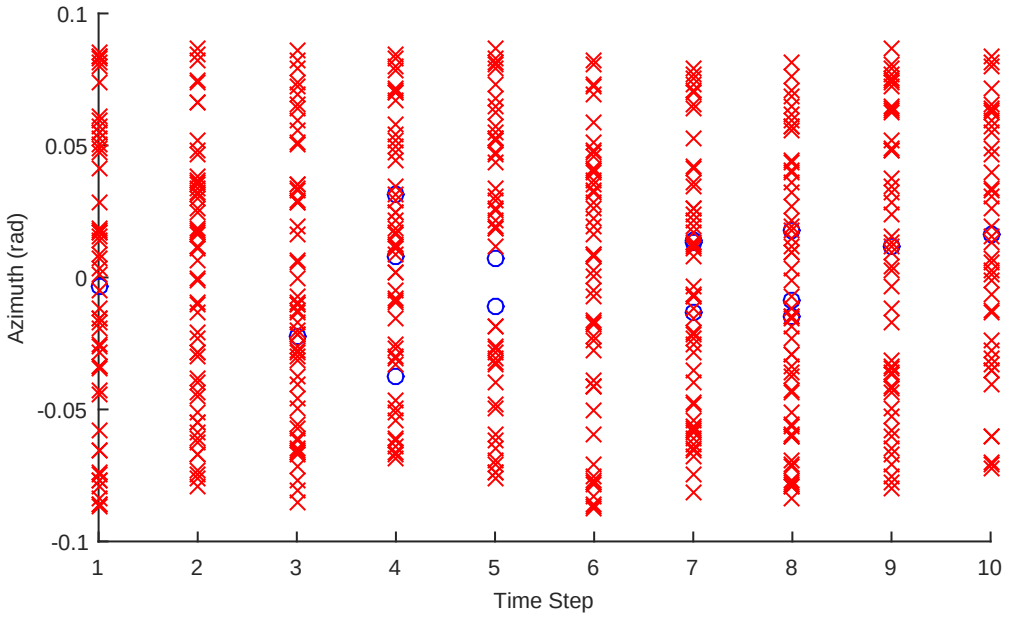


Figure 6.5: Azimuth measurements for the surface target scenario in one batch (4dB post-signal processing SNR). Blue circles are target originated measurements, while red x's are false alarms.

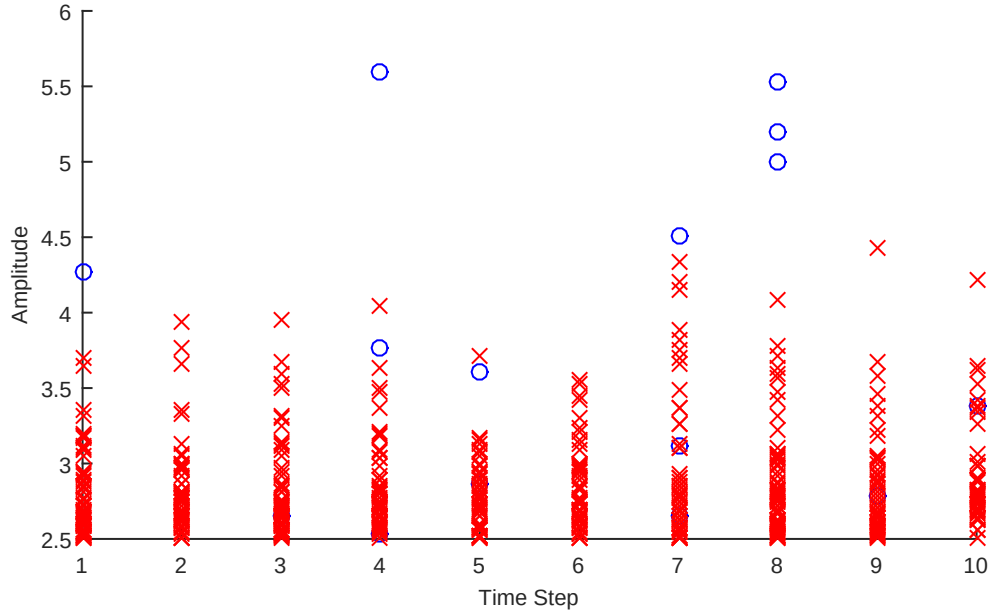


Figure 6.6: The amplitude of the measurements for the surface target scenario in one batch (4dB post-signal processing SNR). Blue circles are target originated measurements, while red x's are false alarms.

is easily distinguishable from the side peaks. It is also much higher than any peak occurring due to clutter. Figure 6.11 shows the LLR surface for a batch containing *no target*, i.e., false alarms only. The small peaks resulting from clutter indicate that a very high probability of track detection is possible with a very low probability of false track (this is further shown in Section 6.7).

We use the Controlled Random Search with local mutation algorithm [34, 46] to perform global optimization using the implementation found in the NLopt library [32]. We then run a local optimization routine from MATLAB using an interior-point algorithm on the highest valued point from the global search to produce the final state estimate. Each run of the search procedure is stopped after 15 seconds (i.e., faster than real time). The RMS errors (at the final time of the batch) and the corresponding CRLB for the simulations are given in Tables 6.2, 6.3, and 6.4. There were no false tracks or missed tracks in any scenario.

	RMSE ($N_w = 20$)	CRLB ($N_w = 20$)	RMSE ($N_w = 30$)	CRLB ($N_w = 30$)
Position	5988 m	6012 m	4925 m	4909 m
Range	4.94 m	4.33 m	3.75 m	3.54 m
Speed	5.99 m/s	1.35 m/s	3.60 m/s	1.09 m/s
Course	30°	20°	23°	16°

Table 6.2: RMSE from 100 Monte Carlo runs of the surface target scenario and the corresponding CRLB.

	RMSE ($N_w = 20$)	CRLB ($N_w = 20$)	RMSE ($N_w = 30$)	CRLB ($N_w = 30$)
Position	5881 m	6012 m	5261 m	4909 m
Range	14.84 m	14.34 m	11.07 m	11.71 m
Altitude	91.93 m	90.05 m	74.96 m	73.53 m
Speed	2.23 m/s	0.65 m/s	1.02 m/s	0.51,m/s
Course	5.3°	2.5°	2.4°	2°

Table 6.3: RMSE from 100 Monte Carlo runs of the constant altitude target scenario and the corresponding CRLB.

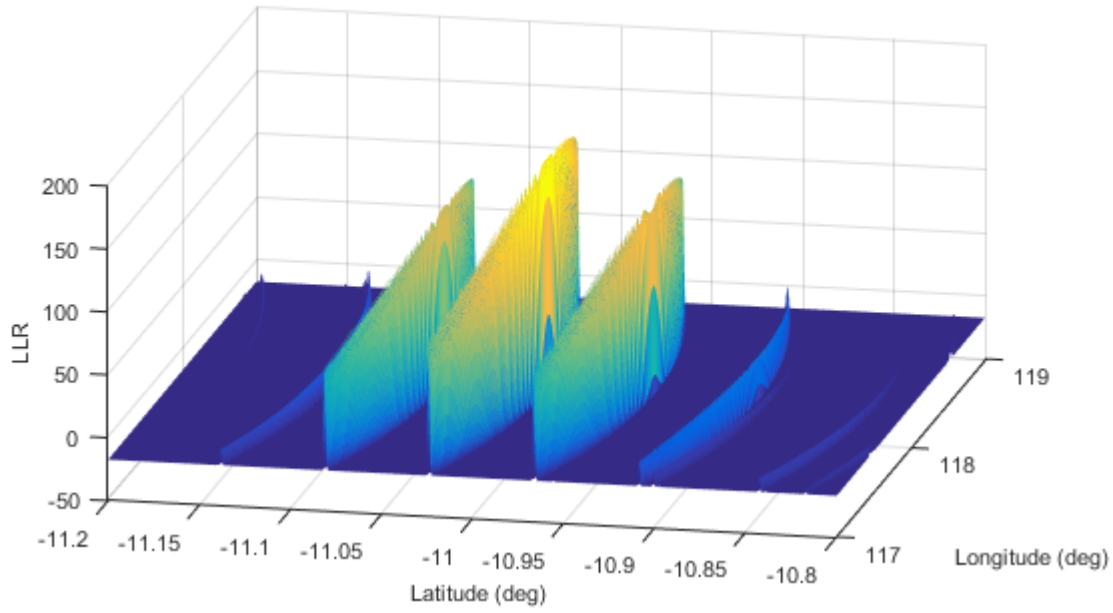


Figure 6.7: The log-likelihood ratio at the true values for course and speed in the surface target scenario.

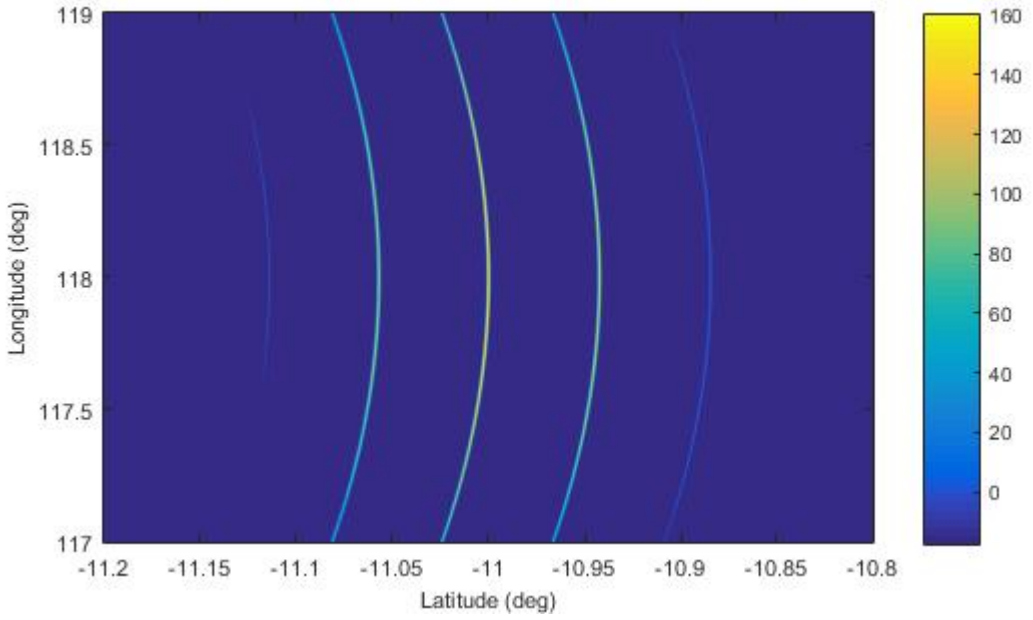


Figure 6.8: The log-likelihood ratio surface at the true values for course and speed in the surface target scenario (4dB post-signal processing SNR).

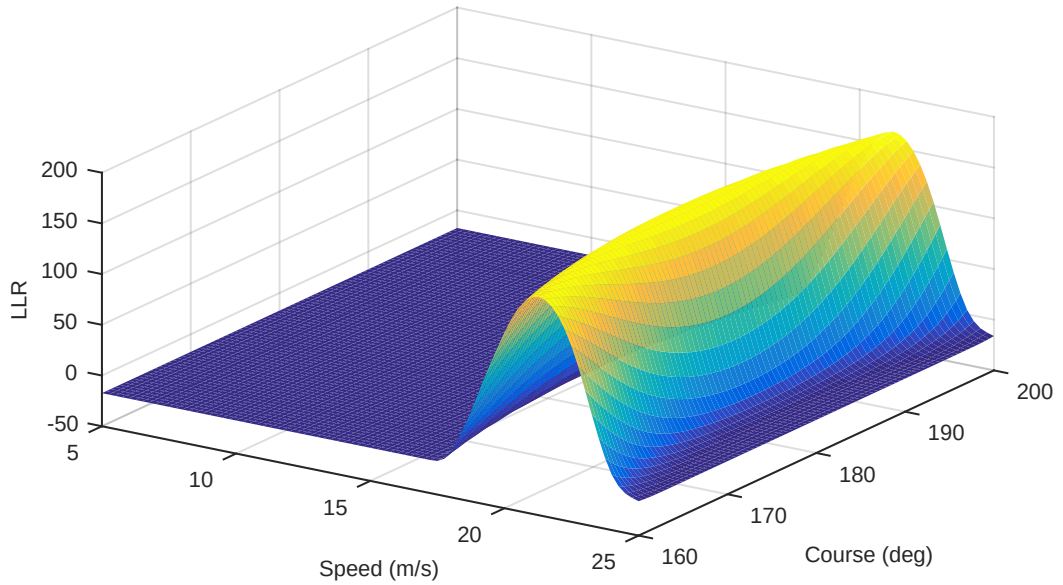


Figure 6.9: The log-likelihood ratio surface at the true values for latitude and longitude in the surface target scenario (the large crossrange errors dominate here).

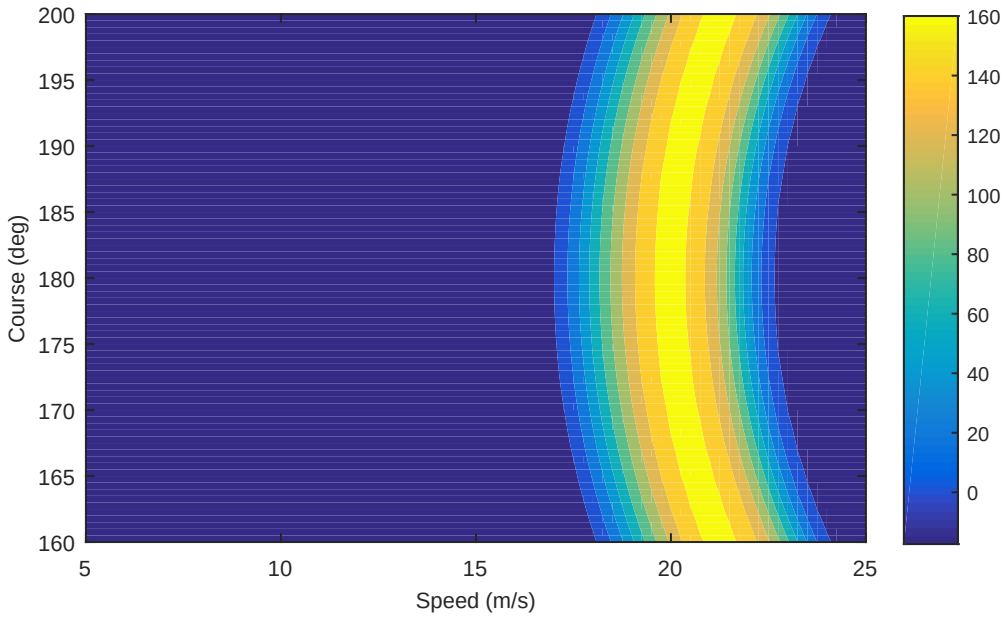


Figure 6.10: The log-likelihood ratio surface at the true values for latitude and longitude in the surface target scenario (the large crossrange errors dominate here).

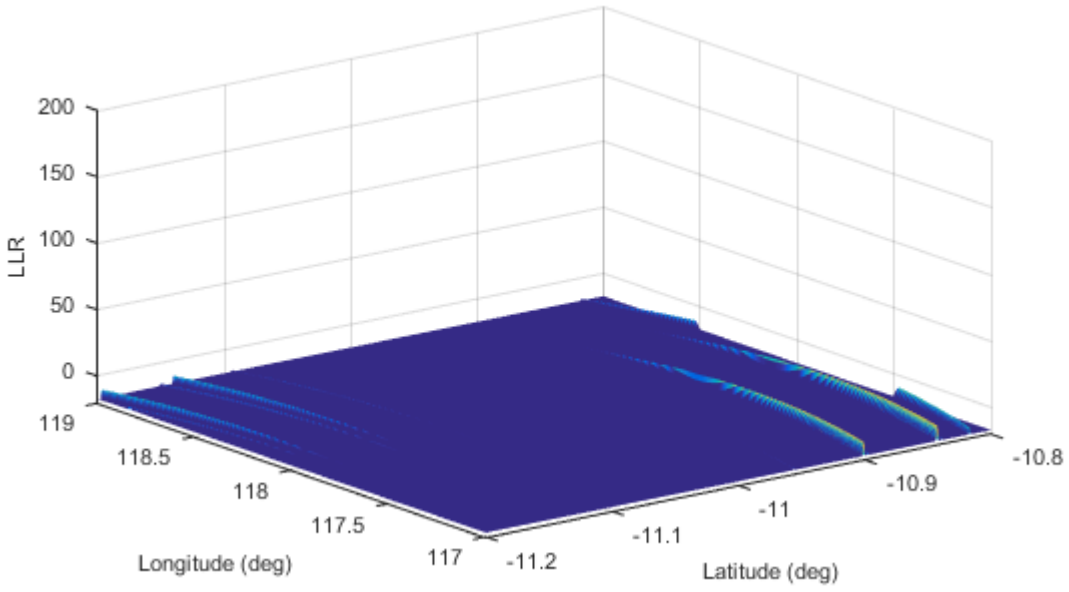


Figure 6.11: The log-likelihood ratio surface for *clutter only*.

	RMSE ($N_w = 10$)	CRLB ($N_w = 10$)	RMSE ($N_w = 20$)	CRLB ($N_w = 20$)
Position	8759 m	8503 m	5952 m	6012 m
Range	6.86 m	6.54 m	4.56 m	4.33 m
Acceleration	0.15 m/s^2	0.093 m/s^2	0.10 m/s^2	0.066 m/s^2

Table 6.4: RMSE from 100 Monte Carlo runs of the constant vertical acceleration target scenario and the corresponding CRLB.

6.6 Multipath Fusion ML-PMHT Cramér-Rao Lower Bound

We develop the Cramér-Rao Lower Bound (CRLB) [7] for the multipath fusion ML-PMHT. We can assume all scans to be independent and also assume the measurements in each scan to be independent. The Fisher Information Matrix (FIM) $\mathbf{J}$ will then be the sum of the FIM's $\mathbf{J}_{i,j}$ of each measurement,

$$\mathbf{J} = \mathrm{E}\{(\nabla_{\mathbf{x}} \ln p[\mathbf{Z}|\mathbf{x}])(\nabla_{\mathbf{x}} \ln p[\mathbf{Z}|\mathbf{x}])^T\}|_{\mathbf{x}=\mathbf{x_0}} = \sum_{i=1}^{N_w} \sum_{j=1}^{m_i} \mathbf{J}_{i,j} \tag{6.13}$$

where

$$\mathbf{J}_{i,j} = \mathrm{E}\{(\nabla_{\mathbf{x}(i)} \ln p[\mathbf{z}_j(i)|\mathbf{x}(i)])(\nabla_{\mathbf{x}(i)} \ln p[\mathbf{z}_j(i)|\mathbf{x}(i)])^T\}|_{\mathbf{x}(i)=\mathbf{x_0}(i)} \tag{6.14}$$

and $\mathbf{x_0}(i)$ is the true value of $\mathbf{x}(i)$. The state vectors $\mathbf{x}(i)$ in the three scenarios considered are given by

$$\mathbf{x}(i) = [\psi(i),\, \lambda(i),\, \theta(i),\, s]' \tag{6.15}$$

$$\mathbf{x}(i) = [\psi(i),\, \lambda(i),\, \theta(i),\, s,\, a]' \tag{6.16}$$

$$\mathbf{x}(i) = [\psi(i),\, \lambda(i),\, \alpha]' \tag{6.17}$$

for the surface target, constant altitude target, and constant acceleration target, respectively. The components of the state vectors are: latitude $\psi(i)$, longitude $\lambda(i)$, course $\theta(i)$, speed s, altitude a, and acceleration α.

The multipath ML-PMHT likelihood for a single measurement is given by (6.18). The gradient of the logarithm of this likelihood gives (6.19), where $\mathbf{D}_\ell(i)$ is the Jacobian of $f_\ell(\mathbf{x}(i))$. Finally, combining equations (6.14) and (6.19) gives us the FIM of one measurement, which has to be evaluated numerically, as (6.20). The calculated CRLB values are shown in Tables 6.2, 6.3, and 6.4.

$$p[\mathbf{z}_j(i)|\mathbf{x}(i)] = \frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell] P(\ell) \tag{6.18}$$

$$\nabla_{\mathbf{x}(i)} \ln p(\mathbf{z}_j(i)|\mathbf{x}(i)) =$$

$$\frac{\pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} P(\ell)|2\pi\mathbf{R}|^{-\frac{1}{2}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]} \mathbf{D}_\ell^T(i)\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]}{\frac{\pi_0}{V} p_0^\tau[a_j(i)] + \pi_1 p_1^\tau[a_j(i)] \sum_{\ell=1}^{n_p} P(\ell)|2\pi\mathbf{R}|^{-\frac{1}{2}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]}} \tag{6.19}$$

$$\mathbf{J}_{i,j} = \int_{\tau}^{\infty} \iint_{V} \frac{\frac{(\pi_1 p_1^{\tau}[a_j(i)])^2}{|2\pi\mathbf{R}|} \sum_{\ell=1}^{n_p} \mathbf{A}_{\ell}(i) \sum_{\ell=1}^{n_p} \mathbf{A}_{\ell}^{T}(i)}{\frac{\pi_0}{V} p_0^{\tau}[a_j(i)] + \pi_1 p_1^{\tau}[a_j(i)] \sum_{\ell=1}^{n_p} P(\ell)|2\pi\mathbf{R}|^{-\frac{1}{2}} e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]}} d\mathbf{z}_j(i) da_j(i) \tag{6.20}$$

$$\mathbf{A}_{\ell}(i) = P(\ell) e^{-\frac{1}{2}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]'\mathbf{R}^{-1}[\mathbf{z}_j(i)-f_\ell(\mathbf{x}(i))]} \mathbf{D}_{\ell}^{T}(i) \mathbf{R}^{-1}[\mathbf{z}_j(i) - f_\ell(\mathbf{x}(i))] \tag{6.21}$$

6.7 False Track and Target Track Detection Probabilities

We use the methods in [62, 63] to determine a threshold for the probability of false track, P_{FT}, and then calculate the probability of track detection, P_{DT}.

6.7.1 Probability of False Track

We will treat the measurement $\mathbf{z}_j(i) \in \mathbb{R}^3$ and its corresponding amplitude LLR $\rho_j(i) \in \mathbb{R}^+$ as random variables in the multipath LLR for a single measurement,

$$\Lambda_{i,j}[\mathbf{z}_j(i)] = \ln \left\{ \pi_0 + \pi_1 V \rho_j(i) \sum_{\ell=1}^{n_p} p[\mathbf{z}_j(i)|\mathbf{x}(i), \ell] P[\ell] \right\} \tag{6.22}$$

Equation (6.22) is a function that transforms these random variables into a new random variable w,

$$w = \Lambda_{i,j}[\mathbf{z}_j(i)], \quad w \in \mathbb{R} \tag{6.23}$$

While in [62] it was possible to get a closed-form expression for the pdf of w when using the LLR for a single path ML-PMHT, here we cannot. This is due to the sum of exponentials that arises from the multiple paths which prevents us from inverting equation (6.22). We must use a numerical approximation of the pdf of w.

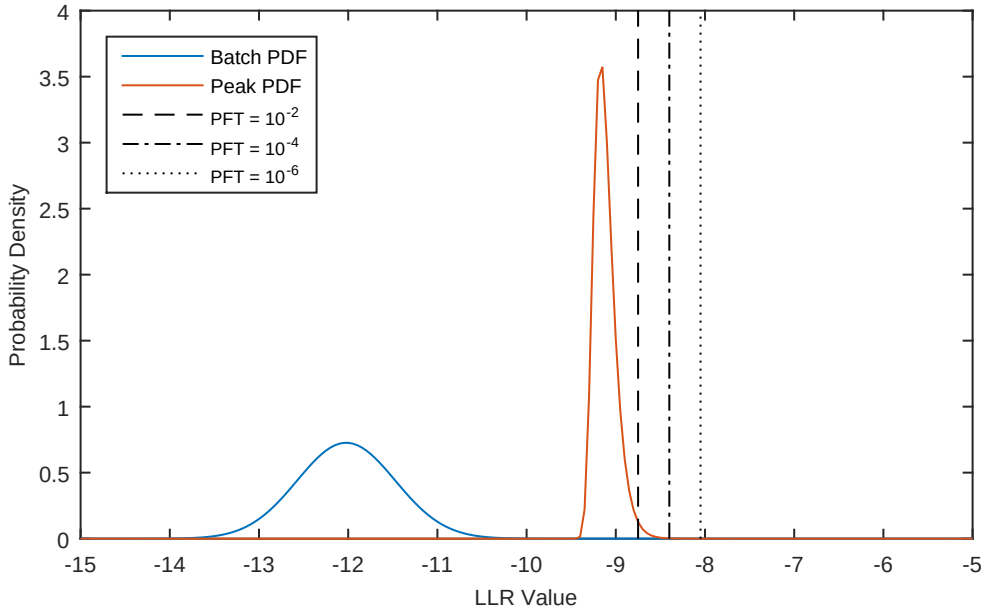


Figure 6.12: The batch and peak pdfs (from *clutter*) along with thresholds for several values of P_{FT}.

We take the pdf of w and convolve it with itself $N-1$ times to find the pdf for a batch of N measurements from clutter. We refer to this resulting pdf as the "batch" pdf; it is the LLR pdf for a batch of measurements. We use $N = 482$, the expected number of measurements from clutter in one batch of measurements in our scenario. Again, following the methodology of [62], we must use the batch pdf to determine the "peak" pdf; this is the pdf of the maximum sample value from M samples from the batch pdf. This peak pdf is determined from extreme value theory. The determination of M is discussed in [62]; we use $M = 10^7$. The batch and peak pdfs, along with thresholds for several values of P_{FT} are shown in Figure 6.12.

6.7.2 Probability of Target Track Detection

Now that we have calculated thresholds using the desired values for P_{FT}, we can use a similar procedure to evaluate P_{DT} for these thresholds using the methods in [63]. We again begin with the multipath LLR for a single measurement given by equation (6.22), but with $\mathbf{z}_j(i)$ as a Gaussian mixture (for the multiple paths) random variable (originating from the target) instead of a uniformly distributed random variable (originating from clutter).

We convolve this pdf with itself $N-1$ times to find the pdf for a batch of N target originated measurements. We use $N = 16$, the expected number of target originated measurements in one batch of $N_w = 10$ scans in our scenario. We do not need to find a peak pdf from this batch pdf; the batch pdf is the peak pdf in this case; this is shown in Figure 6.13. For a P_{FT} of 10^{-6}, we can see that $P_{DT} > 99\%$.

6.8 Conclusions

We applied the multipath ML-PMHT algorithm to three OTHR scenarios: a surface target, a constant altitude target, and a constant vertical acceleration target. This was accomplished using a three-dimensional ray tracing algorithm to simulate the signal propagation paths through the ionosphere. We showed that, with a very low target SNR of 4 dB post-signal processing, the ML-PMHT has excellent track detection (for 20 s batch length this corresponds to an expected number of false tracks per day of $5 \cdot 10^{-3}$) and accuracy in such scenarios. The

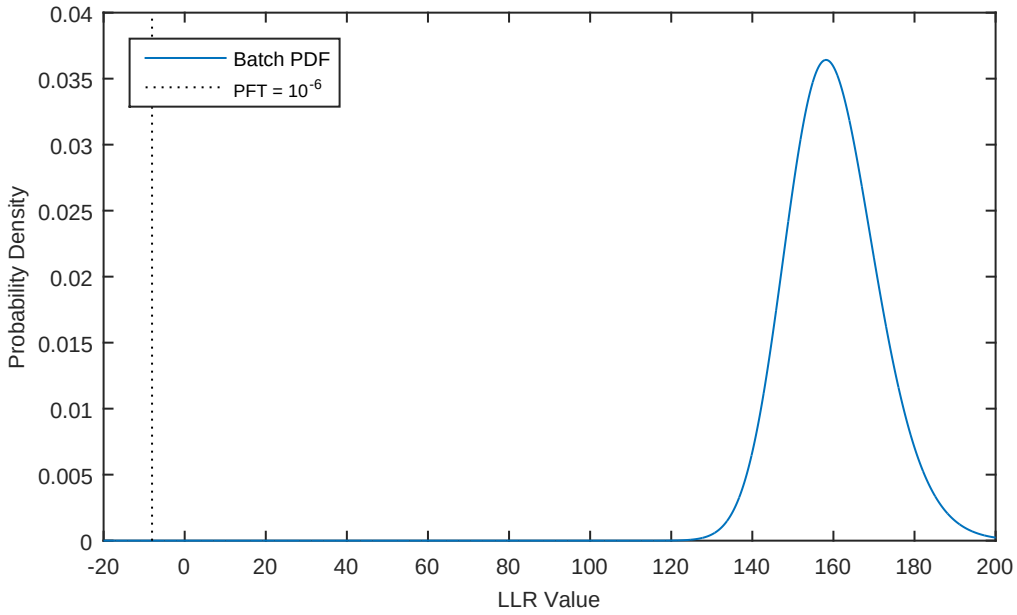


Figure 6.13: The batch pdfs (from the *target*) along with a threshold for a value of P_{FT}.

results indicate that the ML-PMHT can yield very high P_{DT} (probability of track detection) for very low P_{FT} (probability of false track) in such difficult scenarios.

Chapter 7

Conclusions

In this dissertation, we investigated three real-world tracking problems, develop tracking algorithms to solve them, and demonstrate their effectiveness through comparison with state-of-the-art algorithms. First, we developed a fast and coalescence-avoiding approximation to the JPDA for use in scenarios with closely spaced targets by combining the techniques of Roecker and Blom. Second, we used a regularized particle filter to solve the banana and contact lens problems using a multidimensional version of the Epanechnikov kernel for state vectors developed in the course of the research for this dissertation. Finally, using both ideal reflection and refraction models for the ionosphere, we generalized the ML-PMHT to allow for multiple measurement propagation paths and showed it effective in Over-The-Horizon scenarios for various kinds of targets.

Bibliography

[1] R. H. Anderson and J. L. Krolik, "Multipath track association for over-the-horizon radar using a bootstrapped statistical ionospheric model," *Conference Record of the Thirty-Third Asilomar Conference on Signals, Systems, and Computers*, vol. 1, pp. 8–14, 1999.

[2] R. H. Anderson and J. L. Krolik, "Track association for over-the-horizon radar with a statistical ionospheric model," *IEEE Transactions on Signal Processing*, vol. 50, no. 11, pp.2632–2643, 2002.

[3] Arulampalam, M.; Maskell, S.; Gordon, N.; and Clapp, T., "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, February 2002.

[4] A. Azzarone, C. Bianchi, M. Pezzopane, M. Pietrella, C. Scotto, and A. Settimi, "IONORT: A Windows software tool to calculate the HF ray tracing in the ionosphere," *Computers & Geosciences*, vol. 42, pp. 57–63, May 2012.

[5] B. Bakhtiar and H. Alavi, Efficient algorithm for computing data association probabilities for multitarget tracking, *Proceedings of SPIE: Automatic Object Recognition IV*, vol. 2756, Apr. 1996, pp. 130-140.

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

[7] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: Wiley, 2001.

[8] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion*. Storrs, CT: YBS Publishing, 2011.

[9] D. Bilitza and B. Reinisch, "International Reference Ionosphere 2007: Improvements and new parameters", *Journal of Advances in Space Research*, vol. 42, no, 4, pp. 599–609, 2008.

[10] S. S. Blackman and R. F. Popoli, *Design and Analysis of Modern Tracking Systems*. Norwood, MA: Artech House, 1999.

[11] W. R. Blanding, P. K. Willett, and Y. Bar-Shalom, "Offline and Real-Time Methods for ML-PDA Track Validation," *IEEE Transactions on Signal Processing*, vol. 55, no. 5, pp. 1994–2006, May 2007.

[12] H. A. P. Blom and E. A. Bloem, "Bayesian tracking of two possibly unresolved maneuvering targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 2, pp. 612-627, April 2007.

[13] —, "Probabilistic data association avoiding track coalescence," *IEEE Transactions on Automatic Control*, vol. 45, no. 2, pp. 247–259, February 2000.

[14] S. B. Colegrove and S. J. Davey, "PDAF with multiple clutter regions and target models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 1, pp. 110–124, January 2003.

[15] S. B. Colegrove and S. J. Davey, "The probabilistic data association filter with multiple nonuniform clutter regions," *The Record of the IEEE 2000 International Radar Conference, 2000*, pp. 65–70, 2000.

[16] S. B. Colegrove, S. J. Davey, and B. Cheung, "PDAF versus PMHT performance on OTHR data," *Proceedings of the International Radar Conference, 2003*, pp. 560–565, September 2003.

[17] D. F. Crouse, Y. Bar-Shalom, P. Willett, and L. Svensson, "JPDAF versions and implementations: Comparisons for closely-spaced targets," submitted for publication in *IEEE A&E Systems Magazine*, August 2010.

[18] —, "The JPDAF in practical systems: Computation and snake oil," in *Proceedings of SPIE Signal and Data Processing* of Small Targets, vol. 7698, April 2010.

[19] D. F. Crouse, M. Guerriero, P. Willett, R. Streit, and D. Dunham, "A look at the PMHT," in *12th International Conference on Information Fusion*, July 2009, pp. 332-339.

[20] F. Daum comments on C. F. Gauss, "Unstinken Methode der Kleinsten Quadrate," Free translation of various works by Gauss on least squares, Gesammelte Werke, Königliche Gesellschaft der Wissenschaften zu Göttingen, 2010.

[21] F. E. Daum and R. J. Fitzgerald, "Importance of resolution in multiple-target tracking," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 2235, 1994, pp. 329-338.

[22] F. Daum, J. Huang, and A. Noushin, "Exact particle flow for nonlinear filters," in *Proc. SPIE Signal Processing, Sensor Fusion, and Target Recognition*, Orlando, FL, April 2010.

[23] Z. Ding and B. Balaji, "Performance evaluation of four nonlinear filters for two radar applications," Defence Research and Development Canada, Tech. Rep. TM 2010-246, DRDC Ottawa, Ontario, December 2010.

[24] A. Doucet, and A. M. Johansen, *A tutorial on particle filtering and smoothing: Fifteen years later*. Oxford University Press, 2009.

[25] G. Fabrizio, *High Frequency Over-the-Horizon Radar: Fundamental Principles, Signal Processing, and Practical Applications*. McGraw-Hill, 2013.

[26] R. J. Fitzgerald, "Development of practical PDA logic for multitarget tracking by microprocessor," in *Multitarget- Multisensor Tracking: Advanced Applications*, Y. Bar-Shalom, Ed. Norwood, MA: Artech House, 1996, pp. 1-23.

[27] S. V. Fridman and L. J. Nickisch, "SIFTER: Signal inversion for target extraction and registration," *Radio Science*, vol. 39, no. 1, 2004.

[28] F. Gustafsson, "Particle Filter Theory and Practice with Positioning Applications," *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 7, pp. 53–82, July 2010.

[29] J. T. Horwood, N. D. Aragon, and A. B. Poore, "Gaussian sum filters for space surveillance: Theory and simulations," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 6, pp. 1839–1851, November–December 2011.

[30] J. T. Horwood, and A. B. Poore, "Adaptive Gaussian sum filters for space surveillance," *IEEE Transactions on Automatic Control*, vol. 56, no. 8, pp. 1777–1790, Aug. 2011.

[31] R. Jenssen, "Indefinite Parzen Window for Spectral Clustering," in *Proc. IEEE Workshop on Machine Learning for Signal Processing*, pp. 390–395, August 2007.

[32] S. G. Johnson, *The NLopt nonlinear-optimization package* [Online]. Available: http://ab-initio.mit.edu/nlopt

[33] R. M. Jones and J. J. Stephenson, "A versatile three-dimensional ray tracing computer program for radio waves in the ionosphere," U.S. Department of Commerce, OT Report, pp. 75–76, 1975.

[34] P. Kaelo and M. M. Ali, "Some variants of the controlled random search algorithm for global optimization," *J. Optim. Theory Appl.* vol. 130, no. 2, pp. 253–264, 2006.

[35] H. Lan, Y. Liang, Q. Pan, F. Yang, and C. Guan, "An EM Algorithm for Multipath State Estimation in OTHR Target Tracking," *IEEE Transactions on Signal Processing*, vol. 62, no. 11, pp. 2814–2826, 2014.

[36] S. Maskell, "An application of Sequential Monte Carlo samplers: An alternative to particle filters for non-linear non-Gaussian sequential inference with zero process noise," *Data Fusion Target Tracking Conference (DF TT 2012): Algorithms Applications, 9th IET*, May 2012.

[37] S. Maskell, M. Briers, and R. Wright, "Fast mutual exclusion," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 5428, no. 1, August 2004, pp. 526-536.

[38] S. Maskell, M. Briers, R. Wright, and P. Horridge, "Tracking using a radar and a problem specific proposal distribution in a particle filter," *IEE Proceedings on Radar Sonar Navigation*, vol. 152, no. 5, pp. 315–322, October 2005.

[39] D. Musicki, and R. J. Evans, "Measurement Gaussian sum mixture target tracking," *Proc. 9th International Conference on Information Fusion*, Seattle, WA, July 2009.

[40] C. Musso, N. Oudjane, and F. Le Gland, Improving regularized particle filters, in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, editors, *Statistics for Engineering and Information Science*, Springer-Verlag, New York, 2001, ch. 12, pages 247–271.

[41] L. Nyland, J. Prins, A. Goldberg, and P. Mills, "A design methodology for data-parallel applications," *IEEE Transactions on Software Engineering*, vol. 26, no. 4, pp. 293-314, April 2000.

[42] S. Oh, S. Russell and S. Sastry, "Markov chain monte carlo data association for multi-target tracking," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 481-497, Mar. 2009.

[43] S. Oh and S. Sastry, "A polynomial-time approximation algorithm for joint probabilistic data association," in *Proceedings of the American Control Conference*, vol. 2, 2005, pp. 1283-1288.

[44] R. D. Palkki, A. D. Lanterman, and W. D. Blair, "Addressing track hypothesis coalescence in sequential k-best multiple hypothesis tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 1551-1563, July 2011.

[45] D. J. Percival and K. A. B. White, "Multihypothesis fusion of multipath over-the-horizon radar tracks" *Proc. SPIE 3373, Signal and Data Processing of Small Targets*, 1998.

[46] W. L. Price, "Global optimization by controlled random search," *J. Optim. Theory Appl.*, vol. 40, no. 3, pp. 333–348, 1983.

[47] G. W. Pulford and R. J. Evans, "A Multipath Data Association Tracker for Over-the-Horizon Radar," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 4, pp. 1165–1183, October 1998.

[48] P. Quan, Z. Hongcai, W. Peide and H. Zhou, "Development of new practical multitarget data association algorithm," *Proceedings of the First IEEE Conference on Control Applications*, vol. 2, Dayton, OH, Sep. 1992, pp. 1065-1066.

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

[50] J. A. Roecker, "A class of near optimal JPDA algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 30, no. 2, pp. 504-510, April 1994.

[51] J. A. Roecker and G. L. Phillis, "Suboptimal joint probabilistic data association," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 29, no. 2, pp. 510-517, Apr. 1993.

[52] K. Romeo, Y. Bar-Shalom, and P. Willett, "Data fusion with ML-PMHT for very low SNR track detection in an OTHR," in *Proceedings of the 18th International Conference on Information Fusion*, Washington, D.C., July 2015.

[53] K. Romeo, Y. Bar-Shalom, and P. Willett, "Fusion of multipath data with ML-PMHT for very low SNR track detection in an OTHR," *Journal of Advances in Information Fusion*, Accepted for publication, 2015.

[54] K. Romeo, Y. Bar-Shalom, and P. Willett, "Low SNR Track Detection with OTHR Based on a Refraction Model," in *Proceedings of the 2016 IEEE Radar Conference*, Philadelphia, PA, May 2016.

[55] K. Romeo, D. F. Crouse, Y. Bar-Shalom, and P. Willett, "The JPDAF in practical systems: Approximations," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 7698, Orlando, FL, April 2010.

[56] , "A fast coalescence-avoiding JPDAF," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 8393, Baltimore, MD, April 2012.

[57] K. Romeo, P. Willett, and Y. Bar-Shalom, "Particle filter tracking for long range radars," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, #8393-32, Baltimore, MD, April 2012.

[58] M. G. Rutten, S. Maskell, M. Briers, and N. J. Gordon, "Multipath track association for over-the-horizon radar using Lagrangian relaxation," *Proc. SPIE Signal and Data Processing of Small Targets,* 2004.

[59] M. G. Rutten and D. J. Percival, "Comparison of track fusion with measurement fusion for multipath OTHR surveillance," *Proceedings of the 4th International Conference on Information Fusion*, 2001.

[60] M. G. Rutten and D. J. Percival, "Joint ionospheric and target state estimation for multipath OTHR track fusion," *Proc. SPIE Signal and Data Processing of Small Targets* 2001.

[61] T. Sathyan, T.-J. Chin, S. Arulampalam, and D. Suter, "A Multiple Hypothesis Tracker for Multitarget Tracking With Multiple Simultaneous Measurements," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 448–460, June 2013.

[62] S. Schoenecker, P. Willett and Y. Bar-Shalom, "Extreme-Value Analysis for ML-PMHT, Part 1: Target Trackability", *IEEE Trans. Aerosp. Electronic Systems*, 2014 (in print).

[63] S. Schoenecker, P. Willett and Y. Bar-Shalom, "Extreme-Value Analysis for ML-PMHT, Part 2: False Track Probability", *IEEE Trans. Aerosp. Electronic Systems*, 2014 (in print).

[64] S. Schoenecker, P. Willett, and Y. Bar-Shalom, "Maximum Likelihood Probabilistic Multi-Hypothesis Tracker Applied to Multistatic Sonar Data Sets," *Proc. SPIE Conf. on Signal Processing, Sensor Fusion, and Target Recognition*, #8050-09, Orlando, FL, March 2011.

[65] S. Schoenecker, P. Willett, and Y. Bar-Shalom, "ML-PDA and ML-PMHT: Comparing Multistatic Sonar Trackers for VLO Targets Using a New Multitarget Implementation," *IEEE Journal of Oceanic Engineering*, 39(2): 303–317, April 2014.

[66] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE Transactions on Signal Processing*, vol. 56, no. 8, pp. 3447-3457, August 2008.

[67] L. Svensson, D. Svensson, M. Guerriero, and P. Willett, "Set JPDA filter for multitarget tracking," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4677-4691, October 2011.

[68] G. Terejanu, P. Singla, T. Singh, and P. D. Scott, "Uncertainty propagation for nonlinear dynamic systems using Gaussian mixture models", *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 6, pp. 1623–1633, November–December 2008.

[69] X. Tian, and Y. Bar-Shalom, "A consistency based Gaussian mixture filtering approach for the contact lens problem," *IEEE Transactions on Aerospace and Electronic Systems*, to appear 2013.

[70] —, "Coordinate conversion and tracking for very long range radars," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 3, pp. 1073–1088, July 2009.

[71] X. Tian, Y. Bar-Shalom, G. Chen, K. Pham, and E. Blasch, "Track splitting technique for the contact lens problem," *Proc. 14th International Conference on Information Fusion*, Chicago, IL, July 2011.

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

[73] R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan, "The unscented particle filter," Cambridge University Engineering Department, Technical Report CUED/F-INFENG/TR 380, 2000.

[74] R. van der Merwe, E. and Wan, "Gaussian mixture sigma-point particle filters for sequential probabilistic inference in dynamic state-space models," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 6, pp. 701–4, April 2003.

[75] H. L. Van Trees, and K. L. Bell, *Bayesian bounds for parameter estimation and nonlinear filtering/tracking*, 2007.

[76] M. P. Wand and M. C. Jones, *Kernel Smoothing*. Chapman and Hall/CRC, 1995.