A Bayesian Framework for Target Tracking using Acoustic and Image Measurements

A Thesis
Presented to
The Academic Faculty

by

Volkan Cevher

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
January 2005

A Bayesian Framework for Target Tracking using Acoustic and Image Measurements

Approved by:

Dr. James H. McClellan, Chair
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Aaron D. Lanterman
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Lance Kaplan, Adjunct
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Monson Hayes III
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Branislav Vidakovic
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date Approved: 3 December 2004

This dissertation is dedicated to my family.

Mahmut, Aysel, and Emine Selen Cevher:

Thank you for your love, encouragement, and support.

ACKNOWLEDGEMENTS

I probably could have written an acknowledgement *chapter* that is of comparable size to the tracking parts of this thesis. In life, sometimes it is not the end result that matters, but the journey that leads to it that does. This dissertation is an important step in my life; however, what is more important to me are the connections that I made with some of the brightest people coming from all around the world.

I would like to thank my advisor, Dr. James H. McClellan, for being the person that he is. Should you meet him, you will see that a great intellectual can also be modest and humble. Throughout my time here at Georgia Tech, rather than giving me a road to follow, he showed me a map and helped me understand how to use it. I am grateful to have had him as my advisor.

I also would like to thank Dr. Aaron Lanterman for getting me started with the particle filters that occupy so much of this thesis. I would like to thank him for taking his time to teach me particle filters and for lending me so many of his books – he is like a second library in GCATT. Next, I would like to thank Dr. Lance Kaplan for all of his discussions. He has helped me improve my presentations and papers (and eventually this thesis). I am also fortunate to have had both Aaron and Lance on my thesis reading committee and their comments are greatly appreciated.

I have so many friends to thank. First, my group-mates: Mubashir Alam, Chris Alvino, Milind Borkar, Ali Cafer Gürbüz, Ryan Hersey, Qiang Le Sam Li, Rajbabu Velmurugan, and Yeo-Sun Yoon. Then, my other friends at CSIP and school: Toygar Akgün, Majid Fozunbal, Chris Nee, Raviv Raich, Martin Tobias, and Dihong Tian. My friends at SDR: Bilgin Altundaş, Irina Borovskaya, Francois Chaudoreille, Thomas Lebrat, Valery Polyakov, Maria Ponomarenko, and Sébastien Prangère. Their friendship and help were a major part of this thesis.

I also wish to thank my close friends for their patience, support, and kindness. I am

especially indebted to Soner Özgür who is, in many ways, a man of great proportions (0.1 metric ton). I truly enjoyed my daily conversations with him on all matters of life. I am also grateful to my roommate Yetkin İleri and Deniz Doğan for their great friendship. Some people say that Yetkin and I are inseparable; I hope that this will be true after I graduate. Next, I would like to thank Can Uslay for his insightful comments on life; Cansu Altınbüken for reminding me of the reasons why I work so hard; and Tsuyoshi and Zhuqing Yamashita for their close friendship.

I am grateful to the CSIP staff, Christy Ellis, Kay Gilstrap, and Charlotte Doughty, for helping me solve my administrative problems. I also would like to thank my friend Richard Dansereau at Carleton University, Dr. Rama Chellappa and Aswin Sankaranarayanan at the University of Maryland for their help and support.

I would like to also thank our CSIP faculty that created this unique environment at Georgia Tech for me and other students to study.

Finally, I would like to express my deepest appreciation to my family: Mahmut Cevher, my father, whose vision and wisdom inspired me in life; Aysel Cevher, my mother, whose great strength enabled me to try harder at my problems; E. Selen Cevher, my sister, whose trust encouraged me on all the way. I also wish to thank Aynur and Sultan Cevher for their support and Gülhanım and Yusuf Cevher for having such a son as my father.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

For multiple target tracking using acoustic signals, we introduce a new motion model that incorporates an acceleration component along the heading direction of the target. We also show that the target motion parameters can be considered part of a more general feature set for target tracking, e.g., target frequencies, which may be unrelated to the target motion, can be used to improve tracking performance. To include the frequency variable, a new array steering vector is defined for direction-of-arrival (DOA) estimation problems. The independent partition particle filter (IPPF) is used to compare the performances of the two motion models by tracking multiple maneuvering targets using the acoustic sensor outputs directly. To decrease the dependence on the observation model, we rework the acoustic tracker using a flexible observation model based on an image tracking approach. Assuming that the DOA observations might contain clutter and that some of the DOAs might be missing in the observation set, we develope a new tracker to track multiple maneuvering targets. The tracker has a sampling strategy that is robust under the observation noise.

Acoustic nodes, each containing an array of microphones, can track targets in x-y space from their received acoustic signals, if the node positions and orientations are known exactly. However, it is not always possible to deploy the nodes precisely, so a calibration phase is needed to estimate the position and orientation of each node before doing any tracking or localization. An acoustic node can be calibrated from sources of opportunity such as beacons or a moving source. We derive and compare several calibration methods for the case where the node can hear a moving source whose position can be reported back to the node. Since the microphone array can track the angle to the moving source, three DOA-based calibration methods can be formulated based on combining angle estimates, geometry, and the motion dynamics of the moving source. In addition, a maximum-likelihood (ML) solution is presented, along with a Newton-based search algorithm that avoids calculating the whole ML surface. The ML estimate serves as a basis for comparison, so the Cramér-Rao

lower bound on the node position estimates is also derived to show that the effect of position errors from the moving source on the estimated node position is much less severe than the variance in angle estimates from the microphone array. Since calibration from a moving source is, in effect, the dual of a tracking problem, methods derived from particle-filter trackers are used to obtain a robust calibration process.

As a recursive algorithm, the particle filter requires an initialization phase prior to tracking a state vector. These initial samples must be generated from the received data and usually obey a complicated probability distribution. The Metropolis-Hastings (MH) algorithm has been used for sampling from intractable multivariate target distributions and is well suited for the initialization problem. Asymptotically, the MH scheme creates samples drawn from the exact distribution. For the particle filter to track the state, the initial samples need to cover only the region around its current state, marked by the presence of modes. Since the particle filter only needs samples around the mode, we have modified the MH algorithm to generate samples distributed around the modes of the target posterior. By simulations, we show that this "mode hungry" algorithm converges an order of magnitude faster than the original MH scheme for both unimodal and multimodal distributions. Finally, we develope a general framework for the joint state-space tracking problem. A proposal strategy for joint state-space tracking using the particle filters is defined by carefully placing the random support of the joint filter in the region where the final posterior is likely to lie. Computer simulations demonstrate improved performance and robustness of the joint state-space when using the new particle proposal strategy.

CHAPTER I

INTRODUCTION

Target tracking is a broad subject area extensively studied in many engineering disciplines. In this thesis, target tracking implies the temporal estimation of target features such as the target's direction-of-arrival (DOA), the target's boundary pixels in a sequence of images, and/or the target's position in space. Target tracking becomes a signal processing problem when an observer demands automated tracking in the presence of noise or interferance. Some of the challenges addressed in this thesis are (i) developing robust algorithms with high estimation accuracy, (ii) correctly identifying the cases where the algorithms fail to sufficiently explain the phenomena, and (iii) merging independent information streams from two or more sensors to improve tracking capabilities.

The tracking problem is formulated in terms of state-space models. State-space models are mathematical relations used for describing a system's evolution and have extensive applications in many practical problems in control theory, signal processing, and telecommunications. Since exact state-space models of real systems are extremely rare, approximate models are used. Hence, during analytical modelling of some natural phenomena, emphasis is placed on choosing a minimum set of variables essential for completely describing a system's internal status relevant to the problem at hand. This way, satisfactory results can still be achieved despite incomplete modelling of a system due to ignorance or lack of knowledge [14, 44].

Once a system's state-space is described in a probabilistic fashion, sequential Monté-Carlo methods, also known as *particle filters*, can be used to track the state vector as the observations arrive in sequence. In a particle filter, the posteriors describing the state vector are represented by randomly distributed discrete state realizations, called the particles, along with some weights. A *proposal function* determines the internal distribution of the particles and directly affects the efficiency of the filter in representing a target posterior.

Given the random particle support, the particle filter can estimate any statistics of the posterior by proper weighting, and the estimation accuracy can be improved up to the theoretical bounds by increasing the number of particles [25, 48].

This thesis first focuses on the familiar acoustic tracking problem using microphone arrays. Although the problem is a classical one, the advent of particle filters in the late 90's and the increase in computing power, following Moore's law, rejuvenated the problem and has enabled researchers to consider more involved state-space models as alternatives. For example, Chapters 2 and 3 look at some complex state-spaces that can be used for acoustic target tracking using particle filters. We then consider a dual problem to target tracking and discuss ways to determine an observer's position given the on-board measurements of the target track (e.g., GPS). This is known as the calibration problem because it determines the sensor location, which is usually assumed to be known. Various algorithms, such as Monté-Carlo Markov chain sampling methods, are presented for the solution of the calibration problem. Then, an interesting accelerated sampling method is described that speeds up the initialization procedure for the particle filter trackers as well as the convergence of the Monté-Carlo calibration algorithm.

Finally, we consider the fundamental problem of merging two independent sensor systems for target tracking. The motivation for joint estimation is basically two-fold: (i) improve the performance of the estimates by merging different information streams and (ii) maintain adequate robustness of the estimates in the face of unexpected model variations or noisy data. For the joint state-space estimation problem, the particle filter is used because it propagates the probability density function (pdf) of the state vectors. Hence, it allows for heuristic combination methods, which may be problem specific [23], or a general probabilistic framework for combining information.

A Bayesian framework is described for tracking a single joint state vector by merging two overlapping state-space models. A proposal strategy is given that carefully combines the proposal strategies optimal for the individual state-spaces such that the random support of the particle filter is concentrated where the final posterior of the joint state-space lies. The resulting filter has better estimation accuracy than the individual filters while using

comparable computational power. Finally, the framework is applied to a synthetic target tracking problem where both acoustic and video observations of a target are available.

The organization of this chapter is as follows. Because Monté Carlo sampling methods form the basis of the solution techniques in this thesis, a representative review is first provided. Then, the acoustic tracking problem is introduced using a narrow-band observation model along with a locally linear motion model. Next, the calibration problem is presented and its duality to the tracking problem is discussed. Finally, a motivation for joint state-space tracking is given using a specific application scenario. The chapter concludes with a summary of the original contributions of this thesis.

1.1 Monté-Carlo Markov Chain Methods

Consider yourself trying to cross a street where there are no traffic lights. Because of your self-preservation instinct, you first process your visual and auditory inputs to determine whether or not there are any moving cars near by. If there is any, then the problem becomes finding their positions, speeds, and headings to determine a crossing path that avoids collusion.

In many practical situations, we are asked to make some inference on some parameter vector $\mathbf{x}$ (e.g., target motion parameters) given some observations $\mathbf{y}$ (e.g., a sequence of images) about that parameter vector. If there is some function that relates $\mathbf{x}$ to $\mathbf{y}$ such as

$$\mathbf{y} = h(\mathbf{x}) \tag{1}$$

then, the solution is intuitively easy: a single (point) estimate for the parameter vector $\mathbf{x}$ can be made by using the inverse functional dependence $\mathbf{x} = h^{-1}(\mathbf{y})$. This problem is commonly known as the parameter inversion/estimation problem [63]. Later on, the reader will recognize that the many problems in this thesis can be formulated in this generic format. At this point, it should be noted that solving the parameter using the inverse functional is not easy task. It has many sensitivity and robustness issues and should be avoided if possible.

The Bayesian framework treats this problem as one where the continuous parameter $\mathbf{x}$

is a random quantity that is statistically related to the observations $\mathbf{y}$[1]. Instead of making point parameter estimates, a probability density function (pdf) on the parameter is obtained given the observations:

$$p(\mathbf{x}|\mathbf{y}) : \text{Probability density of } \mathbf{x} \text{ given } \mathbf{y}. \tag{2}$$

Probability density functions constitute a self-consistent mathematical modelling of the information about a parameter that allows us to conveniently make various inferences [61]. Mean, mode, median, and high order moment estimates are commonly used in the literature to understand or explain a parameter given its pdf. The pdfs are of particular importance in this thesis because multimodality of a pdf might be more important than its single statistics.

The Bayes' rule is used to determine the pdf (2) by reversing the order:

$$p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x}) \tag{3}$$

where the pdf $p(\mathbf{x}|\mathbf{y})$ is also called the *posterior* (or the target density) for the vector $\mathbf{x}$, $p(\mathbf{y}|\mathbf{x})$ is called the data-likelihood given the state, and $p(\mathbf{x})$ is the state *prior*. Equation (3) simply implies that the target posterior distribution is proportional to the product of the reactive effect of the measurements and our initial belief. The proportionality[2] in (3) is independent of the state $\mathbf{x}$ and is given by the reciprocal of the following expression

$$\int p(\mathbf{y}|\mathbf{x})p(\mathbf{x})d\mathbf{x}. \tag{4}$$

To determine the target posterior, $p(\mathbf{y}|\mathbf{x})$ (or $\mathbf{y} = h(\mathbf{x})$) should be known in the Bayesian framework. However, analytical relationships might not be available. However, if the parameter space is discretized, then other methods can be used to determine the target density pointwise. This would be analogous to using an A/D converter to digitize a continuous signal to use the digital signal processing methods to represent the signal.

Monté Carlo Markov chain (MCMC) methods provide a general statistical framework to simulate multivariate distributions using computer simulations. The key idea of the MCMC

[1]In the later sections, we will consider the case where the parameter vector $\mathbf{x}$ is also evolving as the observations arrive in sequence. For the purposes of this section, $\mathbf{x}$ is assumed to be static.

[2]The symbol $\propto$ implies that the left hand side of the expression is equal to the right hand side multiplied by a constant.

algorithms is that it is possible to construct a Markov chain whose stationary distribution is the *target* distribution that we would like to simulate. MCMC methods can then be applied to simulate the pdfs by sampling from the density functions. Since directly sampling from a distribution is a difficult task in itself, a Markov chain is simulated with a transition kernel so that the stationary distribution of the Markov chain can be used to represent the target distribution.

Among the MCMC methods, the Gibbs sampling algorithm, Metropolis-Hastings algorithm, and acceptance-rejection (AR) schemes are well-known in the literature [15, 24, 26, 29, 31, 32, 37, 51, 68, 74, 75]. The Gibbs sampler can be shown to be special case of the MH algorithm [30]. In the next subsection, we will describe how to calculate samples from a complex distribution (i.e., a general posterior) using the AR and MH algorithms. Then, for evolving systems where it is required that a posterior be updated as the observations arrive, we will illustrate how other methods, called the sequential Monté-Carlo methods, can be used to decrease the computational cost required by the brute force MCMC algorithms.

1.1.1 Acceptance-Rejection Algorithm

The objective of the acceptance-rejection (AR) algorithm is to generate samples from a density $\pi(x) \propto \alpha(x)$, where $\alpha(x)$ is called the unnormalized density. Table 1 gives the pseudo-code for generating samples x from $\pi(x)$ by using another density $\beta(x)$ to be defined later[3]. The Markov chain generated by this process has the target posterior distribution in the limit [75]. For completeness, we present a brief proof to show that the stationary distribution of the simulated Markov chain has the desired density.

Let $\beta(x)$ be a density that satisfies $\alpha(x) \leq c\beta(x)$ for some constant c. By using the scheme in Table 1[4], we show that the cumulative distribution of the chain has the desired

[3]The function $\beta(x)$ should be easier to sample than the function $\alpha(x)$.

[4]The symbol $\sim$ means "is sampled from."

form:

$$
\begin{aligned}
P\left(x \leq t \middle| u \leq \frac{\alpha(x)}{c\beta(x)}\right) &= \frac{P(x \leq t, u \leq \frac{\alpha(x)}{c\beta(x)})}{P(u \leq \frac{\alpha(x)}{c\beta(x)})} \\
&= \frac{\int_{-\infty}^{t} \int_{-\infty}^{\frac{\alpha(x)}{c\beta(x)}} \beta(x) du dx}{\int_{-\infty}^{\infty} \int_{-\infty}^{\frac{\alpha(x)}{c\beta(x)}} \beta(x) du dx} \\
&= \frac{\int_{-\infty}^{t} \frac{\alpha(x)}{c\beta(x)} \beta(x) dx}{\int_{-\infty}^{\infty} \frac{\alpha(x)}{c\beta(x)} \beta(x) dx} \\
&= \frac{\int_{-\infty}^{t} \pi(x) dx}{\int_{-\infty}^{\infty} \pi(x) dx}.
\end{aligned} \tag{5}
$$

Table 1: Acceptance-Rejection Sampling

---

i. Generate a sample y from $\beta(y)$.

ii. Generate a sample $u \sim \mathcal{U}(0,1)$, the uniform distribution on $(0,1)$.

iii. if $u \leq \frac{\alpha(y)}{c\beta(y)}$, then $x = y$. Otherwise go to step i.

Even though the algorithm generates the candidate samples from an independent function, it manages to distribute them according to the target distribution. Note that the movement of the chain is global. As it iterates, the chain can move to a random location on the pdf regardless of its current position. This is in contrast to how the Metropolis-Hastings sampling is done, which is explained below.

1.1.2 Metropolis-Hastings Algorithm

The Metropolis-Hastings (MH) algorithm borrows from the popular AR sampling method. The difference between the algorithms comes from how the chain moves from one iteration to another iteration. Denote $q(x, y)$ as a candidate generating density that can produce some candidates y in the numerical sense given x. Since we are interested in Markov chains, the candidate-generating density is written as a function of both the current state x and the proposed state y. Now, when the generated samples are both from the underlying

distribution $\pi(x)$, the following condition should be satisfied:

$$\pi(x)q(x,y) = \pi(y)q(y,x) \tag{6}$$

Equation (6) is also known as the reversibility condition or the detailed balance [24]. This equation implies that the density $\pi(\cdot)$ is the invariance distribution of the transition kernel $q(x,y)$ in Markov chain terminology. This equation is very important since it not only gives us a condition to be satisfied, but also tells us what to do when it is not satisfied.

Let us assume that

$$\pi(x)q(x,y) > \pi(y)q(y,x) \tag{7}$$

which implies that the Markov chain moves from x to y more often than it moves from y to x. The simplest solution to correct this imbalance is to accept the moves from x to y with some probability $\alpha(x,y)$[5]. Now, to determine the correct move probability, we use (6):

$$\pi(x)q(x,y) \times \alpha(x,y) = \pi(y)q(y,x) \times 1 \tag{8}$$

Hence, we obtain the acceptance probability of a move as

$$\alpha(x,y) = \min\left\{1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\right\} \tag{9}$$

Note that if a new candidate is rejected, the algorithm keeps the current state as the new one. Moreover, if the candidate-generating density is symmetric, then the MH algorithm simply accepts moves that increase the likelihood of the state. Table 2 shows the algorithm pseudo-code.

An important final remark on the MH sampling algorithm is in order here. As in all MCMC algorithms, the convergence of this algorithm is very slow. Considerable attention needs to be given to the choice of the candidate-generating function $q(\cdot,\cdot)$ since it determines the length of the transient stages of the algorithm. Unfortunately, there is no easy way to determine whether or not the algorithm has converged. The literature contains some ad-hoc ways of determining the convergence speed of this algorithm in the literature [32,70]. Because this algorithm is crucial in initializing the acoustic trackers in Chapter 2 and 3,

[5]Implicity, we assume that since the moves from y to x are less often, we accept them with probability 1.

Table 2: Metropolis-Hastings Algorithm

i. Generate a candidate y using $q(x, y)$ and x.

ii. Calculate the acceptance ratio $\alpha(x, y)$:

$$\alpha(x, y) = \min\left\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\right\}.$$

iii. Sample $u \sim \mathcal{U}(0, 1)$.

iv. If $u \leq \alpha(x, y)$, set $x = y$ (accept the move), else, reject it.

and also for the Monté-Carlo calibration solution in Chapter 4, we will show a modification in Chapter 5 that increases the convergence rate significantly under some regularity assumptions. The resulting distribution of the modified MH scheme is an approximation of the true density by Gaussian distributions around the modes of the density.

1.1.3 Sequential Monté-Carlo Methods

The MH algorithm can be implemented to determine the target density function for a dynamically varying system by solving the target density at each iteration separately. To solve for the target density, the algorithm would be initialized *uninformed* and then would be run until convergence is reached. This would not be particularly clever if the system is not changing drastically at each time iteration because the results from the previous time estimation would be discarded when the system evolves to the next time step.

If the system is changing slowly, as in the case of most target tracking scenarios considered here, the distance between the target densities at each time step is expected to be small. Hence, it would be a good idea to keep the previous target density and *reuse* it to help reconstruct the current target density. Monté Carlo methods that use this strategy to sample target posteriors are known as sequential Monté Carlo methods [25, 48]. They are also known as *particle filters*.

Under the Markovian assumption[6], consider the following state-space model described by

$$\begin{aligned} x_t &\sim q_t(x_t|x_{t-1}) \\ y_t &\sim f_t(y_t|x_t) \end{aligned} \tag{10}$$

The function q_t is called the *state density function*; and the observations are explained through the observation density function f_t. Here, we also introduce the cumulative notation: $\mathbf{x}_t = \{x_0, x_1, \ldots, x_t\}$. The target posterior $p(\mathbf{x}_t|\mathbf{y}_t)$ has the form

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_t) &= p(x_0, x_1, \ldots, x_t|y_0, y_1, \ldots, y_t) \\ &= \frac{p(y_t|\mathbf{x}_t, \mathbf{y}_{t-1})p(\mathbf{x}_t|\mathbf{y}_{t-1})}{p(y_t|\mathbf{y}_{t-1})} \\ &= \frac{p(y_t|\mathbf{x}_t, \mathbf{y}_{t-1})p(x_t|\mathbf{x}_{t-1}, \mathbf{y}_{t-1})}{p(y_t|\mathbf{y}_{t-1})} \times p(\mathbf{x}_{t-1}|\mathbf{y}_{t-1}) \end{aligned} \tag{11}$$

or equivalently in terms of the state update and observation functions:

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_t) &\propto f_t(y_t|x_t)q_t(x_t|x_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{t-1}) \\ &\propto \prod_{s=1}^{t} f_s(y_s|x_s)q_s(x_s|x_{s-1})p(x_0|y_0) \end{aligned} \tag{12}$$

where the proportionality is independent of the state. Here we emphasize that the inherent recursive structure in (11) will be exploited while devising the sequential sampling strategy. Now, we would like to determine the target distribution at time t given the distribution $p(\mathbf{x}_{t-1}|\mathbf{y}_{t-1})$. However, before we do that, we will explain how to represent a pdf by using a properly *weighted* set of discrete samples.

The distribution $p(\mathbf{x}_{t-1}|\mathbf{y}_{t-1})$ can be given numerically by a set of discrete state realizations of $\mathbf{x}_{t-1}^{(j)},\ j = 1, 2, \ldots, N$ where N is the number of particles. These realizations can be (i) distributed exactly according to $p(\mathbf{x}_{t-1}|\mathbf{y}_{t-1})$, or (ii) distributed according to some other function $g(\mathbf{x}_{t-1}|\mathbf{y}_{t-1})$ with some proper weights $w_{t-1}^{(j)},\ j = 1, 2, \ldots, N$ to be discussed later. We now give an example of the first case where the representation is exact:

[6]The Markovian assumption is a term commonly used in the signal processing and statistics literature. It implies that the current state distribution, given the previous state and the current observation, is independent of the other states and observations [64].

Example 1: In this example, we will visualize what it means to represent a pdf using discrete realizations. Suppose that the target distribution that we would like to represent is the following Gaussian density:

$$p(\mathbf{x}) = \frac{1}{\sqrt{2\pi \det(\Sigma)}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)\right\} \tag{13}$$

where

$$\mu = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \mathbf{\Sigma} = \begin{bmatrix} 1 & .6 \\ .6 & 4 \end{bmatrix}. \tag{14}$$

We first generate $N = 1000$ particles drawn exactly from the target distribution by using the Cholesky decomposition of the covariance matrix and an *i.i.d.* Gaussian noise generator (shown in Fig. 1). In the figure, an ellipse is shown to emphasize the correlation of how the particles are distributed. This ellipse is generated by the following equation:

$$(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu) = 4. \tag{15}$$

The discrete representation of $p(\mathbf{x})$ shown in Fig. 1 is useful because, given the particles, it is easy to calculate the sample mean and the sample covariance of the Gaussian numerically:

$$E\{\mathbf{x}\} = \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}^{(j)} = \begin{bmatrix} 1.0883 \\ 0.9293 \end{bmatrix} \tag{16}$$

and

$$\begin{aligned} E\left\{(\mathbf{x}-\mu)(\mathbf{x}-\mu)^T\right\} &= \frac{1}{N}\sum_{j=1}^{N}\left(\mathbf{x}^{(j)} - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}^{(i)}\right)\left(\mathbf{x}^{(j)} - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}^{(i)}\right)^T \\ &= \begin{bmatrix} 1.0663 & 0.6189 \\ 0.6189 & 3.9222 \end{bmatrix} \end{aligned} \tag{17}$$

where the last covariance expression is known to be biased [66]. These estimates converge to the true estimates asymptotically (i.e., $N \to \infty$)■

For the second case, i.e., when the particles are distributed according to another function, the attached weights $w^{(j)}$ are chosen so that we can do inferences on the state, we can do it by using

$$E\{h(\mathbf{x})\} \approx \frac{\sum_{j=1}^{N} h(\mathbf{x}^{(j)}) w^{(j)}}{\sum_{j=1}^{N} w^{(j)}} \tag{18}$$

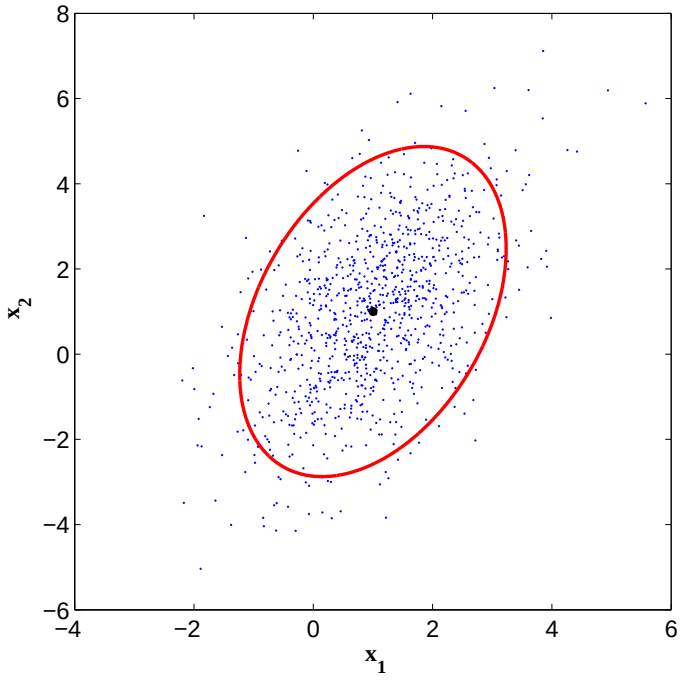


Figure 1: The pdf in (13) is represented by particles directly generated using the distribution itself. The analytical expression for the ellipse is given by (15).

for any integrable function h. Equation (18) is a convenient way of evaluating the expectation without any integration. This second way of representing the posterior is the result of the Bayesian importance sampling method where the basic idea is to distribute the particles according to some function g called the *proposal function* so that the weights satisfy (18). This way, if we choose a proposal function that is easy to sample from (e.g., a Gaussian distribution), significant computational power can be saved because the numerical integration of h is replaced by a pointwise evaluation in (18). This requires that the support of g should cover the support of the target distribution [25]. The weights can be calculated using a simple ratio of the target density to the proposal density:

$$w^{(j)} \propto \frac{p(\mathbf{x}^{(j)})}{g(\mathbf{x}^{(j)})} \tag{19}$$

The following example explains why this is the case.

Example 2: Assume that the target distribution is the same Gaussian as in Example 1, with μ and $\mathbf{\Sigma}$ given in (14). Also, assume that it is difficult to generate correlated samples (e.g., the Cholesky decomposition is not available), hence, instead of the target distribution,

the following proposal function is used to represent the target density:

$$g(\mathbf{x}) \sim \mathcal{N}(\mu_g, \mathbf{\Sigma}_g) \tag{20}$$

where μ_g is the same as μ in (14) and Σ_g is just the diagonal entries of Σ:

$$\mathbf{\Sigma}_g = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix}. \tag{21}$$

Figure 2 shows the distribution of the particles generated by this proposal function. Note the mismatch between the two ellipses caused by the correlation in the original pdf. To determine the proper weights, consider the expectation integral:

$$\begin{aligned} E\{h(\mathbf{x})\} &= \int h(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \\ &= \int h(\mathbf{x}) \frac{p(\mathbf{x})}{g(\mathbf{x})} g(\mathbf{x}) d\mathbf{x} \\ &= E_g \left\{ h(\mathbf{x}) \frac{p(\mathbf{x})}{g(\mathbf{x})} \right\} \end{aligned} \tag{22}$$

By noting the similarity of (22) to (11) and (26), it is straightforward to show that the weights are given by (19).

Then, given the weights, the following estimates can be made:

$$E\{\mathbf{x}\} = \sum_{j=1}^{N} w_*^{(j)} \mathbf{x}^{(j)} = \begin{bmatrix} 0.9834 \\ 0.9885 \end{bmatrix} \tag{23}$$

and

$$\begin{aligned} E\left\{(\mathbf{x}-\mu)(\mathbf{x}-\mu)^T\right\} &= \sum_{j=1}^{N} w_*^{(j)} \left(\mathbf{x}^{(j)} - \sum_{i=1}^{N} w_*^{(i)} \mathbf{x}^{(i)} \right) \left(\mathbf{x}^{(j)} - \sum_{i=1}^{N} w_*^{(i)} \mathbf{x}^{(i)} \right)^T \\ &= \begin{bmatrix} 1.0133 & 0.6382 \\ 0.6382 & 3.9980 \end{bmatrix} \end{aligned} \tag{24}$$

where

$$w_*^{(j)} = \frac{w^{(j)}}{\sum_j w^{(j)}}. \tag{25}$$

Note that although the particles are misaligned with the true distribution, the estimated mean and variance are very good because of the correction effect of the weights. Also, the estimation performance is again expected to improve as we increase the number particles

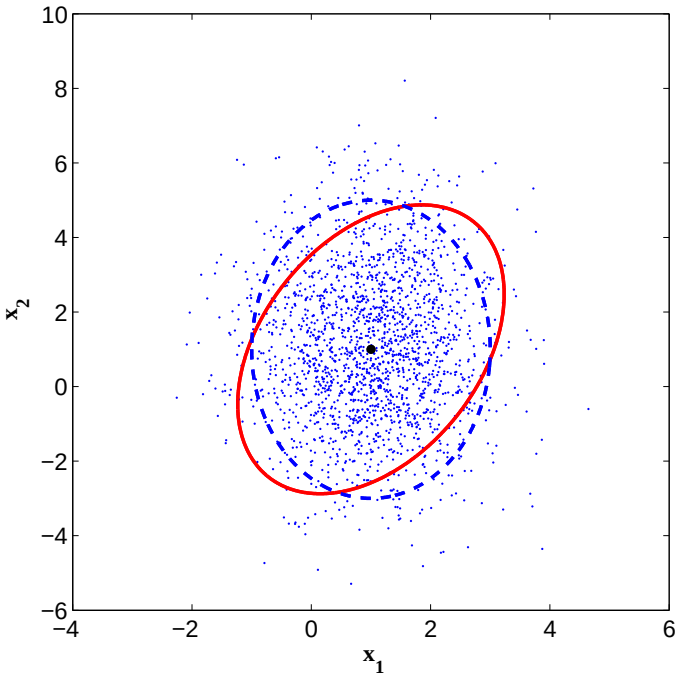


Figure 2: The pdf in (13) is represented by $N = 1000$ particles generated using the proposal distribution $g(\mathbf{x})$ described by (20). The analytical expression for the ellipse (solid line) is expressed by (15). This ellipse shows the pdf we would like to represent using the particles. The ellipse drawn with the dashed line is given by $(\mathbf{x} - \mu_g)^T \Sigma_g^{-1} (\mathbf{x} - \mu_g) = 4$. The dashed ellipse shows how the particles are distributed.

because, in the limit, the random support of the particles generated by the proposal function will cover the whole space of $\mathbf{x}$.■

For the state-space described by (10), the posterior at time t can be represented by the pair $(\mathbf{x}_t^{(j)}, w_t^{(j)})$ where

$$w_t^{(j)} \propto w_{t-1}^{(j)} \frac{f_t(y_t|x_t) q_t(x_t|x_{t-1})}{g(x_t|x_{t-1}, y_t)} \tag{26}$$

and $\mathbf{x}_t^{(j)} \sim g(x_t^{(j)}|x_{t-1}, y_t)$. The choice of the proposal function $g(x_t|x_{t-1}, y_t)$ is important because it creates a trade-off between the performance of the filter and its computational complexity. For a given number of particles, the performance of the filter increases as the proposal function better approximates the target posterior. In general, the mismatch of the proposal function and the target pdf, as in Example 2, will decrease the estimation performance. However, to better approximate the posterior, more computation might be required while generating the samples from the proposal function (e.g., extra computation is required to generate correlated samples in Example 1.)

Although the above scheme of updating weights sequentially as the data arrive seems to be mathematically convenient, another step, called *resampling*, is required. As the filter iterates, all the weights, except for a few, that represent the posterior tend to zero after a few steps [25]. Hence, after a few iterations, not only is the posterior under-represented, but also computation is wasted on calculating particles with weights that have no contribution to the final estimates. This is known as the degeneracy of the particle filter algorithm [25]. The objective of the resampling stage is to rejuvenate the important particles to produce better future states as the system evolves and, therefore, to keep the variance of the (normalized) weights as small as possible[7].

There are many modifications to this general sequential sampling algorithm. Auxiliary particle filters use an index variable to increase robustness, whereas Rao-Blackwellization uses a partitioning idea to decrease the computational complexity [17, 48, 62]. There are also various resampling schemes such as simple random sampling, residual sampling, and Monté Carlo resampling [48]. Regardless of the modifications, the structure of the filter is

[7] If the weights in (26) are all ones, then the proposal function is pointwise equal to the posterior (Example 1). Obviously, this is the best representation of the posterior.

very simple:

- Generate particles using a proposal function,
- Evaluate the proper weights,
- Do inference and then, if required, resample.

Equipped with the basics of Monté-Carlo methods, we now move onto tracking problems formulated in a probabilistic fashion. Specifically, we will first consider the acoustic target tracking problem.

1.2 Acoustic Target Tracking

Acoustic arrays consist of multiple microphones, possibly directional, that listen to ambient sounds emitted by objects in the surrounding environment. They are attractive in target tracking because they operate passively, allow random deployment in the field, are cheap compared to other sensor modalities (e.g., IR cameras), and have relatively low power requirements. They can also be employed as an integral part of an intelligent collaborative sensing network, which may also include different types of sensors such as seismic sensors, magnetic sensors, imaging sensors, and so on.

Acoustic arrays can track a target's position if the target is in the near-field of the array. If the array size (also known as the aperture size) is relatively large such that the acoustic waves are seen as spherical waves by the array, then spatial filtering algorithms can be used to determine the target position in polar coordinates (R, θ) where R refers to the range of the target and θ is the DOA of the target measured with respect to the array reference (Fig. 3). However, in most practical cases, the sounds arrive as plane waves because the target is far away from the array and only DOA information can be extracted [42].

We will focus on far-field tracking problems where the target DOA track needs to be calculated given the acoustic microphone outputs. Accordingly, the acoustic microphone outputs first need to be related to the target DOAs under some target signal assumptions. One important observation model that achieves this objective is called the narrow-band observation model. This model relates the acoustic array response to a constant frequency

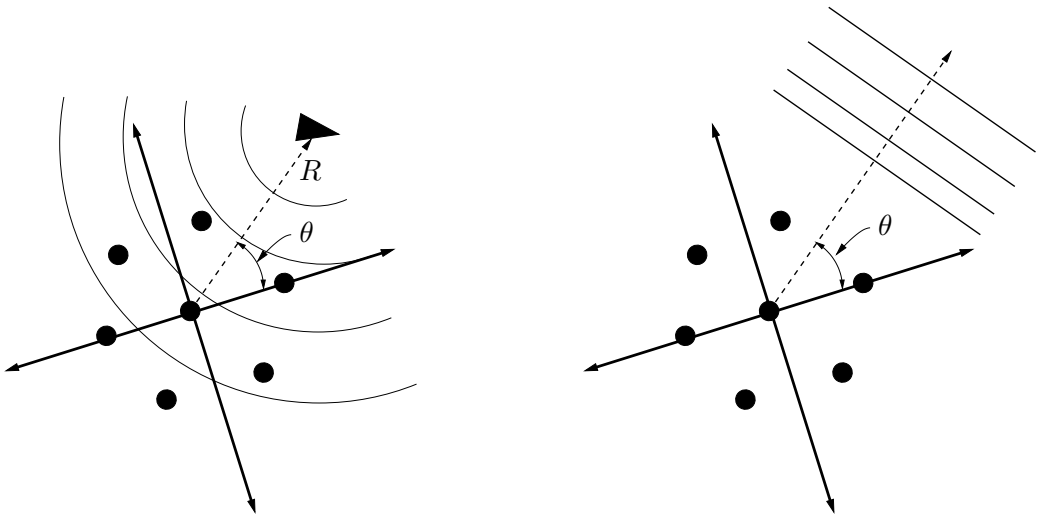


Figure 3: Microphone positions are shown with black dots. When the target is in the near-field of the array, acoustic sounds can be used to determine the location of the target described by the polar coordinates: (R, θ). When the target is in the far-field of the array, only angle information can be extracted. In this case, multiple arrays are needed to determine the location of the target.

source at DOA θ through some steering vectors under isotropic and non-dispersive medium assumptions.

1.2.1 Narrow-band Observation Model

To derive the narrow-band observation model, consider K far-field targets coplanar with a sensor node consisting of P acoustic sensors. The sensor node (or microphone array) is not assumed to possess any special structure. A steering vector associated with the array defines the array response for a source at DOA θ. The complex envelope of the received narrow-band signal with center frequency[8] f_0 at the p^{th} sensor corresponding to the k^{th} target can be written as

$$s_k(t) \propto e^{-j2\pi f_0 t} \Rightarrow s_k\left(t - ({\alpha_k}^T \mathbf{z}_p)\right) = s_k(t) e^{-j2\pi f_0 ({\alpha_k}^T \mathbf{z}_p)}, \quad k = 1, 2, \cdots, K \tag{27}$$

where $\mathbf{z}_p$ is the p^{th} sensor position and $\alpha_k \triangleq (1/c)\,[\,\cos\theta_k\,,\,\sin\theta_k\,]^T$ is the k^{th} slowness vector in Cartesian coordinates (c is the speed of sound.) Equation (27) leads to the

[8]Each target is assumed to have the same narrow-band frequency f_0. Although counterintuitive, the case where each target has a different frequency is easier to handle through temporal filtering.

following array steering vector for the k^{th} source signal:

$$\mathbf{a}(\theta_k) = \begin{bmatrix} e^{-j2\pi f_0 ({\alpha_k}^T \mathbf{z}_1)} \\ e^{-j2\pi f_0 ({\alpha_k}^T \mathbf{z}_2)} \\ \vdots \\ e^{-j2\pi f_0 ({\alpha_k}^T \mathbf{z}_P)} \end{bmatrix} \tag{28}$$

Then, the array outputs are a mixture of K target signals impinging on the array as follows:

$$\mathbf{y}(t) = \mathbf{A}(\mathbf{\Theta}(t))\mathbf{s}(t) + \mathbf{n}(t) \tag{29}$$

where $\mathbf{y}(t) \in \mathbb{C}^{P\times 1}$ is the noisy array output vector, $\mathbf{n}(t) \in \mathbb{C}^{P\times 1}$ is a complex additive noise, and $\mathbf{s}(t) \in \mathbb{C}^{K\times 1}$ is the signal vector. $\mathbf{A}(\mathbf{\Theta}(t)) \in \mathbb{C}^{P\times K}$ is called the steering matrix and has the steering vectors in its columns:

$$\mathbf{A}(\mathbf{\Theta}(t)) = \Big[\, \mathbf{a}(\theta_1),\, \mathbf{a}(\theta_2),\, \cdots,\, \mathbf{a}(\theta_K) \,\Big], \quad \mathbf{\Theta}(t) = \Big[\, \theta_1,\, \theta_2,\, \cdots,\, \theta_K \,\Big] \tag{30}$$

This model possesses an interesting mathematical structure and has a significant impact on the DOA estimation problem. It has led to many practical and fast DOA estimation techniques such as MUSIC, MVDR, and eigenvalue beamformers [16, 42, 71, 76].

Beamforming is the name given to a wide variety of array processing algorithms that focus an array's signal processing capabilities in a particular direction [42]. Narrow-band beamformers use a batch of data and the array model (29) to determine an array covariance matrix assuming target stationarity. Then, this covariance matrix and the steering vectors (30) are used to generate power-vs-angle patterns whose peaks correspond to the target DOAs.

In general, the estimation performance of algorithms using the narrow-band model deteriorates when the target signals have a wide-band structure, or when rapid target motion spreads the array spatial spectrum [33,77]. Unfortunately, the complexity of the observation model increases significantly as some of the assumptions in this model are relaxed to more accurately represent reality.

DOA estimation methods based on batch processing (e.g., MUSIC, MVDR, etc.) do not reuse the information from the previous batch to help refine the estimates at the current

batch. Some tracking algorithms achieve information transfer from the previous batch to *a priori* information in the current batch by imposing motion constraints on the target [18, 60, 77]. This way, the estimated DOAs avoid the bias effects caused by non-stationary array covariances at the expense of more computational power.

1.2.2 State Model Based on Constant Velocity Motion

In many problems, the DOAs are calculated by a beamformer and are assumed to have a normal distribution around the true DOA track. Then, a motion model (e.g., modified polar coordinates [38]) is used to refine the DOA estimates using Kalman filtering [2, 4, 27, 43, 54]. Unless the batch period used to calculate the DOAs is small, the DOAs will be biased because the array covariance matrix becomes non-stationary [77]. Figure 4 shows an example where, due to rapid target movement, the target DOA changes from 90 to 83.5 degrees in 1 second. If the acoustic data generated by the narrow-band model is processed by the MVDR beamformer using a batch period of 1 second, the target DOA is estimated around 87 degrees (shown with the solid horizontal line). Note that MVDR reports this DOA as the target DOA at time $t = 0$ s. However, if MVDR uses a batch period of 1/20 seconds, the estimated DOAs distribute themselves around the true DOA track with a negligible bias. In this case, it is possible to also estimate motion parameters as well as a better target DOA estimate at time $t = 0$s. Later chapters will show how to use motion models to improve DOA estimates.

The constant velocity motion model described below is used in acoustic target tracking problems because it is strongly locally observable given the acoustic data [77]. We will discuss the importance of observability in this scenario later on, but first, we make the following distinction in the application of motion models: Motion models can filter the pre-calculated DOA estimates to project the noisy estimates on to a space constrained by the motion (smoothing), or they can assist in the calculation of the DOA estimates in conjunction with the smoothing operation.

The motion model can be derived by assuming that the targets have constant speeds with a Brownian disturbance acting on their heading directions [18, 60, 77]. This model has

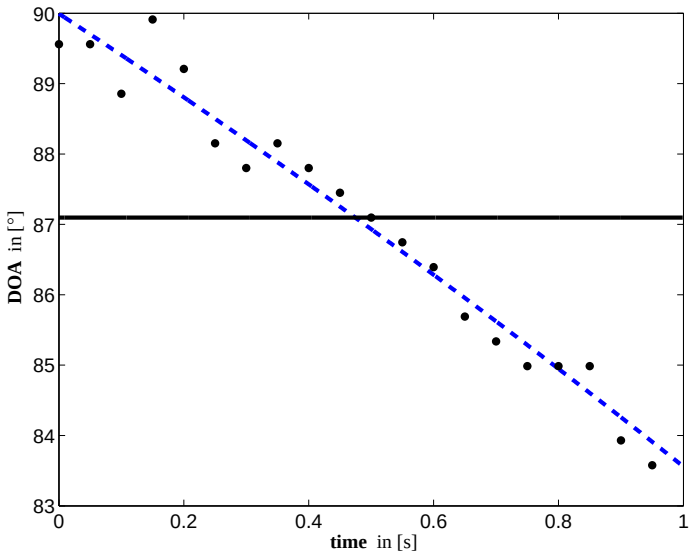


Figure 4: The true DOA track is shown with the dashed line. Black dots are the MVDR estimates with a batch size of 1/20s, whereas the solid horizontal line corresponds to a batch size of 1s. The sampling frequency of the acoustic data is $F_s = 1000$Hz and the center frequency of the target is $f_0 = 40$Hz. A uniform circular array with 15 microphones is used, where the minimum microphone distance is 0.45 times the wavelength of the target sinusoid signal. The array signal-to-noise ratio is 7dB.

the following state vector:

$$\mathbf{x}(t) \triangleq \begin{bmatrix} \theta(t) \\ Q(t) \\ \phi(t) \end{bmatrix} \tag{31}$$

where $\theta(t)$ and $\phi(t)$ are the target DOA and the heading direction. The compound variable $Q(t) \triangleq \log q(t)$ is the logarithm of target velocity over range: $q(t) \triangleq v/r(t)$. Figure 5 illustrates the geometry of the problem. Target DOAs are measured clockwise with respect to the y-axis, whereas the target heading directions are measured counterclockwise with respect to the x-axis.

The state update equation can be derived by relating the DOAs of the target at times t and $t+T$ using the geometrical relation of position 1 at $(r(t)\sin\theta(t), r(t)\cos\theta(t))$ to position 2 at $(r(t)\sin\theta(t) + vT\cos\phi(t), r(t)\cos\theta(t) + vT\sin\phi(t)))$ (Fig. 5). Then, it is straightforward to obtain the following update relations:

$$\tan\theta(t+T) = \frac{r(t)\sin\theta(t) + vT\cos\phi(t)}{r(t)\cos\theta(t) + vT\sin\phi(t)} \tag{32}$$

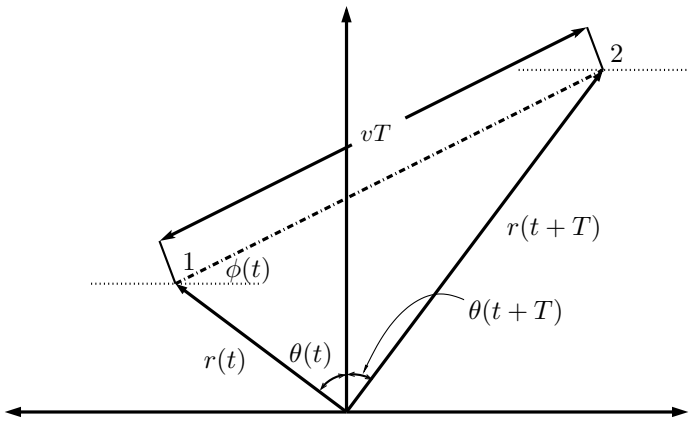


Figure 5: The target is at position 1 at time t and moves to position 2 in T seconds with a constant speed. The target is assumed to be in the far-field of the sensor array whose center coincides with the origin.

and

$$r(t+T) = \sqrt{r^2(t) + 2r(t)vT\sin\left(\theta(t)+\phi(t)\right) + v^2T^2} \tag{33}$$

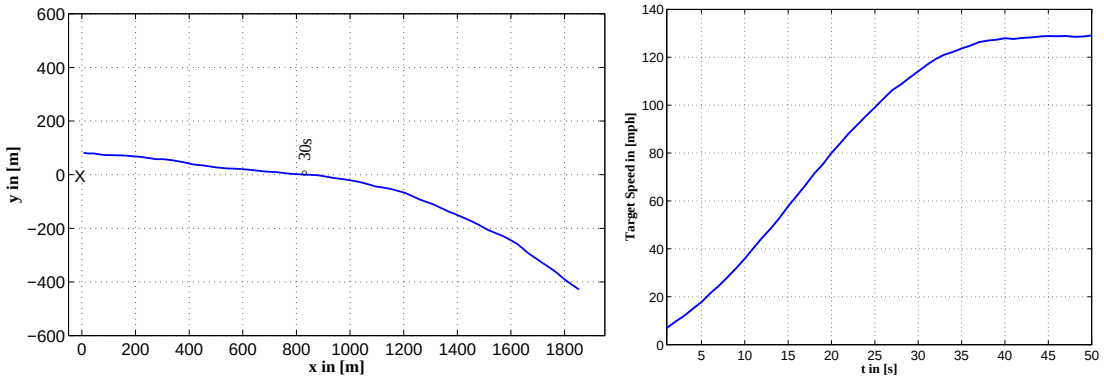
Equations (32) and (33) form a scalable system for the target motion dynamics at hand. To elaborate on this, consider scaling the range and the speed of the target by the same factor. It can be shown that this scaled target has the same set of update equations as above since the scale factor can be cancelled out. This, in fact, leads to the introduction of the compound variable $q(t)$ defined earlier. In the state update, however, the logarithm of $q(t)$ is used since an additive noise component can be employed (as opposed to multiplicative noise when $q(t)$ is used) for notational convenience. Also, note that target maneuvers (e.g., perpendicular accelerations) are modelled in the state update through the state noise on $\phi(t)$:

$$\begin{aligned}\mathbf{x}(t+T) &= \hat{\mathbf{f}}(\mathbf{x}(t), \mathbf{u}(t+T)) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{u}(t+T) \\ &= \begin{bmatrix} \arctan\left\{\frac{\sin\theta(t)+e^{Q(t)}T\cos\phi(t)}{\cos\theta(t)+e^{Q(t)}T\sin\phi(t)}\right\} \\ Q(t) - 1/2\log[1+2e^{Q(t)}T\sin\left(\theta(t)+\phi(t)\right) + (e^{Q(t)}T)^2] \\ \phi(t) \end{bmatrix} + \begin{bmatrix} u_\theta(t+T) \\ u_Q(t+T) \\ u_\phi(t+T) \end{bmatrix}\end{aligned} \tag{34}$$

where $\mathbf{u} \sim \mathcal{N}\left(\mathbf{0}, \text{diag}\{\sigma_\theta^2, \sigma_Q^2, \sigma_\phi^2\}\right)$. When the target speed becomes negative due to acceleration, the Q term becomes complex due to the logarithm. This problem can be solved

by updating only the real part of Q, and adding the imaginary part (which is π when q is negative) to the heading parameter ϕ, which, in effect, will reverse the direction of the target.

This model is useful for explaining the motion of the targets whose velocity is changing slowly. However, when the targets are actually accelerating, the model might lead to incorrect motion estimates. Figures 6, 7, and 8 illustrate the performance of the independent partition particle filter (IPPF) [60] that uses the constant velocity motion model applied to an accelerating target. Chapter 2 will discuss this filter in more detail. The target track and velocity evolutions are given in Fig. 6. Even though the IPPF DOA estimates are close to the true DOA estimates as illustrated in Fig. 7(a), the target track generated using the state estimates and the correct initial target position is not correct (Fig. 7(b)). This is because the target acceleration is explained through the changes in the target heading variable (Fig. 8(b)). The simulation parameters for this example are given in Chapter 2 because this example will be reused to compare the constant velocity model with the acceleration model developed in this thesis.

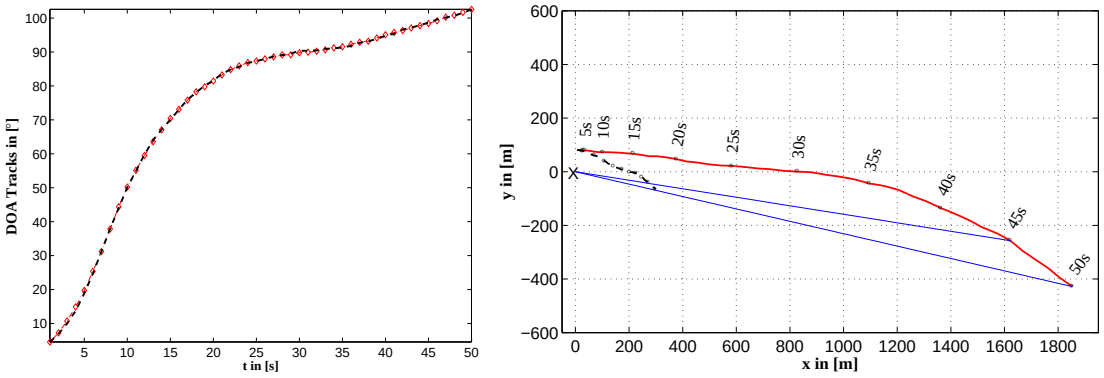


(a) (b)

Figure 6: (a) The acoustic array is situated at the origin. The target moves away from the sensor with constant acceleration for about 30s. Then it has almost constant speed for $t > 30$s. (b) Target speed vs. time. At time $t = 30$s, the target also starts to maneuver.

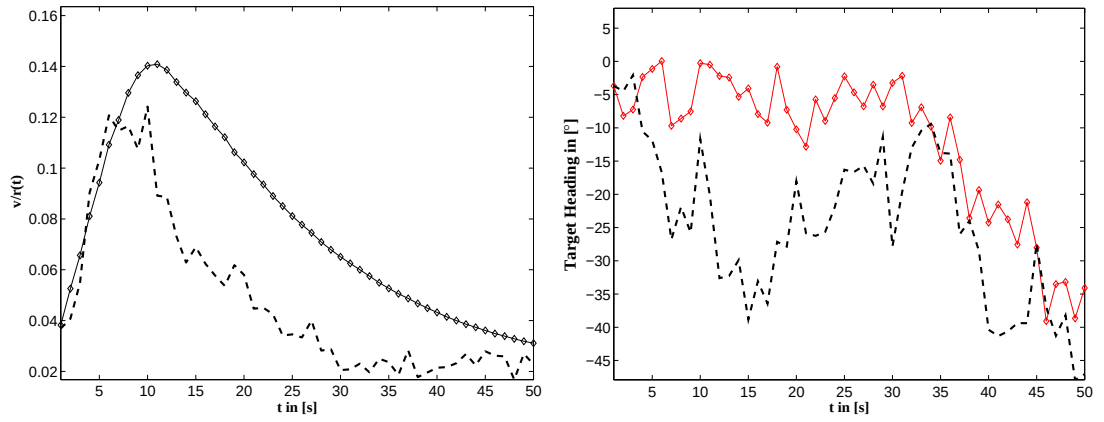
1.2.3 State-of-the-Art

The IPPF algorithm is an application of the particle filter algorithm to the tracking framework developed by Zhou *et al.* [60, 77]. The algorithm is novel due to (i) the usage of the



(a) (b)

Figure 7: (a) Diamonds correspond to the true target DOA track. DOAs estimated by the IPPF algorithm are shown with the dashed line. (b) Dashed line shows the target track estimate constructed using the IPPF estimates.



(a) (b)

Figure 8: (a,b) IPPF estimates (dashed lines) are compared to the true motion parameter values (diamonds).

partitioning approach and the derivation of the related proposal functions, (ii) application of local linearization techniques for approximating the acoustic state posterior, and (iii) a novel Monté Carlo Markov chain initialization for the state vector. Also, the authors use reference priors to derive a probability density function for the observations given the state by marginalizing out the signal vectors and the unknown signal variance in the narrow-band observation model [11].

The state partitioning approach is useful to decrease the computational cost of particle filters when multiple objects are being tracked [49, 50]. Partitioned sampling allows you to operate in several lower dimensional state-spaces as opposed to working on the full state-space [50]. To generate the combined state vector for multiple targets, each target vector is considered as an independent partition as they are generated. Thus, the filter proposal function generates each partition independently to be combined as a particle. To do this, it also assigns each partition a probability from the data, and the particles are formed by concatenating the vectors sampled from the generated partitions with respect to their partition probability. This way, partitions for each target with high partition probabilities are selected more often than other partitions with low probability, resulting in better particle proposals. Hence, the filter can avoid sampling the whole state space, effectively beating the curse of dimensionality.

The IPPF paper [60] presents the particle filter backbone for the state-space model developed by Zhou *et al.* [77]. The multiple target tracking algorithm developed by Zhou employs the Newton search algorithm to minimize the negative of a maximum-likelihood cost function derived from the narrow-band observation model and the constant velocity motion model. Their paper [77] also also derive the gradient and the Hessian terms needed for the Newton algorithm and also demonstrates approximations for making the filter more robust. Moreover, it appears that [77] was the first paper to demonstrate the spatial spectrum spread caused by rapid target movement. This biasing effect is the main cause for DOA errors in some tracking scenarios and hence its derivation is one of the main contributions of [77]. The data association problem is also handled automatically by the Newton search algorithm by simultaneously estimating the target vectors using a global ML cost

function.

The multiple target tracking algorithms by Zhou [77] and Orton [60], and also another acoustic tracking algorithm by Sword *et al.* [72], have built-in mechanisms that handle data association issues. In general, DOA tracking algorithms handle data association in various ways: (i) probabilistic data association methods estimate the states by summing over all the association hypotheses weighted by the probabilities obtained by the likelihood [6, 7, 22, 58], (ii) smoothness assumptions on the target (motion) states allows a natural ordering of the data [45], (iii) computationally costly ML/EM methods use the likelihood functions to search for a global maximum [65], and (iv) nearest neighbor methods provide easy heuristics to perform measurement updates. Most of these methods use mean and the covariance approximations on the sufficient statistics of the state, which may be estimated with a Kalman filter. However, for nonlinear state-spaces with non-Gaussian noise assumptions, Monté-Carlo methods are needed to adequately capture dynamic, possibly multimodal, statistics.

Observation models, motion models, noise assumptions, multiple observers, and data association problems all add to the complexity of the acoustic target tracking problem. These models are needed to assume away many issues in the tracking problem so that the resulting systems are implementable under some criterion. Quite often, this criterion is not always the best estimation performance. In most cases, signal processing algorithms must be developed under a power-performance trade-off. Also, algorithm designs must be flexible for fusing other information streams provided by sensor networks.

In the later chapters, we will show our improvements to the observation/motion models. For now, a dual problem to acoustic tracking is considered in the following section.

1.3 Calibration: the Dual of Target Tracking

In the far-field tracking problem, we know that an acoustic array can determine the DOA of a target given the acoustic observations. Now, consider the acoustic tracking problem where the unknown target sounds propagate in air and are collected at a stationary acoustic array. In this problem, a crucial assumption is that the array center position and a reference

angle for the array orientation are known, so that the DOA measurement of the target is meaningful. Now, assume that the target has an on-board GPS system that provides reasonably accurate position estimates that are then communicated to the acoustic array. Then, we can reverse the acoustic tracking problem statement: given the maneuvering target position information and its acoustic data, how can the array determine its own position and orientation?

According to the results of Nardone's paper [55], it is possible to determine the array position because the acoustic array is stationary, and hence the target is always outmaneuvering the array. The orientation estimation problem is introduced as a side problem in this case because the acoustic array is only capable of estimating angles. Hence, the orientation can be treated as an auxiliary parameter in the system[9]. Here, it is important to note that the array solves for its own position and orientation by receiving the missing information: the target GPS track.

The reader, naturally, might be asking why the acoustic array would not have a GPS receiver to eliminate this calibration problem? There are a few reasons: (i) having an accurate GPS receiver increases the battery requirements for the acoustic array, (ii) it also makes the array more susceptible to jamming by outside sources, and (iii) an independent calibration method can refine the node position and orientation information even if a GPS receiver is available (self-diagnostic).

Sensor array calibration might imply different problems, depending on the context. One common problem is the calibration of individual microphone positions, which form the array manifold for the acoustic array outputs. For this type of problem, methods to find the sensor positions and the Cramér-Rao lower-bound expressions have been given [28, 57]. In this thesis, sensor array calibration implies the calibration of a network of sensor nodes each consisting of omnidirectional microphones. The individual positions of each microphone are known within the sensor node, and the node is capable of detecting, tracking, and classifying acoustic targets. There has been work on calibrating these types of sensor arrays, which

[9]That is, the position of the array can be found without determining the array's orientation. The reverse is not true.

are also known as unattended ground sensors (UGS) [20], [53]. Under the assumption that each node can estimate the time-of-arrival and the DOA of acoustic sources and that some known acoustic calibration sources are available, individual node positions and orientations can be found as shown in [53].

A new approach is proposed in this thesis to achieve the same goal of calibrating UGSes with a set of assumptions different than those used in [53]. Node position estimation is first performed on the acoustic microphone outputs directly, as opposed to some auxiliary node output such as DOA estimates. The maximum-likelihood (ML) estimator of the node position is derived using an acoustic array data model. Then, a Newton type of search algorithm is introduced to avoid calculating the whole ML surface. For the Newton algorithm, an initial node position and orientation close to the true values should be estimated. This can be achieved by using DOA calibration algorithms, such as a synthetic-aperture based node localization algorithm. Other special calibration methods based on the Metropolis-Hastings algorithm and relative motion concepts are also developed. The common theme for these methods is their estimation of the node positions by first calculating target DOAs and then using some geometrical arguments. We also give Cramér-Rao performance bounds for the calibration problem for the different estimation methods and consider the sufficiency of these auxiliary DOA estimates.

1.4 Joint State-Space Tracking

Particle filters, whose mechanics are based on the classical narrow-band array observation model, have poor performance when applied to field data acquired with acoustic nodes because they do not take into account (i) target harmonics, (ii) acoustic propagation losses, and (iii) temporary wideband characteristics of the observed target signals. These recursive trackers also require good initialization for determining the number of targets and their initial motion parameters depending on the adapted motion model. When the targets of interest exhibit harmonics, it is difficult to determine the number of targets correctly. This is because it is nearly impossible to separate two targets with different frequencies at the same DOA from one target with multiple harmonic frequencies. Moreover, as the targets enter

the detection range of the acoustic nodes, they have low signal-to-noise ratios (SNR) due to acoustic propagation losses, which also makes it harder to estimate the motion parameters robustly for initialization.

Recently, hybrid nodes that contain an acoustic array collocated with a camera have been proposed. The camera usually has a narrow field-of-view (FOV), but it likely would be steerable. Also, it could be an IR or optical camera, or even stereoscopic. New algorithms are needed to intelligently fuse the information coming from both modalities under some performance-power cost function. However, before tackling the performance vs. power analysis, the reason for acoustic-video fusion must be addressed.

Video DOA tracking algorithms can provide high resolution estimates at each frame, and are relatively insensitive to the target range. They can also count the number of targets much more robustly than other types of sensor modalities. A video tracking algorithm would produce more reliable and higher resolution multi-target motion state estimates than an acoustic node, especially at long range. However, the following disadvantages of the video necessitate an auxiliary sensor modality such as acoustics:

Narrow FOV: The detection region of a camera is limited by its narrow FOV. A second modality such as acoustics can detect targets over the full 360 angle range and cue the video.

High Power Consumption: Video tracking algorithms require high frame rates for robust target tracking. When imaging at moderate resolutions, video requires more power than an acoustic sensor or a seismic sensor. Hence, if the auxiliary sensor can help reduce the frame rate for video tracking or eliminate it when possible, the lifetime of the hybrid node can be maximized under some acceptable performance level.

Target Occlusion: A video tracker is incapacitated when the target is occluded, for example, by dust clouds. However, acoustics can take over the tracking during an occlusion scenario and help the video recover after the occlusion is over. Also, if the hybrid node is blocked by some object such as a tree (because of its narrow fov) during deployment ,

the second modality may still enable the node tracking capability. Acoustics is one such example.

Multimodal fusion might be required for a number of reasons. The most important of these reasons comes from the observability issue of the individual modalities. The ultimate objective of any tracking system is to determine the target position in a defined coordinate system. However, the target position might not be readily observable by most of the sensor modalities due to the observation process or modelling. For example, an acoustic node cannot distinguish the positions of two targets in the far-field of the array if the targets are moving with the same DOA track. A camera might be able to estimate the position of a target in a polar coordinate system by scaling, but only if the target template is available. However, by merging two acoustic node's DOA estimates along with the array position information, the actual target positions may be observed, which achieves an important fusion objective.

Other important fusion objectives might include, but are not limited to (i) lowering the power costs of the individual modalities, (ii) achieving robust tracking, and (iii) increasing the resolution of the estimates. In the case of a joint acoustic-video tracker that increases the resolution of the DOA estimates, the combined tracker can handle more general motion models but might be overkill because the video tracker can already make high resolution estimates with low sensitivity to the target range. However, the interplay of the two modalities can generate a robust tracker with the ability to correctly count the number of targets and track them through occlusions or through the regions of low acoustic signal-to-noise ratio. Hence, if a robust tracker is formulated so that the power required by the video can be adaptively varied through the help of acoustics, the above fusion objectives can be achieved under some desirable power-performance trade-off.

There are various practical issues related to having a fully joint acoustic-image tracker using a particle filter. The first issue, not directly related to the particle filter, is the time-varying bias between the camera observations and the acoustic observations due to the acoustic propagation delays. The acoustic observations always have a time lag compared to the video observations, and this lag varies with the target range. The second issue is

directly related to the particle filter. If the joint tracker is to use a common state vector that can explain both acoustic and image observations, some parts of this new state vector will only be observable by one modality, while others will be observed by both modalities. For example, the target shape variables used in video are not observable by the acoustic signals, whereas the target DOA is observable in both domains. Hence, a novel sampling strategy needs to be developed to effectively place the random support of the particles to represent the target posterior used for the tracking.

In Chapter 6, we propose a particle filtering structure that can handle multiple sensor modalities. A novel sampling strategy is proposed for generating a common state vector that has variables observable by only some of the modalities and other variables that can be observable by all of the modalities. We discuss the advantages and disadvantages of having a common state vector as a joint-tracker compared to having different trackers and merging their results in a fusion framework. A synthetic acoustic-image tracker is used as an example. The difference in the approach taken in this thesis is the development of the particle filtering framework for a fully joint target tracker as opposed to having different modalities cue each other [23, 36, 73] or combining separate tracking algorithms running at the same time [47].

1.5 Contributions of the Thesis

Traditionally in target tracking, much emphasis is put on the motion model that realistically represents the target's movements. In Chapter 2, we introduce a new motion model that incorporates an acceleration component along the heading direction of the target. We also show that the target motion parameters can be considered part of a more general feature set for target tracking. This is exemplified by showing that target frequencies, which may be unrelated to the target motion, can also be used to improve the tracking performance. To include the frequency variable, a new array steering vector is presented for the direction-of-arrival (DOA) estimation problems. The independent partition particle filter (IPPF) is used to compare the performances of the two motion models by tracking multiple maneuvering targets using the acoustic sensor outputs directly.

To decrease the dependence on the observation model, in Chapter 3, we rework the acoustic tracker using a flexible observation model based on an image tracking approach. Assuming that some of the DOA observations might have clutter in them and that some of the DOAs might be spurious in the observation set, we develop a new tracker to track multiple maneuvering targets. The tracker has a novel sampling strategy that is robust with respect to the observation noise. A Newton algorithm is also applied for efficiently determining the proposal function.

In Chapter 4, we treat the calibration problem for an acoustic node containing an array of microphones at fixed positions with respect to each other. We first present the maximum-likelihood (ML) estimator for the node-center position as well as its orientation. The calibration is based on using a calibrating target that carries a global positioning system (GPS) on board. The effects of the GPS errors on the node position estimates and the related Cramér-Rao lower bounds are also derived. We then present a Newton based search algorithm along with different initializations to avoid calculating the whole likelihood surface. One of these initialization methods is based on calculating the node position and orientation using a synthetic aperture created by the calibrating target. A Metropolis-Hastings method is also introduced and is applied to the field data.

As a recursive algorithm, the particle filter requires initial samples to track a state vector. These initial samples must be generated from the received data and usually obey a complicated distribution. The Metropolis-Hastings (MH) algorithm is used for sampling from intractable multivariate target distributions and is well suited for this initialization problem. Asymptotically, the MH scheme draws samples from the exact distribution. For the particle filter to track the state, the initial samples need to cover only the region around its current state. This region is marked by the presence of modes. Since the particle filter only needs samples around the mode, in Chapter 5, we modify the MH algorithm to generate samples distributed around the modes of the target posterior. By simulations, we show that this "mode hungry" algorithm converges an order of magnitude faster than the original M-H scheme for both unimodal and multimodal distributions.

Finally, in Chapter 6, we develop a framework for the joint state-space tracking problem. A proposal strategy for joint state-space tracking using particles filters is given. The state-spaces are assumed Markovian and inexact; however, each state-space is assumed to sufficiently describe the underlying phenomenon. Joint tracking is achieved by carefully placing the random support of the joint filter where the final posterior is likely to lie. Computer simulations demonstrate improved performance and robustness of the joint state-space through the proposed strategy.

In this thesis, we mainly focus on target tracking problems using the particle filtering framework. In Fig. 1.5, an outline of the thesis is given where we also highlight our contributions that are discussed in this section. The figure should give the reader an overall picture as to what to expect in the later chapters of the thesis.

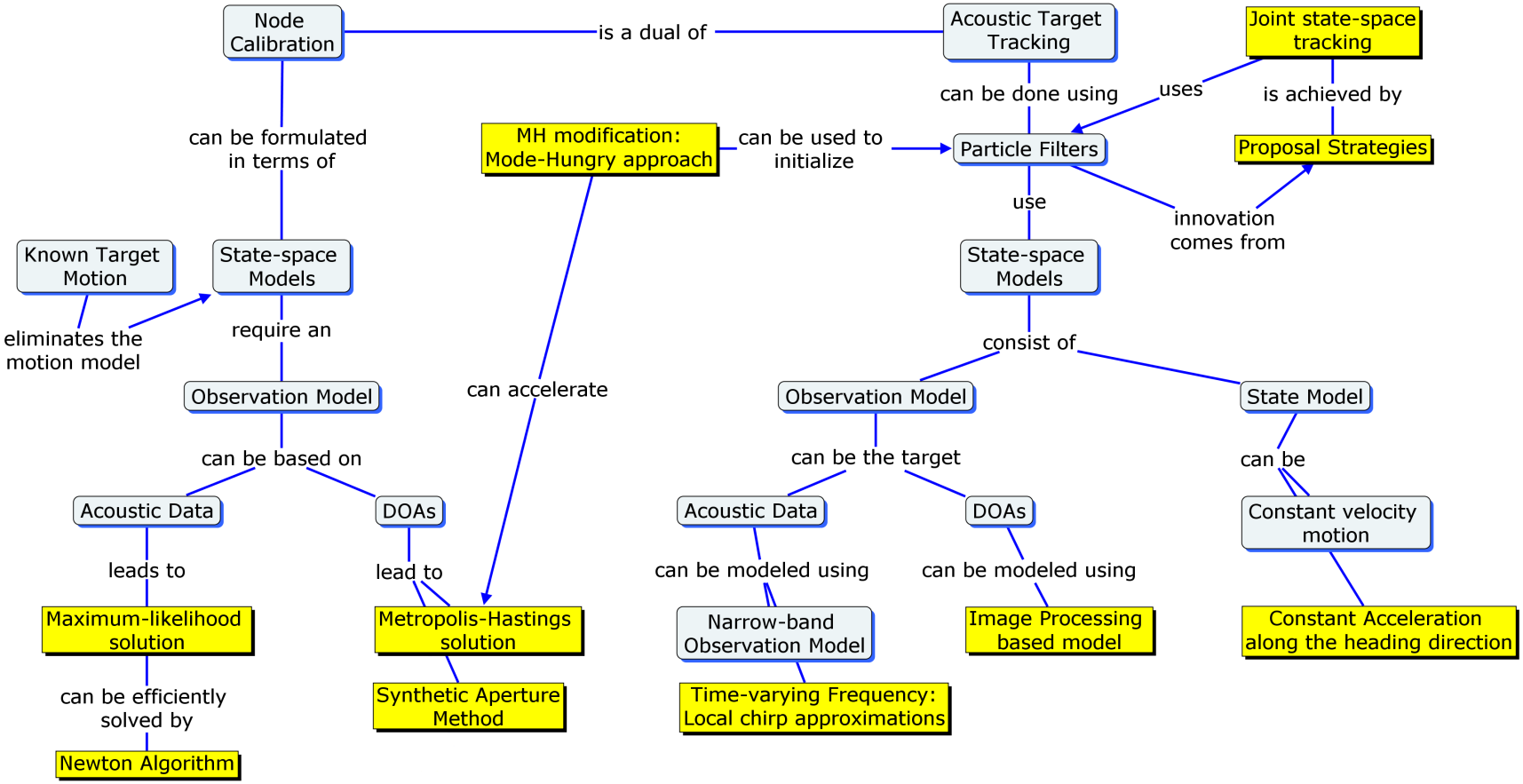


Figure 9: Thesis outline. New contributions are shown in the rectangular boxes.

CHAPTER II

GENERAL DIRECTION OF ARRIVAL TRACKING USING ACOUSTIC NODES

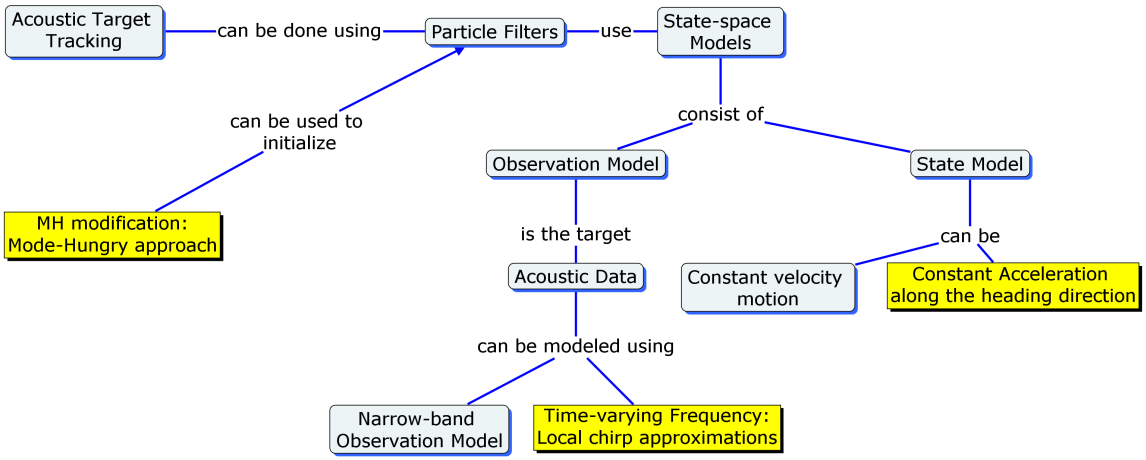


Figure 10: Chapter outline with the new contributions shown in rectangular boxes.

2.1 Introduction

The direction-of-arrival (DOA) estimation problem has been extensively studied in the signal processing literature [42]. Narrow-band solutions based on beamforming such as multiple signal classification (MUSIC) [71], minimum variance beamforming, and Pisarenko's method suffer degraded performance when the targets are moving relatively fast during the estimation batch, also known as the snapshot period. The performance loss in these high resolution algorithms can be attributed to the nonstationarity of the array data caused by rapid target motion. To remedy this problem, one can exploit target motion dynamics to jointly estimate the DOAs while tracking targets [77]. These refined DOA estimates provide better performance in exchange for increased computational complexity.

In the case of wideband acoustic signals, the pioneering work of Wang and Kaveh [76] on coherent subspace processing coherently integrates the array autocorrelation matrices

corresponding to the multiple frequencies of interest, so that the SNR is increased and resolution gains can be achieved. The work by Gershman and Amin [33] approximates the signals within the DOA snapshot period as chirps and performs time-frequency MUSIC on the acoustic array outputs. A time-varying frequency model enables these wideband methods to tackle more realistic estimation problems. However, these algorithms, like the narrow-band techniques, produce snapshot DOA estimates, and hence, require heuristics for target association.

Advances in large scale integration of computer systems have made Monté-Carlo techniques a feasible alternative for solving target tracking problems. Conventionally, given a model of the target dynamics, the underlying motion equations are simplified by linearization and Gaussian noise assumptions so an analytical solution can be obtained. The extended Kalman filter is such a method; it is also the best minimum mean-squares linear estimator for the problem at hand. Monté-Carlo techniques, on the other hand, do not linearize nor do they assume Gaussian noise. They approximate the posterior density of interest, i.e., a density that describes the likelihood of a target's position in space, by particles that represent a discrete version of the posterior. The idea is that if a sufficient number of *effective* particles can be used, the estimation performance will be close to the theoretically optimal solution.

A solution to the multiple target tracking problem has been given for the narrow-band case by Orton and Fitzgerald [60] using an independent partition particle filter (IPPF). The independent partition assumption of the IPPF solves the DOA association problem common to multiple target tracking algorithms. Given the target motion dynamics developed in [77], the implementation of the IPPF is shown in [60] to track constant velocity targets with a Brownian disturbance acting on the target heading directions. In this thesis, the IPPF is again used, but along with a new target motion model that includes accelerations along the heading direction of each target. Simulations show that since the new motion model enables the IPPF to relax the constant velocity assumption, target tracking is improved when the targets have high accelerations, as intuitively expected.

In target tracking, it is usually a good idea to retain certain target features even if they

are not related to the motion parameters. This may, in turn, help the data fusion problem with different types of sensors since the extra algorithmic computational load is usually much less taxing on the energy budget of the sensor than transmitting the raw data to a central processor. Hence, in addition to the motion parameters, we also show how to incorporate new features such as time-varying frequency signatures of the targets into the particle filter algorithm. It is assumed that a separate time-frequency tracker is tracking the dominant instantaneous frequencies of the targets; however, the issues related to the frequency estimation are not considered in this thesis. It is demonstrated by simulations that in the case of multiple targets, the introduction of the frequency variable improves the tracking performance of the IPPF when two targets have similar motion parameters but different time-frequency signatures.

Observability is one of the main concerns in acoustic tracking problems because it must be possible to determine the target states from the array data uniquely. Observability of a time-varying system depends on both the observations and the state equation, i.e., on the acoustic sensor outputs and the motion model [14]. However, when the observations are sufficient to determine the target DOAs, the observability test reduces to a simple rank test on the Jacobian of the local representation of the motion model [77]. In this thesis, it is assumed that the observations are sufficient to determine the target DOAs and the system observability is proved using the observability rank condition. We argue that with the same assumption on the observations, it is possible to automatically generate the IPPF code to do target tracking for some other target motion dynamics as long as the rank condition is satisfied. This enables dynamic switching of motion models appropriate for different types of targets, resulting in a more flexible tracker.

The array model employed in the simulations has a special structure. We use a single node consisting of a circular omnidirectional microphone array that supplies DOA and frequency information about the targets. However, the derivations do not depend on the particular structure of the nodes. Note that tracking implies the temporal estimation of the target DOAs as opposed to target positions. Spatial diversity of multiple nodes can be exploited in triangulating the targets of interest, which also presents a data fusion problem.

However, data fusion from multiple nodes can be facilitated by some of the variables in the target feature set such as target orientation and frequency. It should be noted that these two variables are invariant with respect to the node positions when the nodes have the same reference frame[1]. Hence, in a simple application such as triangularization, if the DOA information coming from each node also includes these features, the data association for multiple targets can be done optimally with minimal information exchange.

The organization of this chapter is as follows. Section 2.2 introduces models that account for the target motion dynamics as well as the acoustic observations. Section 2.3 describes how to construct the necessary probability density functions that form the backbone of the particle filter solution. The IPPF algorithm details are discussed in Sect. 2.4 before the simulation results, which are presented in Sect. 2.5.

2.2 Data Models

Consider K far-field targets coplanar with a sensor node consisting of P acoustic sensors. The sensor node (or sensor array) is not assumed to possess any special structure. Two motion models will be presented, differing in the assumptions of constant velocity versus constant acceleration.

2.2.1 State Model-I

The targets are assumed to have constant speeds with Brownian disturbances acting on their heading directions [60,77]. With the introduction of the frequency variable, the model has the following state vector:

$$\mathbf{x}_k(t) \triangleq \begin{bmatrix} \theta_k(t) \\ Q_k(t) \\ \phi_k(t) \\ f_k(t) \end{bmatrix} \tag{35}$$

where $\theta_k(t)$, $\phi_k(t)$, and $f_k(t)$ are the DOA, the heading direction, and the instantaneous frequency of the k^{th} target. $Q_k(t) \triangleq \log q_k(t)$ is a compound variable where $q_k(t) \triangleq v_k/r_k(t)$.

[1]If the nodes have different reference directions, the orientation angles of targets can be found in another reference frame by simple additions and subtractions of the reference angles of the nodes.

Target DOAs are measured clockwise with respect to the y-axis, whereas target heading directions are measured counterclockwise with respect to the x-axis. Figure 11 illustrates the geometry of the problem. The state update equation can be derived by relating the

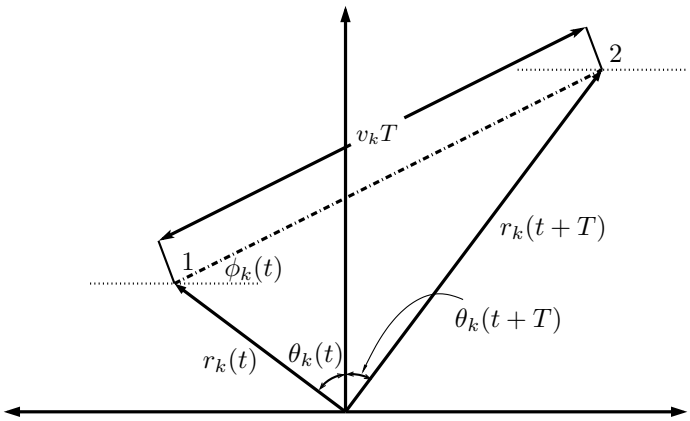


Figure 11: The k^{th} target is at position 1 at time t and moves to position 2 in T seconds with a constant speed. The target is assumed to be in the far-field of the sensor array whose center coincides with the origin.

DOAs of the target at times t and $t+T$ using the geometrical relation of position 1 at $(r_k(t)\sin\theta_k(t), r_k(t)\cos\theta_k(t))$ to position 2 at $(r_k(t)\sin\theta_k(t)+v_kT\cos\phi_k(t), r_k(t)\cos\theta_k(t)+v_kT\sin\phi_k(t))$. Then, it is straightforward to obtain the following update relations:

$$\tan\theta_k(t+T) = \frac{r_k(t)\sin\theta_k(t) + v_kT\cos\phi_k(t)}{r_k(t)\cos\theta_k(t) + v_kT\sin\phi_k(t)} \tag{36}$$

and

$$r_k(t+T) = \sqrt{r_k^2(t) + 2r_k(t)v_kT\sin\left(\theta_k(t)+\phi_k(t)\right) + v_k^2T^2} \tag{37}$$

Equations (36) and (37) form a scalable system for the target motion dynamics. Consider scaling the range and the speed of the k^{th} target. This scaled target has the same set of update equations as above since the scale factor can be cancelled out. This, in fact, leads to the introduction of the compound variable $q_k(t)$ defined earlier. In the state update, however, the logarithm of $q_k(t)$ is used since an additive noise component can be employed (as opposed to the multiplicative noise when $q_k(t)$ is used). Note that the particle filter can cope with a general type noise and that additivity of the noise is not required; however, the logarithm is used to decouple the noise from the state update, which is given below. Also, note that target maneuvers, e.g., the perpendicular accelerations, are modelled in the state

update through the state noise on $\phi_k(t)$:

$$\mathbf{x}_k(t+T) = \hat{\mathbf{f}}_{\mathbf{I}}(\mathbf{x}_k(t), \mathbf{u}(t+T)) = \mathbf{f}_{\mathbf{I}}(\mathbf{x}_k(t)) + \mathbf{u}(t+T)$$
$$= \begin{bmatrix} \arctan\left\{\frac{\sin\theta_k(t)+e^{Q_k(t)}T\cos\phi_k(t)}{\cos\theta_k(t)+e^{Q_k(t)}T\sin\phi_k(t)}\right\} \\ Q_k(t) - 1/2\log[1+2e^{Q_k(t)}T\sin(\theta_k(t)+\phi_k(t)) + (e^{Q_k(t)}T)^2] \\ \phi_k(t) \\ f_k(t) + 2b_k(t)T \end{bmatrix} + \begin{bmatrix} u_{\theta,k}(t+T) \\ u_{Q,k}(t+T) \\ u_{\phi,k}(t+T) \\ u_{f,k}(t+T) \end{bmatrix} \tag{38}$$

where $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \mathrm{diag}\{\sigma_\theta^2, \sigma_Q^2, \sigma_\phi^2, \sigma_f^2\})$. When the target speed becomes negative due to deceleration, the Q_k term becomes complex due to the logarithm. This problem can be solved by updating only the real part of Q_k, and adding the imaginary part (which is π when q_k is negative) to the heading parameter ϕ_k, which, in effect, reverses the direction of the target.

Implicit in (38) is a second order polynomial approximation for the phase of the signals of interest. Hence, for $t \in [t_i, t_i + T)$, the following relation is assumed on the k^{th} signal:

$$s_k(t) \approx \alpha_k(t_i) e^{j2\pi[f_{0,k}(t_i)t + b_k(t_i)t^2]} \tag{39}$$

where $\alpha_k(t_i)$ is the complex amplitude, $f_k(t) = f_{0,k}(t_i) + 2b_k(t_i)t$ is the instantaneous frequency, and $2b_k(t_i)$ is the rate of change of the instantaneous frequency of the k^{th} signal at time t during the i^{th} batch period. This model assumes that the dominant narrow-band frequencies of each target are tracked by separate time-frequency filters. Some frequency tracking examples using Markov chains can be found in [8,9,34]. Note that the state estimate of the IPPF also results in a distribution on the instantaneous target frequencies, which can be exploited by the frequency tracker in determining $b_k(t_{i+1})$ for the next recursion.

The variances of the components of the state noise vector $\mathbf{u}$ are usually small, which may lead to sample impoverishment [5] (explained in Sect. V.A.3). In fact, if the process noise is zero, the state variables can be treated as static variables in an estimation problem where using a particle filter may not be appropriate. Techniques to prevent this sample impoverishment or degeneracy are discussed in [60]. Finally, the state noise vector is chosen

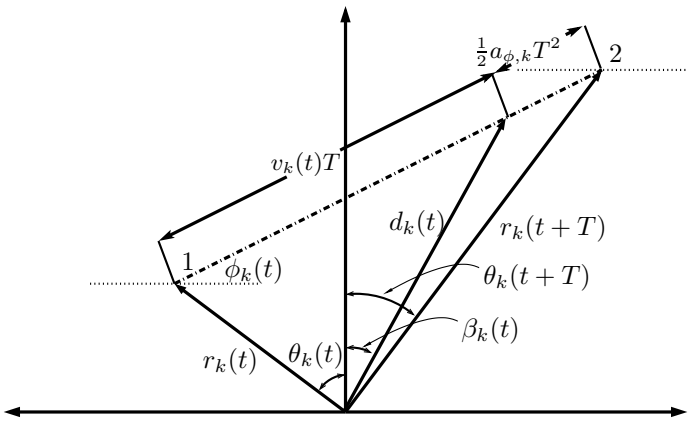


Figure 12: The k^{th} target is at position 1 at time t and moves to position 2 in T seconds with constant acceleration. Notice that $\beta_k(t)$ corresponds to $\theta_k(t+T)$ in Fig. 11.

to be Gaussian due to its analytical tractability. Justifications of this model can also be found in [60].

2.2.2 State Model-II

In this second formulation, the targets are now assumed to have slowly varying accelerations along their heading directions. The new state vector is the following:

$$\mathbf{x}_k(t) \triangleq \begin{bmatrix} \theta_k(t) \\ Q_k(t) \\ \psi_k(t) \\ \phi_k(t) \\ f_k(t) \end{bmatrix}, \tag{40}$$

where $\theta_k(t)$, $Q_k(t)$, $\phi_k(t)$, and $f_k(t)$ are as defined above. The parameter $\psi_k(t)$ is defined using the k^{th} target's acceleration $a_{\phi,k}$ along its heading direction: $\psi_k(t) \triangleq \frac{a_{\phi,k}}{2r_k(t)}$. Figure 12 illustrates the geometry of the problem used to derive the state update relations.

The state update equation can be derived in a manner similar to State Model-I:

$$\mathbf{x}_k(t+T) = \hat{\mathbf{f}}_{\mathbf{II}}(\mathbf{x}_k(t), \mathbf{u}_k(t+T)) = \mathbf{f}_{\mathbf{II}}(\mathbf{x}_k(t)) + \mathbf{u}_k(t+T) \tag{41}$$

with

$$\theta_k(t+T) = \beta_k(t) + \tan^{-1}\left[\frac{T^2\rho_k(t)\cos(\beta_k(t)+\phi_k(t))}{1+T^2\rho_k(t)\sin(\beta_k(t)+\phi_k(t))}\right] + u_{\theta,k}(t+T) \tag{42}$$

where

$$\beta_k(t) = \tan^{-1}\left[\frac{\sin\theta_k(t) + Te^{Q_k(t)}\cos\phi_k(t)}{\cos\theta_k(t) + Te^{Q_k(t)}\sin\phi_k(t)}\right] \tag{43}$$

and

$$\rho_k(t) = \frac{a_{\phi,k}}{2d_k(t)} \tag{44}$$

$$\begin{aligned} Q_k(t+T) = {} & \log\{2T\rho_k(t)[1 + 2Te^{Q_k(t)}\sin(\theta_k(t) + \phi_k(t)) + T^2e^{2Q_k(t)}]^{1/2} + e^{Q_k(t)}\} \\ & - \frac{1}{2}\log[1 + 2Te^{Q_k(t)}\sin(\theta_k(t) + \phi_k(t)) + T^2e^{2Q_k(t)}] \\ & - \frac{1}{2}\log[1 + 2T^2\rho_k(t)\sin(\beta_k(t) + \phi_k(t)) + T^4\rho_k^2(t)] + u_{Q,k}(t+T) \end{aligned} \tag{45}$$

$$\begin{aligned} \psi_k(t+T) &= \frac{\rho_k(t)}{[1 + 2T^2\rho_k(t)\sin(\beta_k(t) + \phi_k(t)) + T^4\rho_k^2(t)]^{1/2}} + u_{\psi,k}(t+T) \\ \rho_k(t) &= \frac{\psi_k(t)T}{[1 + 2Te^{Q_k(t)}\sin(\theta_k(t) + \phi_k(t)) + T^2e^{2Q_k(t)}]^{1/2}} \end{aligned} \tag{46}$$

$$\phi_k(t+T) = \phi_k(t) + u_{\phi,k}(t+T) \tag{47}$$

$$f_k(t+T) = f_k(t) + 2b_k(t)T + u_{f,k}(t+T) \tag{48}$$

The state noise is defined similarly as $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \text{diag}\{\sigma_\theta^2, \sigma_Q^2, \sigma_\psi^2, \sigma_\phi^2, \sigma_f^2\})$. Intuitively, the noise variables u_ψ and u_Q need to be correlated due to the dependence of the velocity on the acceleration along the heading direction. An expression for the correlation between these two noise variables can be derived to better track the targets when the targets are not actually changing their directions. However, as the targets undergo slow maneuvers between each time step, it appears that the correlation between the acceleration along the heading direction at the beginning of the batch and the actual target speed becomes much weaker. Hence, the noise variables on $Q_k(t)$ and $\psi_k(t)$ are modelled independently to better accommodate maneuvering targets, as well as for simplicity.

2.2.3 Observation Model

The sensor array in the simulations consists of P omnidirectional acoustic sensors situated uniformly on a circle of radius R. A steering vector associated with the array defines the complex array response for a source at DOA θ. The received signal at the p^{th} sensor corresponding to the i^{th} target is first derived using the same second order polynomial approximation on the phases of the signals shown in (39). For an isotropic and non-dispersive

medium, the signal received at the p^{th} sensor can be written as

$$\begin{aligned} s_i(t-({\alpha_i}^T\mathbf{z}_p)) &\approx s_i(t)e^{j2\pi[-f_{0,i}(t)({\alpha_i}^T\mathbf{z}_p)-2b_i(t)t({\alpha_i}^T\mathbf{z}_p)+b_i(t)({\alpha_i}^T\mathbf{z}_p)^2]} \\ &\approx s_i(t)e^{j2\pi[b_i(t)({\alpha_i}^T\mathbf{z}_p)^2-f_i(t)({\alpha_i}^T\mathbf{z}_p)]} \end{aligned} \tag{49}$$

where $i = 1, 2, \cdots, K$, $\mathbf{z}_p$ is the p^{th} sensor position, and $\alpha_i \triangleq (1/c)[\cos(\theta_i), \sin(\theta_i)]^T$ is the i^{th} slowness vector in Cartesian coordinates. Equation (49) leads to the following array steering vector for the i^{th} source signal:

$$\mathbf{a}(\theta_i) = \begin{bmatrix} e^{j2\pi\{b_i(t)({\alpha_i}^T\mathbf{z}_1)^2-f_i(t)({\alpha_i}^T\mathbf{z}_1)\}} \\ e^{j2\pi\{b_i(t)({\alpha_i}^T\mathbf{z}_2)^2-f_i(t)({\alpha_i}^T\mathbf{z}_2)\}} \\ \vdots \\ e^{j2\pi\{b_i(t)({\alpha_i}^T\mathbf{z}_P)^2-f_i(t)({\alpha_i}^T\mathbf{z}_P)\}} \end{bmatrix} \tag{50}$$

A similar derivation of the array steering vector for signals with constant narrow-band frequencies can be found in [42]. Note that, in the second line of (49), the term $b_i(t)({\alpha_i}^T\mathbf{z}_p)^2$ can be ignored for small array aperture sizes and slowly varying frequencies and we obtain the same steering vector used in [33].

As mentioned above, the phase characteristics of the signals of interest are approximated with chirps within a batch period T. We also require that each steering vector $\mathbf{a}(\theta_i)$ uniquely correspond to a signal whose direction is the objective of the DOA estimation problem. Signals coming from multiple targets are added to form the observations. For the IPPF, these acoustic observations are updated every $\tau = T/M$ seconds where M denotes the number of batch samples. Then, the array outputs are written as follows:

$$\mathbf{y}(t) = \mathbf{A}(\mathbf{\Theta}(t))\mathbf{s}(t) + \mathbf{n}(t) \qquad t/\tau = 1, 2, \cdots, M \tag{51}$$

In (51), $\mathbf{y}(t)$ is the noisy array output vector, $\mathbf{n}(t)$ is additive noise (e.g., $\mathbf{n}(t) \sim \mathcal{N}(0, \sigma_n^2\mathbf{I})$), and $\mathbf{A}(\mathbf{\Theta}(t))$ has the steering vectors in its columns. It should be noted that the sensor positions must be perfectly known to define $\mathbf{A}(\mathbf{\Theta}(t))$ for this model [21].

For notational convenience and tractability, the data collected at each time increment τ during the batch period T is stacked to form the data vector $\mathbf{Y}_t : MP \times 1$. The signal vector $\mathbf{S}_t$ and the noise vector $\mathbf{W}_t$ are formed in the same manner. Thus, the array data

(or observation) model for the batch period can be compactly written as:

$$\begin{aligned} \mathbf{Y}_t &= \hat{\mathbf{h}}(\mathbf{\Theta}(t), \mathbf{W}_t) = \mathbf{h}(\mathbf{\Theta}(t)) + \mathbf{W}_t \\ \mathbf{Y}_t &= \mathbf{A}_t \mathbf{S}_t + \mathbf{W}_t \end{aligned} \tag{52}$$

where the steering matrix $\mathbf{A}_t = \text{diag}\{\mathbf{A}(\theta(t)), \mathbf{A}(\theta(t+\tau)), \cdots, \mathbf{A}(\theta(t+(M-1)\tau))\}$ implicity incorporates the DOA information of the targets.

2.2.4 Observability of the Motion Model

Even though the formulation of the problem as a state and observation model seems intuitive at first, it would be unwise to use this type of formulation if the new state model (41) is not observable given the measurements (52). For the array model, it is assumed that the array can resolve the target DOA parameters uniquely. This leaves only the examination of the motion model to determine the observability of the system [14,77]. It is verified in [77] that the state model described by (38) satisfies the strongly locally observability rank condition, which checks whether the determinant of the Jacobian of the state vector with respect to the target DOA is non-zero [69].

It follows that the new state vector $\mathbf{x}_k(t) \in \mathbb{R}^5$ is observable if $\{\theta_k(t + m\tau); m = 0, 1, \cdots, \Gamma_k - 1\}$ and $\{b_k(t); m = 0, 1, \cdots, \Gamma_k - 1\}$ are sufficient to determine $\mathbf{x}_k(t)$ for a finite integer Γ_k . The Jacobian of the state vector (41) with respect to the target DOAs is very tedious to calculate by hand. However, a symbolic algebra manipulator can be programmed to prove that the Jacobian matrix is full rank, making the new state vector observable.

2.3 PDF Constructions for the IPPF

The particle filter is a convenient way of recursively updating a target posterior of interest. In this section, we will show the derivations necessary for the operation of the particle filter, which are summarized in Fig. 13.

2.3.1 Data Likelihood

While formulating the state update equations in our problem, one encounters two nuisance parameters: the signal vector $\mathbf{S}_t$ and the noise variance for the additive Gaussian noise

vector $\mathbf{W}_t$. For simplicity, we will assume that the noise variance is approximately constant during the batch period $[t, t+T)$. Following the notation introduced in [60] , we will denote this noise variance as $\sigma_w^2(t)$ corresponding to the batch period starting at time t. The noise has a complex Gaussian probability density function (pdf) described by Goodman [35]. The data likelihood, given the signal and noise vectors, can be written as follows:

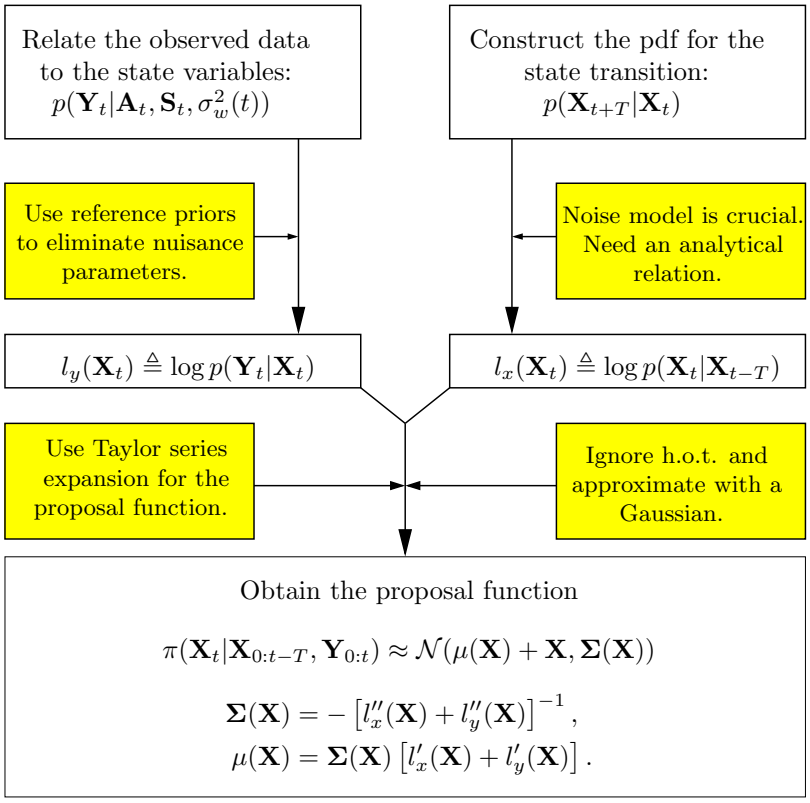


Figure 13: The mechanics of the necessary derivations needed by the particle filter. $\mathbf{X}$ is chosen to be the predicted state vector by the state update relation without any noise.

$$\begin{aligned} L &\triangleq \log p(\mathbf{Y}_t|\mathbf{A}_t, \mathbf{S}_t, \sigma_w^2(t)) \\ &= -MP\log\pi - MP\log\sigma_w^2(t) - \frac{1}{\sigma_w^2(t)}(\mathbf{Y}_t - \mathbf{A}_t\mathbf{S}_t)^H(\mathbf{Y}_t - \mathbf{A}_t\mathbf{S}_t) \end{aligned} \tag{53}$$

If the priors are known for the signals and for the noise variance given the state vector at time t, they can be integrated out (a procedure known as marginalization). If one desires to assume the least about these parameters and let the observed data speak for itself, then reference priors comes into play[2]. Even for moderate sample sizes, the information in the

[2]Bernardo derives the reference prior using an estimation model based on communication channel with

data dominates the *prior* information because of the vague nature of the prior knowledge [12]. The intuitive choice of the prior is usually the uniform prior on the *natural* space of the parameter. A good discussion of these issues can be found in [10, 12, 46].

The square root of the determinant of the Fisher information matrix is used as our reference prior (a.k.a. Jeffrey's prior). The resulting reference prior is not integrable (and hence, is improper) on the entire unbounded space for the parameter vectors. This, in turn, stipulates compactness arguments on the parameter space such as those used in [12], [67]. Assuming that the columns of $\mathbf{A}_t$ are linearly independent, the reference prior is given by [13]

$$p(\mathbf{S}_t|\mathbf{A}_t) \propto |\mathbf{A}_t^H \mathbf{A}_t|^{1/2}, \tag{54}$$

where $|.|$ denotes the determinant of a matrix. At this point, we can use (54) to integrate out the signal vector from our problem.

$$\begin{aligned} p(\mathbf{Y}_t|\mathbf{A}_t, \sigma_w^2(t)) &= \int p(\mathbf{Y}_t|\mathbf{A}_t, \mathbf{S}_t, \sigma_w^2(t)) p(\mathbf{S}_t|\mathbf{A}_t) d\mathbf{S}_t \\ \Rightarrow p(\mathbf{Y}_t|\mathbf{A}_t, \sigma_w^2(t)) &\propto \exp\left[-\frac{\mathbf{Y}_t^H (\mathbf{I} - \mathbf{A}_t(\mathbf{A}_t^H \mathbf{A}_t)^{-1} \mathbf{A}_t^H) \mathbf{Y}_t}{\sigma_w^2(t)} \right]. \end{aligned} \tag{55}$$

Finally, notice that (54) is not integrable if $\mathbf{S}_t$ has infinite multidimensional support. However, a bounded amplitude condition $\{\mathbf{S}_t : |[\mathbf{S}_t]_i| < \gamma_i\}$ can be easily imposed on the i^{th} signal component for some large γ_i. This makes the prior (54) integrable on the signal vector space and, in turn, the marginalization integrals become proper. This condition is always satisfied in practice because the signals of interest must have finite magnitudes at all times.

2.3.2 State Likelihood

In the previous section, we omitted the motivation for constructing the pdf for the data and put the emphasis on the use of reference priors. Now, it is necessary to elaborate on the reasons for constructing the pdfs for the data and the state. The state and observation models (38), (41) and (52) form a hidden Markov model (HMM), which can be compactly

a source and data [11]. The reference prior maximizes the mutual information between the source and the data.

described by the following pdfs:

$$\begin{aligned} &p(\mathbf{X}_t|\mathbf{X}_{t-T}) \\ &p(\mathbf{Y}_t|\mathbf{X}_t) \end{aligned} \tag{56}$$

where $\mathbf{X}_t = [\mathbf{x}_1^T(t), \mathbf{x}_2^T(t), \cdots, \mathbf{x}_K^T(t)]^T$, and $p(\mathbf{Y}_t|\mathbf{X}_t) = p(\mathbf{Y}_t|\mathbf{A}_t)$ or $p(\mathbf{Y}_t|\mathbf{A}_t, \sigma_w^2(t))$, depending upon whether or not we treat the noise variance as a known parameter. Here, we introduce a common notation used in the particle filtering literature, $\mathbf{z}_{0:t} \triangleq \{\mathbf{z}_0, \mathbf{z}_T, \ldots, \mathbf{z}_t\}$. The recursive update for the HMM model described by (56) can be written as follows [5]:

$$p(\mathbf{X}_{0:t}|\mathbf{Y}_{0:t}) = p(\mathbf{X}_{0:t-T}|\mathbf{Y}_{0:t-T}) \frac{p(\mathbf{Y}_t|\mathbf{X}_t)p(\mathbf{X}_t|\mathbf{X}_{t-T})}{p(\mathbf{Y}_t|\mathbf{Y}_{0:t-T})} \tag{57}$$

Hence, the recursive evaluation of $p(\mathbf{X}_{0:t}|\mathbf{Y}_{0:t})$ requires the pdfs shown in (56). The previous section considered the construction of the second pdf in the model. This section will concentrate on the first pdf in (56).

The objective is to find $p(\mathbf{X}_t|\mathbf{X}_{t-T})$ given the state model. By inspection of (38) or (41), one can see that $\mathbf{X}_t$ is also normal with mean $\mathbf{f}_{\mathbf{I},\mathbf{II}}(\mathbf{X}_{t-T})$ and covariance equal to that of the additive noise. Therefore, we can write the pdf for the state update as follows:

$$p(\mathbf{X}_t|\mathbf{X}_{t-T}) \sim \mathcal{N}(\mathbf{f}_{\mathbf{I},\mathbf{II}}(\mathbf{X}_{t-T}), \text{diag}\{\sigma_\theta^2, \sigma_Q^2, \sigma_\Psi^2, \sigma_\phi^2, \sigma_f^2\}) \tag{58}$$

We have two important remarks on the construction of the pdfs for our problem. The first is that we generally need analytical expressions for the pdfs to use the particle filter, which does not assume a Gaussian model in general. The second remark is about the model order of the HMM. The motion equations describe a first order HMM model and hence the update equations (57) depend only on the previous state. If more complicated motion equations are formulated in the state model so that the HMM model order increases, then a new recursive update formulation becomes necessary.

2.3.3 The proposal function

An appropriate choice of the proposal function $\pi(\cdot)$ may reduce the variance of the simulation errors[3]. However, it was shown analytically in [25] that the importance weights

[3]If we choose the exact posterior as the proposal function, then due to the nature of the data generating process, the variance of the estimator is inversely proportional to the number of particles N [1].

have increasing variance with time, which leads to increasing estimation errors (the term "simulation errors," will be used interchangeably). Here, we restate an important result: the unconditional variance of the importance weights, i.e., with the observations $\mathbf{Y}_{0:t}$ being interpreted as random variables, increases over time. This fact is known as the degeneracy phenomenon: after a few iterations, all but one of the normalized importance weights will be very close to zero [25].

It is crucial to note that the optimal proposal function $\pi(\mathbf{X}_t|\mathbf{X}^{(i)}_{0:t-T}, \mathbf{Y}_{0:t})$ is proportional to $p(\mathbf{Y}_t|\mathbf{X}_t) \times p(\mathbf{X}_t|\mathbf{X}^{(i)}_{t-T})$, with the proportionality independent of $\mathbf{X}_t$. We have previously derived analytical relations for $p(\mathbf{Y}_t|\mathbf{X}_t)$ and $p(\mathbf{X}_t|\mathbf{X}_{t-T})$. Moreover, define

$$\begin{aligned} l_y(\mathbf{X}_t) &\triangleq \log p(\mathbf{Y}_t|\mathbf{X}_t) \\ l_x(\mathbf{X}_t) &\triangleq \log p(\mathbf{X}_t|\mathbf{X}_{t-T}) \end{aligned} \tag{59}$$

$$\begin{aligned} \mathbf{\Sigma}(\mathbf{X}) &= -[l''_x(\mathbf{X}) + l''_y(\mathbf{X})]^{-1} \\ \mu(\mathbf{X}) &= \mathbf{\Sigma}(\mathbf{X})[l'_x(\mathbf{X}) + l'_y(\mathbf{X})] \end{aligned} \tag{60}$$

then a suboptimal proposal function that minimizes the variance of the importance weights is given by the following [25, 60]:

$$\pi(\mathbf{X}_t|\mathbf{X}_{0:t-T}, \mathbf{Y}_{0:t}) \approx \mathcal{N}(\mu(\mathbf{X}) + \mathbf{X}, \mathbf{\Sigma}(\mathbf{X})) \tag{61}$$

where $\mathbf{X}$ is judiciously chosen to be the mode of $p(\mathbf{X}_t|\mathbf{X}_{t-T}, \mathbf{Y}_t)$ so that $\mu(\mathbf{X}) \approx \mathbf{0}$ [25].

2.4 Algorithm Details

In this section, we will give the details of our modifications to the independent partition particle filtering algorithm by Orton and Fitzgerald [60]. The outline of the IPPF is given in [60] and repeated for completeness in Table 3.

2.4.1 Partitioning and Data Association

Each particle in the IPPF consists of multiple state vectors, e.g., for three targets, a particle consists of three different state vectors corresponding to each target's motion parameters and frequency. The target association problem is solved by the independence assumption on these partitions. To elaborate, the independence assumption results in separate partition

Table 3: Pseudo Code for the IPPF

---

i. At time t, for each particle i $(i = 1, \ldots, N)$ and its partition k $(k = 1, \ldots, K)$, $\mathbf{x}_t^{(i)}(k)$:

- sample from the partition proposal function: $\mathbf{x} \sim \pi_k(\mathbf{x}_t(k)|\mathbf{x}_{t-T}^{(i)}(k), \mathbf{Y}_t)$,
- set $\mathbf{x}_t^{(i)}(k) = \mathbf{x}$ and calculate the partition weight $q_k^{(i)}(\mathbf{x})$,
- normalize the partition weights across all particles for each partition:

$$q_k^{(i)} = \frac{q_k^{(i)}}{\sum_i q_k^{(i)}},$$

- for each partition, resample with replacement across the particles using the distribution generated by $q_k^{(i)}$ and generate a new set of particles $\mathbf{X}_t^{(i)}$ with their respective $q_k^{(i)}$ for each partition. Also, reindex $\mathbf{X}_{t-T}^{(i)}$ accordingly.

ii. For each particle i $(i = 1, \ldots, N)$, $\mathbf{X}_t^{(i)}$:

- calculate the importance weights using

$$w_t^{(i)} = w_{t-T}^{(i)} \frac{p(\mathbf{Y}_t|\mathbf{X}_t^{(i)}) p(\mathbf{X}_t^{(i)}|\mathbf{X}_{t-T}^{(i)})}{\pi(\mathbf{X}_t|\mathbf{X}_{t-T}^{(i)}, \mathbf{Y}_t) \prod_k q_k^{(i)}},$$

- normalize the importance weights $w_t^{(i)}$ across the particles.

iii. Resample the particles $\mathbf{X}_t^{(i)}$ using a Metropolis-Hastings scheme, keeping the reversibility of the chain.

---

proposal functions $\pi_k(.)$. To generate the partition proposal functions, (60) is calculated for the whole particle; however, only the block diagonal portions of $\mathbf{\Sigma}(\mathbf{X})$ are used for each partition, which results in $\pi_k(\mathbf{x}) \approx \mathcal{N}(\mu(\mathbf{x}) + \mathbf{x}, \mathbf{\Sigma}_k(\mathbf{X}))$, with $\mathbf{\Sigma}_k(\mathbf{X})$ corresponding to the k^{th} $(dim\{\mathbf{x}\} \times dim\{\mathbf{x}\})$ block diagonal matrix entry of $\mathbf{\Sigma}(\mathbf{X})$. In particular, the off-diagonal matrices in the particle Hessian corresponding to the cross partitions are ignored. Note that after a particle is formed, the discrepancies generated by this method are alleviated by the importance weights, which are calculated using the full Hessians generated from the new particle.

When the target DOAs cross, previous target states help distinguish the next state through the state update probability. The important thing to remember is that unless one target is moving in tandem with another target, the partitions for the two targets will differ from each other in the other elements of the state vector (e.g, frequency, heading direction, and so on), which will be emphasized by the partition probability $q_k(\mathbf{x})$. Then, the partition cross sampling is used to help the data association by generating particles having high probability across all partitions. This method of generating particles helps the IPPF to propagate particles with good predictive states and automatically handles the data association.

One modification is to use the state transition probability (58) for the weighted resampling functions $q_k(\mathbf{x})$. This choice alone seems to constrain the particles by the state update equation, and hence is expected to have poor performance for maneuvering targets. However, this choice of the weighted resampling function makes sure that the created particles form a cloud around the expected mode of the target state. The maneuvering target cases, on the other hand, are handled by the critical Monté-Carlo Markov chain (MCMC) resampling step. A classical Metropolis-Hastings scheme, which keeps the reversibility of the Markov chain, can be used to resample the state space using the data likelihood. In the test cases run, the algorithm seems to handle maneuvers better as the number of iterations increases in the MCMC step. Details of an MCMC resampling scheme are given in [59].

When the targets maneuver, the expected mode of the next state predicted by the state update equation changes. At the resampling stage, the particles that are closer to this

changed mean survive, while the particles around the predicted mean diminish. Hence, the resampling step, in effect, not only makes the particles span most of the state space, but also compensates for the effects of the maneuver. Note that maneuvering has more impact on the heading direction than the other state variables. Hence, a slight modification exploiting this fact in the resampling step may also improve the performance of the algorithm for a fixed number of particles.

2.4.2 Effects of the Frequency Variable

The new state vectors include new motion variables, but the most important extension comes from the frequency variable, which requires the derivation of new gradients and Hessians (60) for the linearization of the optimal proposal function. We will concentrate on l'_y and l''_y, since it is necessary to approximate l''_y by a positive definite matrix[4], and l'_y is used in setting up the equations. Derivations of l'_x and l''_x based on the new motion parameters are straightforward. The notation in this section closely follows [77]. Define

$$\begin{aligned} J(t) &\triangleq \mathbf{Y}_t^H(\mathbf{I} - \mathbf{A}_t(\mathbf{A}_t^H\mathbf{A}_t)^{-1}\mathbf{A}_t^H)\mathbf{Y}_t \\ &= \sum_{m=0}^{M-1} |\mathbf{y}(t+m\tau) - \mathbf{P}_A(t+m\tau)\mathbf{y}(t+m\tau)|^2 \\ &= \sum_{m=0}^{M-1} \mathrm{trace}\{\mathbf{P}_A^{\perp}(t+m\tau)\hat{\mathbf{R}}_y(t+m\tau)\} = \sum_{m=0}^{M-1} J_m(t) \end{aligned} \tag{62}$$

where $\mathbf{P}_A$ and $\mathbf{P}_A^{\perp} \triangleq \mathbf{I} - \mathbf{P}_A$ are the projection matrices onto the column and the null spaces of $\mathbf{A}$ and $\mathbf{A}^H$, respectively. $\hat{\mathbf{R}}_y(t+m\tau) = \mathbf{y}(t+m\tau)\mathbf{y}^H(t+m\tau)$ is the one-sample estimate of the array covariance matrix at the batch time indexed by m. Note that $l_y = -MJ/\sigma_w^2$; hence, the gradients and the Hessians of $J(t)$ are linearly related to l'_y and l''_y. Defining the gradient of $J(t)$ as G, we obtain

$$G \triangleq \frac{\partial J(t)}{\partial \mathbf{X}_t} = \mathrm{vec}\left\{ \frac{1}{M} \sum_{m=0}^{M-1} \Big[V_m(t)\mathrm{diag}(\nabla_\theta J_m) + \Xi_m(t)\mathrm{diag}(\nabla_f J_m) \Big] \right\} \tag{63}$$

[4]The parameter l''_y represents the local covariance of the particles around their modes and is required to be positive definite; however, this positive definiteness is not always available, and modifications are required to guarantee that l''_y remains positive definite at each iteration of the filter. See [77] for more discussion.

where

$$\begin{aligned} V_m(t) &\triangleq V(t+m\tau) = \Big[\, \mathbf{v}_1(t+m\tau),\, \mathbf{v}_2(t+m\tau),\, \cdots,\, \mathbf{v}_K(t+m\tau) \,\Big] \\ \Xi_m(t) &\triangleq \Xi(t+m\tau) = \Big[\, \xi_1(t+m\tau),\, \xi_2(t+m\tau),\, \cdots,\, \xi_K(t+m\tau) \,\Big] \end{aligned} \tag{64}$$

with $\mathbf{v}_k(t+m\tau) \triangleq \frac{\partial \theta_k(t+m\tau)}{\partial \mathbf{x}_k(t)}$ and $\xi_k(t+m\tau) \triangleq \frac{\partial f_k(t+m\tau)}{\partial \mathbf{x}_k(t)}$. Moreover,

$$\begin{aligned} \nabla_\theta J_m &= \Big[\, \partial J_m/\partial\theta_1(t+m\tau),\, \partial J_m/\partial\theta_2(t+m\tau),\, \cdots,\, \partial J_m/\partial\theta_K(t+m\tau) \,\Big] \\ \nabla_f J_m &= \Big[\, \partial J_m/\partial f_1(t+m\tau),\, \partial J_m/\partial f_2(t+m\tau),\, \cdots,\, \partial J_m/\partial f_K(t+m\tau) \,\Big] \end{aligned} \tag{65}$$

Equation (63) follows from the chain rule, where the target frequency and its DOA are assumed independent from each other. Define $\Lambda_{i,m}(t)$ and $\Upsilon_{i,m}(t)$ as the Hessian of $\theta_i(t+m\tau)$ and $f_i(t+m\tau)$ with respect to $\mathbf{X}_t$, respectively; and form $\Lambda_m(t) = \mathrm{diag}[\Lambda_{1,m}(t), \ldots, \Lambda_{K,m}(t)]$ and $\Upsilon_m(t) = \mathrm{diag}[\Upsilon_{1,m}(t), \ldots, \Upsilon_{K,m}(t)]$. Then, the Hessian $H \triangleq \frac{\partial^2 J(t)}{\partial \mathbf{X}_t \partial \mathbf{X}_t}$ is given by

$$H = H_{\theta\theta} + H_{ff} + H_{\theta f} \tag{66}$$

where

$$H_{\theta\theta} = \frac{1}{M}\sum_{m=0}^{M-1} \Big\{ [\nabla^2_{\theta\theta} J_m \otimes 1] \odot [\mathrm{vec}V_m(t)\mathrm{vec}^H V_m(t)] + [\mathrm{diag}(\nabla_\theta J_m \otimes I)] \odot \Lambda_m(t) \Big\} \tag{67}$$

$$H_{ff} = \frac{1}{M}\sum_{m=0}^{M-1} \Big\{ [\nabla^2_{ff} J_m \otimes 1] \odot [\mathrm{vec}\Xi_m(t)\mathrm{vec}^H \Xi_m(t)] + [\mathrm{diag}(\nabla_f J_m \otimes I)] \odot \Upsilon_m(t) \Big\} \tag{68}$$

$$H_{\theta f} = \frac{1}{M}\sum_{m=0}^{M-1} \Big\{ [\nabla^2_{\theta f} J_m \otimes 1] \odot [\mathrm{vec}V_m(t)\mathrm{vec}^H \Xi_m(t) + \mathrm{vec}\Xi_m(t)\mathrm{vec}^H V_m(t)] \Big\} \tag{69}$$

The operator vec stands for the concatenation of the columns of a matrix; $\otimes$ and $\odot$ denote the Kronecker and Schur products, respectively. In addition, $\mathbf{1}$ denotes a matrix of all ones and I the identity matrix, both being $\dim\{\mathbf{x}\} \times \dim\{\mathbf{x}\}$.

To guarantee the positive definiteness of l''_y, the terms containing $\Lambda_m(t)$ and $\Upsilon_m(t)$ on (67) and (68) are ignored while calculating the Hessian in (66) as discussed in [77]. Define $A_i(t+m\tau) = \partial A(t+m\tau)/\partial\theta_i(t)$, $C_m(t+m\tau) = \partial A(t+m\tau)/\partial f_m(t)$, and $\gamma_i(t+m\tau) = \frac{\partial \mathbf{P}^\perp_A(t+m\tau)\mathbf{X}_{t+m\tau}}{\partial\theta_i(t)}$. The following derivatives need to be approximated while calculating (67)

and (68):

$$\frac{\partial^2 J_m}{\partial\theta_i(t)\partial\theta_j(t)} \simeq 2\Re\{\gamma_i^H(t+m\tau)\gamma_j^H(t+m\tau)\}$$
$$\frac{\partial^2 J_m}{\partial f_i(t)\partial f_j(t)} \simeq 2\Re\{\mathrm{tr}[A^{\dagger H}(t+m\tau)C_i^H(t+m\tau)P_A^{\perp}(t+m\tau)C_j^H(t+m\tau)A^{\dagger H}(t+m\tau)]\} \tag{70}$$

$H_{\theta f}$ is not guaranteed to be negative definite and can be ignored; however, the authors found that with the approximation below, $H_{\theta f}$ almost never affects the definiteness of the Hessian and can be used in the calculation of the Hessian (66):

$$\frac{\partial^2 J_m}{\partial\theta_i(t)\partial f_i(t)} \simeq 2\Re\{\mathrm{tr}[A^{\dagger H}(t+m\tau)A_i^H(t+m\tau)P_A^{\perp}(t+m\tau)C_j^H(t+m\tau)A^{\dagger H}(t+m\tau)]\}. \tag{71}$$

2.5 Simulation Results

In the simulations, our objectives are (i) compare the effectiveness of the two state formulations for tracking targets; (ii) show the effect of the frequency variable on tracking. Comparison of State Model-I with the extended Kalman filter also can be found in [60] and, hence, is not repeated here. Issues related to initialization of the filters can be found in [59].

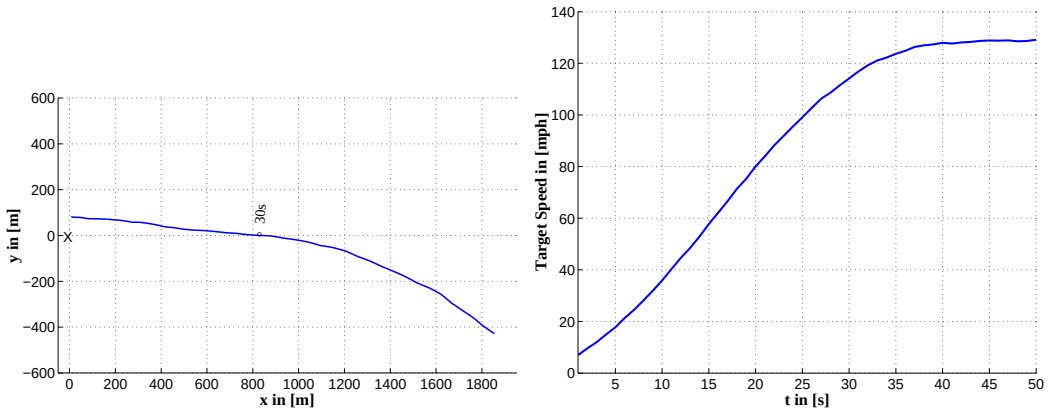
2.5.1 Single Target Tracking

A circular sensor array of 15 omnidirectional sensors is used to track a single target. The radius of the array is such that the inter-element spacing is equal to 0.45 times the wavelength (λ) of the target of interest. Figure 14 shows the track and the temporal speed evolution of the target. The simulation parameters are given in Table 4.

Table 4: Simulation Parameters (A)

Number of Particles, N	100
θ noise σ_θ	0.1°
Q noise σ_Q	0.1
ψ noise σ_ψ	0.0001
ϕ noise σ_ϕ	4°
Signal to Noise Ratio, SNR	7dB
Target Narrow-band Frequency, f_0	200Hz
Number of Batch Samples, M	8

In Table 4, some explanation of M, the number of batch samples, is necessary. If the DOA tracking is done by a snapshot algorithm, a much higher number of batch samples



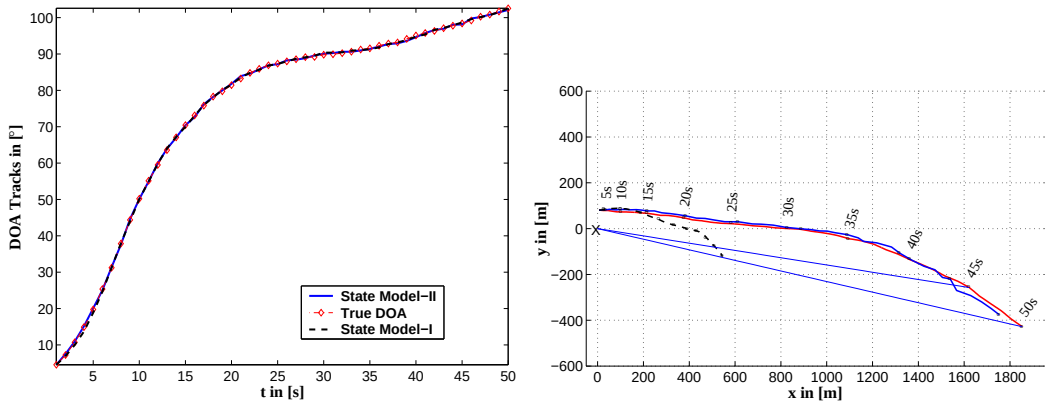
(a) (b)

Figure 14: (a) One target initially heading in the positive x-direction with speed approximately equal to 7 mph, starts to maneuver at $t = 30$s. The sensor node is situated at the origin. (b) The target has almost constant acceleration between $t = 0$ and $t = 35$. For $t > 35$, the speed of the target is almost constant during the maneuver.

would be necessary to get high resolution DOAs. However, inclusion of the motion dynamics reduces the number of batch samples required to estimate the target DOAs. Ideally, a higher number of batch samples also helps the IPPF estimate the DOAs better since the gradient and Hessian terms that form the mean and covariance matrix of the approximate proposal function incorporate more data, and hence are expected to improve. Interestingly, using synthetic data, the authors determined that even $M = 2$ makes a good approximation to these parameters when the target accelerations are small and the algorithm is initialized close to the true values. Additional improvement in DOA estimation performance by increasing M is empirically found to quickly reach a point of diminishing returns. When State Models I and II are run for the same target track in Fig. 14 with $M = 2$, both models perform the same. However, as M is increased, State Model-II starts to perform better.

Figure 15 illustrates the estimation performance of the IPPF. The DOA estimation of both filters is almost identical as shown in Fig. 15(a). However, if the true target tracks were estimated using the IPPF state estimates in conjunction with the correct target initial range and speeds (which are *not* available to the IPPF), the resulting tracks would be quite different, as illustrated in Fig. 15(b). Even if the DOAs are the same, State Model-I explains the increase in target speed with a change in the heading direction, but leads to an incorrect

heading direction estimate, which is a crucial parameter in data fusion. Figure 16 shows the estimated ϕ and Q parameters of the target. As one can observe, different values of these parameters can lead to the same DOA track even if they do not correspond to the true physical target track.



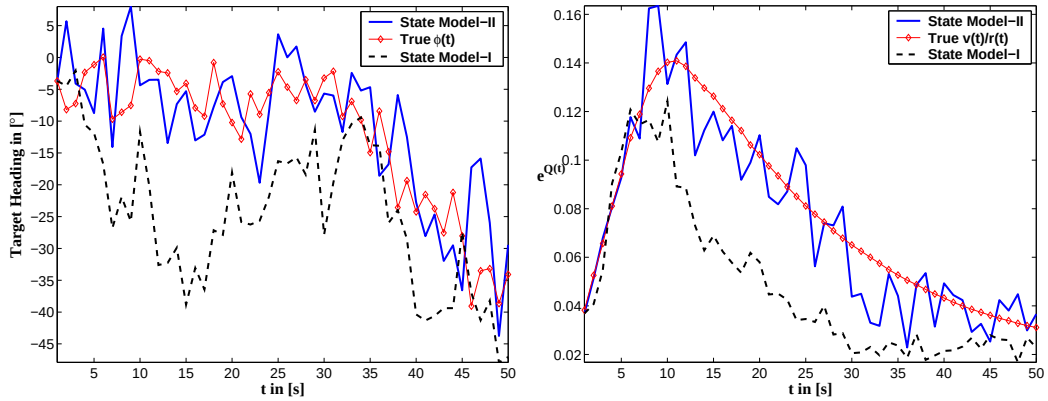
(a) (b)

Figure 15: (a) The diamonds are the true DOA track. The dashed line is the estimate from State Model-I while the solid line is the estimate from State Model-II. The two estimates are nearly identical in tracking the target DOAs. (b) Small errors in the DOAs are accentuated by the range. State Model-II obtains a good estimate of the true target track because it uses acceleration.

2.5.2 Multiple Target Tracking with Varying Narrow-band Frequencies

It is challenging to track two narrow-band targets whose DOA tracks are closer than the Rayleigh resolution. The IPPF produces high-resolution DOA estimates; however, it can fail when the target DOAs as well as the movement parameters are very close. The objective in this section is to show that by incorporating a frequency variable into the state model, it is sometimes possible to still track targets even in this difficult case. In this example, three target tracks are simulated with different frequency scenarios (Fig. 17). Table 5 summarizes the simulation parameters. Figure 18 shows the tracking performance when the targets have the same time-frequency signatures. Targets marked with the diamonds (1) and the circles (2) have very similar DOA tracks after time $t = 15s$. Initially, the IPPF does a good job in tracking; however, as targets get closer, it fails.

Figure 19 simulates the same problem, but, in this case, targets 1 and 2 have different time-frequency signatures. Note that the IPPF had problems resolving the DOAs of these



(a) (b)

Figure 16: (a) State Model-I tries to explain target accelerations through the heading parameter; hence, it fails to capture the true target parameters. The true target heading direction has additive Gaussian noise with $\sigma_\phi = 4^\circ$, which is marked with diamonds. The solid line is the State Model-II estimate. (b) Note that $e^{Q(t)}$ corresponds to $v(t)/r(t)$ for the target. State Model-II has a better $Q(t)$ estimate since the target headings are close to the true headings.

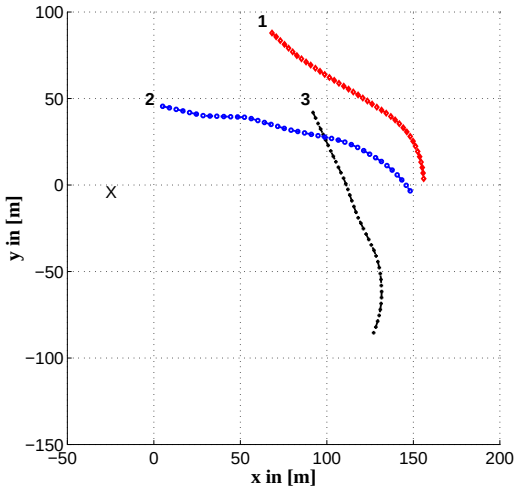
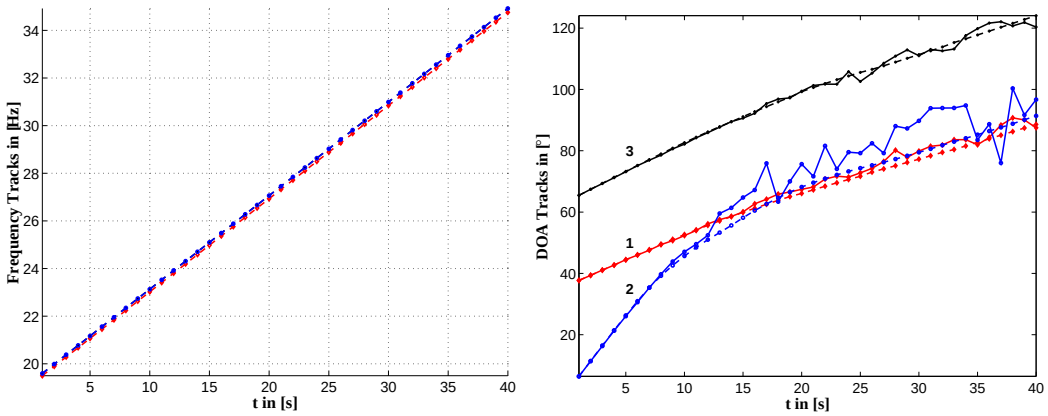


Figure 17: Three targets move with constant speed but with some Brownian disturbance acting on their heading directions. State Model-I is sufficient for the tracking in this case.

Table 5: Simulation Parameters (B)

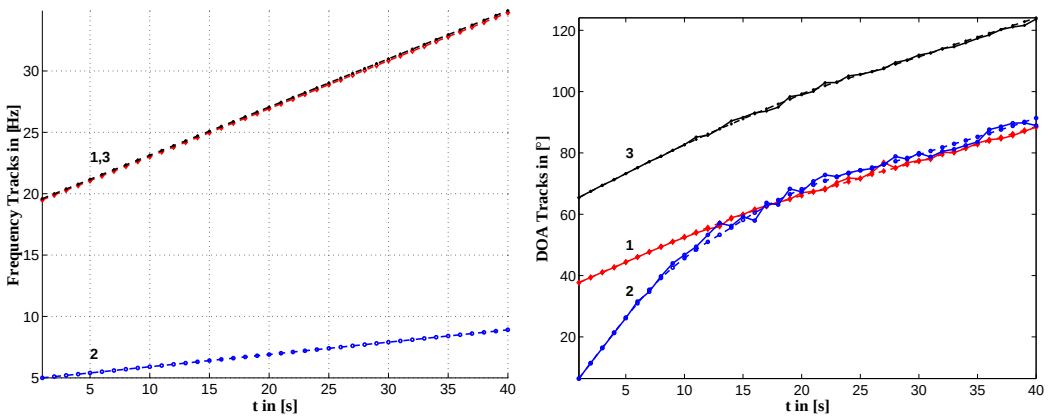
Number of Particles, N	100
θ noise σ_θ	0.1°
Q noise σ_Q	0.01
ϕ noise σ_ϕ	4°
Frequency noise σ_f	0.001
Signal to Noise Ratio, SNR	7dB
Number of Batch Samples, M	8

targets since they had close motion parameters and had the same time-frequency signature. Due to the way the particles are generated (by the partition proposal functions $\pi_k(.)$ as explained in Sect. 2.4), the DOA tracking of target 3 is also affected as shown in Fig. 18. Hence, the change in the time-frequency signature of target 2 alleviates the independence assumption on the partitions and improves the DOA tracking performance of the filter as illustrated in Fig. 19 (b). Moreover, the partition weights $q_k(\mathbf{x})$ for each partition not only depend on the motion parameters, but also the frequency. Hence, when the time-frequency signatures of the targets differ, the IPPF can create better predictive states, resulting in better tracking overall.



(a) (b)

Figure 18: (a) Initial target frequencies are 19.50Hz, 19.60Hz, and 19.60Hz. (b) As the targets 1 and 2 get close to each other, their motion parameters are not sufficient to distinguish their DOAs; hence, the IPPF's DOA tracking performance deteriorates.



(a) (b)

Figure 19: (a) Only the frequency track corresponding to the target marked with circles is changed from the previous example. (b) The tracking is now improved due to the difference in the frequency of the targets that move close to each other.

2.6 Conclusions

The state and observation equation set is the heart of any tracking model. In this chapter, two new state models and a new observation model were demonstrated using the IPPF. Given an observable state update, it is relatively easy to generate the IPPF equations automatically for an acoustic observation model that can resolve target DOAs. When all the targets have either zero acceleration or small accelerations, both state models (I and II) have nearly identical tracking performance. Since the computational complexity of state model II is about twice that of state model I, this trade-off favors model I. In one of the examples shown, two targets with the same DOA track have quite different tracks in x-y space depending on the acceleration model. The ghost track estimated by model I, due to the incorrect heading direction estimate, would be undesirable when doing data association with multiple nodes; hence, model II is preferred since it included acceleration in its state.

A derivation for the array steering matrix in the case where the target signal phases are locally quadratic was presented. In addition, it was demonstrated that if the state is also augmented to include the frequency of individual targets, then it is possible to track targets with similar motion parameters when the targets have different time-frequency signatures. This is an important result, since it enables acoustic trackers using these new state models to track multiple targets moving close to each other, such as in a convoy.

CHAPTER III

AN ACOUSTIC MULTIPLE TARGET TRACKER

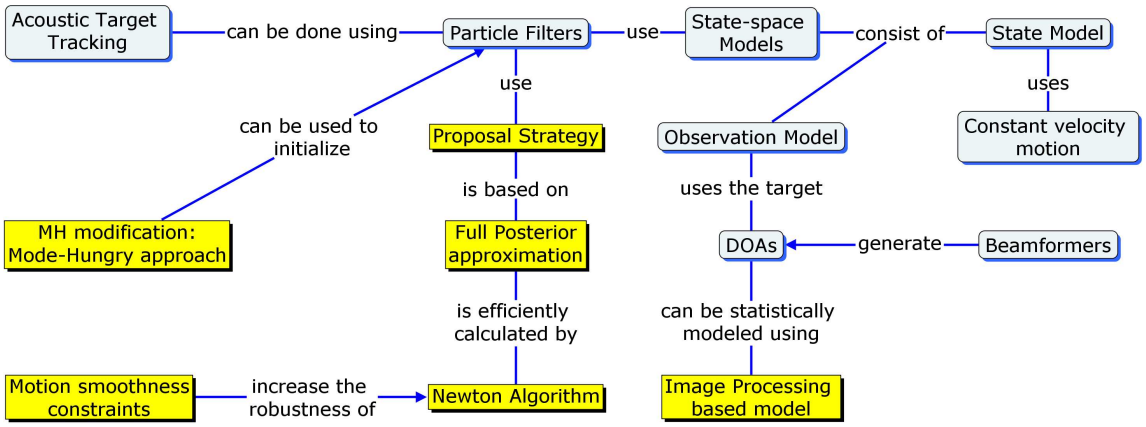


Figure 20: Chapter outline with the new contributions shown in rectangular boxes.

3.1 Introduction

Tracking the bearing angles of multiple maneuvering targets using acoustic arrays is a classic problem with many challenging aspects. In the literature, the problem is usually formulated in terms of state-space models where target direction-of-arrivals (DOA) are related to the acoustic microphone outputs through an observation equation, and the updates on the state, which may include variables other than DOA alone, are explained by locally linear motion models [18, 60, 77]. The performance of these algorithms relies heavily on how accurately these models represent the observed natural phenomena.

One important observation model is the far-field narrow-band observation model, where the array response is related to a source with a constant narrow-band frequency at a DOA, θ, through some steering vectors under the assumptions of an isotropic and non-dispersive medium [42]. This model possesses a well-understood mathematical structure and has had a significant impact on the DOA estimation problem because it has led to many practical DOA estimation techniques (e.g., MUSIC, MVDR, eigenvalue beamformers, etc.). However,

the estimation performance of algorithms based this model deteriorates when the target signals are wide-band, or when rapid target motion spreads the array spatial spectrum. The complexity of this observation model increases significantly as some of the assumptions in the model are relaxed to more accurately represent reality, rendering the tracking algorithms analytically intractable.

The presence of multiple targets increases tracking complexity even more because of data association issues, where a mechanism is needed, in effect, to sort the received data for each target. The association problem could be handled in several different ways: (i) probabilistic data association methods estimate states by summing over all the association hypotheses weighted by the probabilities obtained from the likelihood [7], (ii) smoothness assumptions on the target (motion) states allow a natural ordering of the data [45], (iii) computationally costly ML/EM methods use likelihood functions to search for a global maximum, and (iv) nearest neighbor methods provide easy heuristics to perform measurement updates. Most of these methods use the mean and the covariance approximation on the sufficient statistics for the state, which may be estimated with a Kalman filter. For nonlinear state-spaces with non-Gaussian noise assumptions, Monté-Carlo methods should be used to adequately capture dynamic, possibly multimodal, statistics.

A particle filter naturally accommodates solutions for state-space problems where the observations arrive in sequence. The state probability density function is represented by discrete state samples (particles) distributed according to the underlying distribution (as explained by the state-space), either directly or by proper weighting [48]. Hence, the filter can approximate any statistics of the distribution arbitrarily accurately by increasing the number of particles. This convergence result has been proven for the general case. In the particle filtering framework, the data association problem is undertaken implicitly by the state-space model interaction. However, to increase the efficiency of the algorithm, various methods have been proposed, such as the partitioning approach covered in the previous chapter, or other Bayesian approaches [41].

In the previous chapter, we have shown a particle filter tracker that tracked multiple maneuvering targets using various observation/state-update models and the partitioning

approach. In the filter mechanics, the data-likelihood functions were derived in terms of the acoustic microphone outputs based on the discussed observation models. In this chapter, on the other hand, instead of directly using the signals as explained by an observation model, a particle filtering algorithm is introduced that uses a sufficient statistics of the state vector, namely a batch of DOA estimates $\mathbf{y}_{t,f} = \{y_{t+m\tau,f}(j_m)\}_{m=0}^{M-1}$ from a beamformer appropriate for the local frequency characteristics of the target signals.

If the observed acoustic data deviates from the observation models introduced in the previous chapter, the performance of the particle filter built on these models also degrades. The front-end beamformers grant the new particle filter the ability to be independent of the observation model. This is because when the acoustic signals are better explained by different observation model, a different beamformer matching the observations can be used. Therefore, the filter input does not change. In this case, because the filter uses a sufficient statistic as opposed to the full acoustic data, overall computation is also decreased as compared to the particle filter in the previous chapter.

The new filter state, similar to the previous chapter, consists of the DOA $\theta_k(t)$, the heading direction $\phi_k(t)$, and the logarithm of velocity over range $Q_k(t) = \log v_k/r_k(t)$ for each target k, where the total number of targets K is assumed known. The state is reported at regular time intervals of $T = M\tau$ seconds. The batch DOAs may also depend on frequency f (e.g., DOAs calculated at different narrow-band frequencies), which is suitable in specific tracking scenarios. The number of DOA estimates j_m at time $t + m\tau$ is also modelled as time-dependent.

The observations are assumed to be normally distributed around the true DOA tracks with a constant missing data probability and clutter rate. To increase the efficiency of the algorithm, the filter uses an approximation of the full posterior to generate particles and a robust Newton search method to approximate the data-likelihood. Data association is handled automatically by the particle filter by imposing smoothness constraints on the target motion. Lastly, all DOA observations within the batch are assumed locally stationary to avoid angle spread due to target motion since the batch sampling period τ is assumed small.

The organization of this chapter is as follows. Section 3.2 gives a brief overview of the motion model and provides the needed data-likelihood expressions. The particle filter details are covered in Sect. 3.3 where a constrained Newton method is introduced to approximate the filter proposal function. Finally, a representative set of simulation results is given in Sect. 3.4.

3.2 State-Space Formulation

The particle filter state vector $\mathbf{x}_k(t)$ consists of the concatenation of the individual motion vectors $x_k(t) \triangleq [\ \theta_k(t)\ ,\ Q_k(t)\ ,\ \phi_k(t)\ ]^T$, as partitions, into the state vector $\mathbf{x}_t = [\ x_1^T(t),\ x_2^T(t),\ \ldots,\ x_K^T(t)\ ]^T$. The parameters $\theta_k(t)$ and $\phi_k(t)$ are measured counterclockwise with respect to the x-axis. The nonlinear state update equation can be derived from the geometry imposed by the assumed constant velocity motion [18, 77]:

$$x_k(t+T) = h_T\left(x_k(t)\right) + u_k(t) \tag{72}$$

where $u_k(t) \sim \mathcal{N}\left(0, \Sigma_u\right)$ with $\Sigma_u = \mathrm{diag}\{\sigma^2_{\theta,k}, \sigma^2_{q,k}, \sigma^2_{\phi,k}\}$ and $h_T(x_k(t)) =$

$$\begin{bmatrix} \tan^{-1}\left\{\frac{\sin\theta_k(t)+T\exp Q_k(t)\sin\phi_k(t)}{\cos\theta_k(t)+T\exp Q_k(t)\cos\phi_k(t)}\right\} \\ Q_k(t) - \frac{1}{2}\log\left\{1 + 2T\exp Q_k(t)\cos(\theta_k(t)-\phi_k(t)) + T^2\exp 2Q_k(t))\right\} \\ \phi_k(t) \end{bmatrix} \tag{73}$$

The observations $\mathbf{y}_{t,f} = \{y_{t+m\tau,f}(j_m)\}_{m=0}^{M-1}$ consist of all the DOA estimates produced by a beamformer at each batch index m. Hence, the acoustic data of duration T is segmented into M segments of duration τ. These segments are processed by a proper beamformer, based on the temporal frequency structure of the signals, to calculate possible DOA estimates. This procedure may be repeated F times, once for each narrow-band signal, for each frequency indexed by f. Note that only the peak locations are kept in the beamformer power pattern. Moreover, the peak values indexed by j_m need not be ordered or associated with the previous index $m-1$ and the number of peaks to retain can even be time-dependent.

Figure 21 illustrates the observation model. It is assumed that the batch of DOAs, $\mathbf{y}_{t,f}$, form a normally distributed cloud around the true target DOA tracks with a constant

missing data probability κ[1], and may have spurious peaks Poisson distributed with rate λ. The variance of the DOAs, σ^2, is assumed constant and may be estimated using the DOAs from the previous estimation period in conjunction with the understanding the estimation techniques the specific beamformer uses. For example, for an ML estimator using the narrow-band model, there is a formula that calculates the additive array noise variance given in [77]. The array noise can then be related to an expected DOA noise through the formulas in [71].

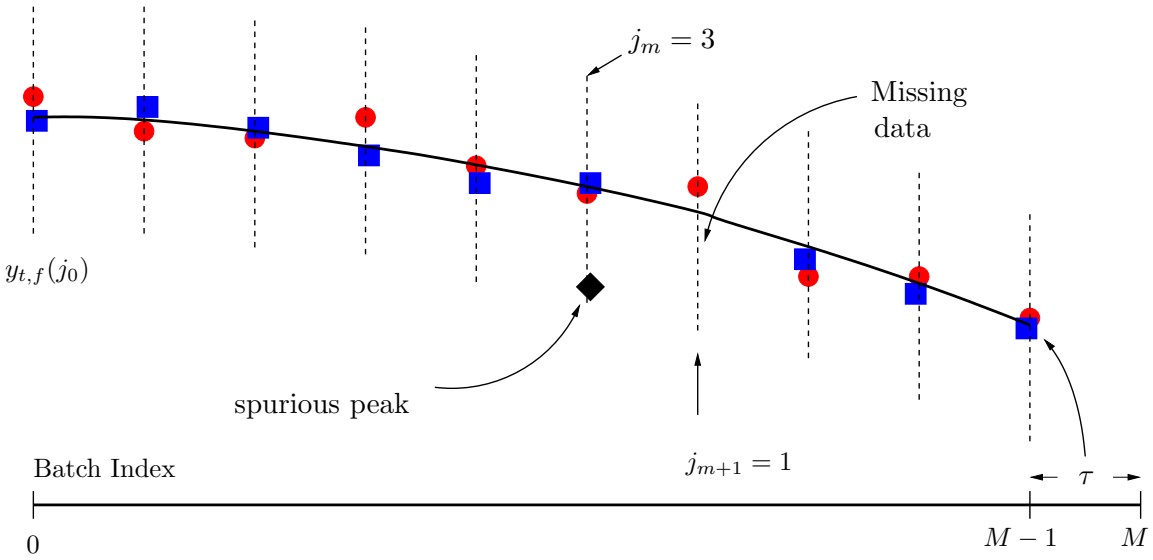


Figure 21: The circles and squares are the DOA estimates at different frequencies calculated using the acoustic data received during a period of length τ. Given the observations $\mathbf{y}_{t,f}$, the objective of the tracker is to determine the state $x_k(t)$ that completely parameterizes the solid curve.

This specific observation model is similar to that used in active contour image tracking problems [40]. The data likelihood given the state under the assumptions described above can be written as $p(\mathbf{y}_t|\mathbf{x}_t) \propto$

$$\prod_f \prod_k \prod_m \left\{ 1 + \frac{1}{\sqrt{2\pi}\kappa\lambda} \sum_{j_m} \exp - \frac{(h^{\theta}_{m\tau}(\mathbf{x}_t) - y_{t+m\tau,f}(j_m))^2}{2\sigma^2} \right\}, \tag{74}$$

where $\mathbf{y}_t = \{\mathbf{y}_{t,f}\}_{f=1}^{F}$ denotes the cumulative DOA data calculated in time interval $[t, t+T)$

[1]This model assumes that only one DOA at each f belongs to the target or that the target is missed. By assuming that the probability of the true measurement being in the observed data is equal for each j_m, a constant data miss probability κ may be assumed [40].

and h^{θ} refers to the DOA components. Equation (74) also assumes that each frequency observation set is independent of each other. If there is a known internal correlation structure, a joint density can be formulated to replace (74).

3.3 Particle Filter Details

The efficiency of the particle filter algorithm depends on the proposal functions that determine the random support of the particles to be *properly* weighted for estimation. In this section, a proposal function, denoted as $g(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-1})$, is derived to approximate the target posterior density directly:

$$g(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-1}) \approx p(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-1}) \propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}), \tag{75}$$

where $p(\mathbf{x}_t|\mathbf{x}_{t-1}) \sim \mathcal{N}(f_T(\mathbf{x}_{t-1}), \Sigma_u)$ and $p(\mathbf{y}_t|\mathbf{x}_t)$ is given by (74). Moreover, the proportionality in (75) is independent of the current state $\mathbf{x}_t$. This approximation, in effect, moves the particle stream towards high probability regions of the posterior so that more particles survive the final resampling step, producing better future states as the system evolves [48].

The posterior density should be approximated such that the resulting proposal function is as close to the posterior as possible and, at the same time, is easy to sample from. Hence, different Gaussian approximations to the full posteriors are commonly used in the literature [25]. In our case, we will approximate the data-likelihood by a Gaussian so that the proposal function will also be Gaussian. Hence, a mean μ_y and a covariance Σ_y should be first determined from the observed data $\mathbf{y}_t$. Then, by using the Gaussian parameters of the state update, an analytical relation for the proposal function will be given.

The mode of $p(\mathbf{y}_t|\mathbf{x}_t)$, denoted as $\mathbf{x}_M$, is a good candidate for the parameter μ_y. Then, the Hessian H of (74) at the mode can also be used as the covariance estimate $\Sigma_y = H^{-1}$. To calculate these parameters, a Newton search algorithm can be used on the negative log-likelihood of the data [77]. However, the resulting analytical relations have numerical sensitivity issues. As an alternative, we propose using the following cost function to

determine the mode $\mathbf{x}_M$:

$$J = -\sum_f \sum_k \sum_m \sum_{j_m} \exp\left\{-\frac{(h^\theta_{m\tau}(\mathbf{x}_t) - y_{t+m\tau,f}(j_m))^2}{2\sigma^2}\right\} + \frac{1}{2}(\mathbf{x}_t - \mathbf{x}_0)^T \Sigma^{-1}(\mathbf{x}_t - \mathbf{x}_0) \tag{76}$$

This cost function consists of two terms: the first term has the same minima as the negative log-likelihood function of the data distribution, and the second term is a regularization term that forces the solution $\mathbf{x}_M$ to lie close to some vector $\mathbf{x}_0$ with respect to a weighted distance measure defined by Σ.

Nominally, the mode should be within the particle cloud coming from the previous iteration after being propagated through the state update. Hence, an easy way to determine the mode would be to choose the particle best explaining the current data set (denoted as $\mathbf{x}_0$ in (76)). Unfortunately, when the targets maneuver, $\mathbf{x}_0$ may fall outside actual data observations. This necessitates the correction accomplished by the Newton algorithm to determine the actual mode $\mathbf{x}_M$.

Note that the cost function, without the regularization term, only depends on the angle distances. The gradients in that case may lead to physically infeasible (motion) changes in the states $Q(t)$ and $\phi(t)$ to account for fractional angle errors while determining $\mathbf{x}_M$. Hence, the regularization is introduced to constrain the solution space to lie in the Σ neighborhood of $\mathbf{x}_0$, and, at the same time, impose smoothness on the target motion. The parameter Σ also bounds the covariance of the data-likelihood approximation.

If we define $G = \partial J/\partial \mathbf{x}_t$ and $H = \partial^2 J/\partial \mathbf{x}_t \partial \mathbf{x}_t^T$, the Newton recursion is given by the familiar expression $\mathbf{x}_t^{l+1} = \mathbf{x}_t^l - \mu_l H^{-1} G$. The step size μ_l should be decreased adaptively, making sure that the cost function is always decreasing. Although time-consuming, it is straightforward to derive analytical expressions for G and H (similar calculations can be found in [77] as well as in Chapters 2 and 4.)

Even with the available analytical relations, the calculation of the Hessian still poses problems. If the Hessian of (76) is directly calculated from the exact formulas, it is possible to show that the resulting expression for H is not guaranteed to be positive definite, and

modifications are necessary to make the Newton correction $H^{-1}G$ effective at each iteration[2]. Hence, while calculating the final expression of the Hessian, terms including second order derivatives are neglected. In this case, the Hessian is a function of the outer product of the gradient, and it is possible to prove that it is positive definite.

After the Gaussian approximation to the data-likelihood described above, the final expression for the proposal function to be used in the particle filter is given by :

$$g(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-1}) \sim \mathcal{N}(\mu_g, \Sigma_g), \quad \text{where} \tag{77}$$

$$\Sigma_g = \left(\Sigma_y^{-1} + \Sigma_u^{-1}\right)^{-1}, \quad \mu = \Sigma_g\left(\Sigma_y^{-1}\mathbf{x}_M + \Sigma_u^{-1}h_T(\mathbf{x}_{t-1})\right). \tag{78}$$

Then, the particle filter incremental weights are given by

$$u^{(i)} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{g(\mathbf{x}_t^{(i)}|\mathbf{y}_t, \mathbf{x}_{t-1}^{(i)})} \tag{79}$$

The filter implementation used in the simulations also has a resampling (with replacement) stage after estimation.

3.4 Simulations

The objective of the simulations is to demonstrate the performance of the algorithm, using data generated according to the assumed models. For comparison, in the last example, we also generated acoustic data based on the narrow-band array model and calculated the observations using the minimum variance distortionless response (MVDR) beamformer. To initialize the trackers, we used a particle cloud with the correct mean, and a covariance of $\Sigma_0 = \text{diag}\{(2^\circ)^2\ ,\ (0.1)^2\ ,\ (4^\circ)^2\}$. For the Newton algorithm that approximates the data-likelihood for the particle proposal, an initial step size of 0.1 is used, which is decreased adaptively until 1000 iterations are reached. In the simulations, we used $\Sigma = \sqrt{2}\Sigma_u$. A complete list of the simulation parameters is given in Table 6.

Figure 22 demonstrates a single-target tracking scenario. The observed DOAs are Gaussian distributed around the true DOA track with variance $(2^\circ)^2$. The filter does a good job of catching the target as it maneuvers at $t = 16$s because it uses a large heading process

[2]The same issue applies to the the negative log-likelihood of (74).

Table 6: Simulation Parameters.

Fig.	N	T	M	F	σ	$\sigma_{u,\theta}$	$\sigma_{u,Q}$	$\sigma_{u,\phi}$
22, 24	100	1	10, 20	1	2°	1°	0.05	10°
23	100	1	10	2	3°	1°	0.05	10°

noise variance. Figure 23 shows a much more difficult scenario for the tracker, where DOA estimates calculated at two independent frequencies are given, each with the correct mean and a variance of $(3^\circ)^2$. There is a little bias in the DOA estimates between $t = 4$s and $t = 6$s where the targets are crossing. This bias is, in part, also due to the target maneuvers, which start at $t = 4$s. The filter maintains the track coherence in this difficult case using the independent observations. We observed that the filter can confuse the targets without the second set of DOAs coming from a different frequency. Although its estimates deteriorate in the region where targets are crossing as well as maneuvering, the filter locks back on the targets after the transient region by using the independent observations as well as the motion smoothness constraints.

For the last example (Fig. 24), acoustic data sampled at $F_s = 1000$Hz is generated for two targets, each with narrow-band frequencies $f_1 = 40$Hz and $f_2 = 80$Hz, using the narrow-band observation model [42]. The filter state is also augmented to include a frequency variable: $x_k(t) \triangleq [\ \theta_k(t)\ ,\ Q_k(t)\ ,\ \phi_k(t)\ ,\ f_k(t)\ ]^T$ [18]. Then, Gaussian noise is added to the array data such that the noise standard deviation is equal to one-tenth of the sinusoid amplitudes. The microphone array used for the simulation has 15 microphones situated uniformly on a circle such that the minimum inter-microphone distance is .45 times the wavelength of the second signal. The acoustic data is run through the MVDR beamformer, where the highest three peaks are picked with no particular order, at each batch index. The 1000 data samples in the batch are segmented into $M = 20$ sections, each containing 50 samples used to calculate each DOA. DOAs are used to output per state vector at a period of $T = 1$s.

Note that in this case, ignoring the model dependent angle bias ($\approx 0.2^\circ$), the calculated DOAs are distributed around the true DOA track with much less variance ($\approx (0.4^\circ)^2$) than the assumed variance $\sigma^2 = (2^\circ)^2$. Hence, the actual data-likelihood is narrower than what

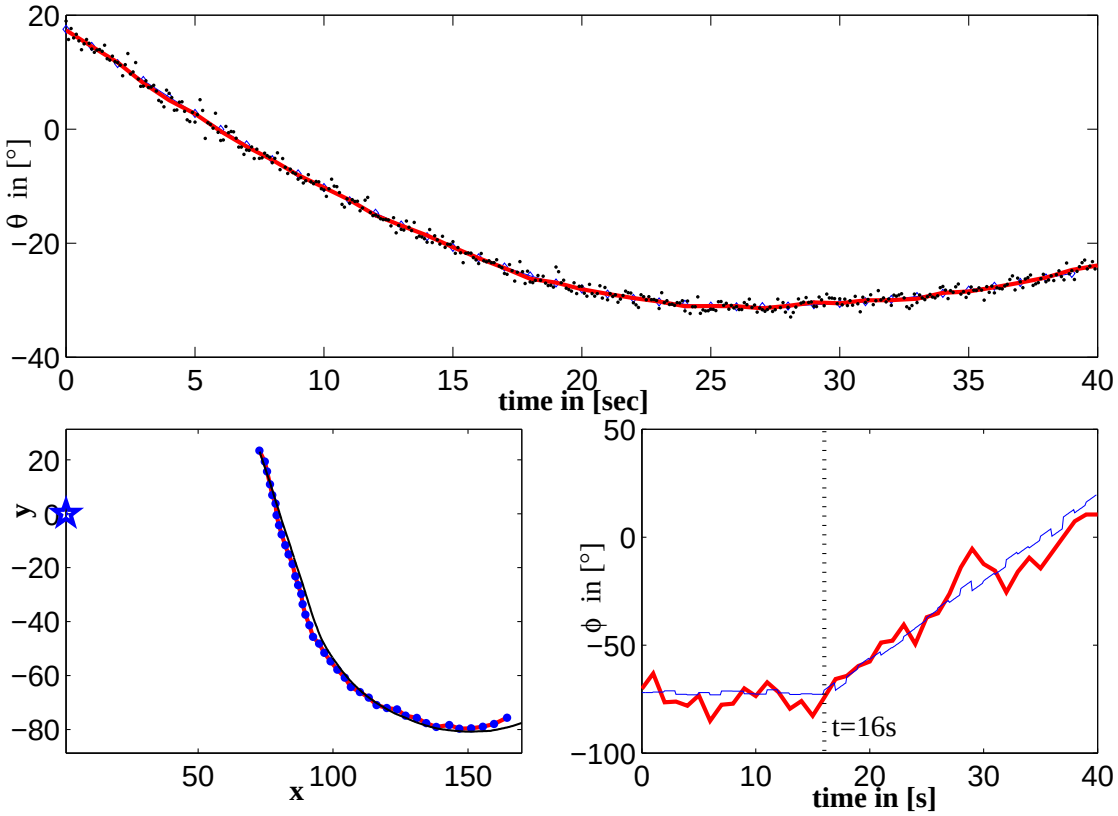


Figure 22: *Top*: Black dots are the DOA observations generated by adding Gaussian noise to the true target DOA track. The filter estimate is indistinguishable from the true DOA track. *Left*: Ground truth vs. filter estimate. The sensor array is shown with the star. The filter track estimate is calculated by using the filter outputs along with the correct initial target position. *Right*: Filter heading estimates vs. the true target heading.

is assumed by the tracker. This decreases the number of effective particles that contribute to the estimation accuracy. This is expected, since the filter is not matched to the data; however, this also demonstrates the robustness of the algorithm for the unmatched prior case.

3.5 Conclusions

In this chapter, a robust acoustic tracker is formulated in a flexible framework that makes minimal assumptions on the observations. Even though the filter requires a batch of data to process, it can be implemented for online applications because the data-likelihood approximation can be done as the data is being accumulated. Since the filter uses the Bayesian framework, it can avoid the data association problems commonly encountered in target

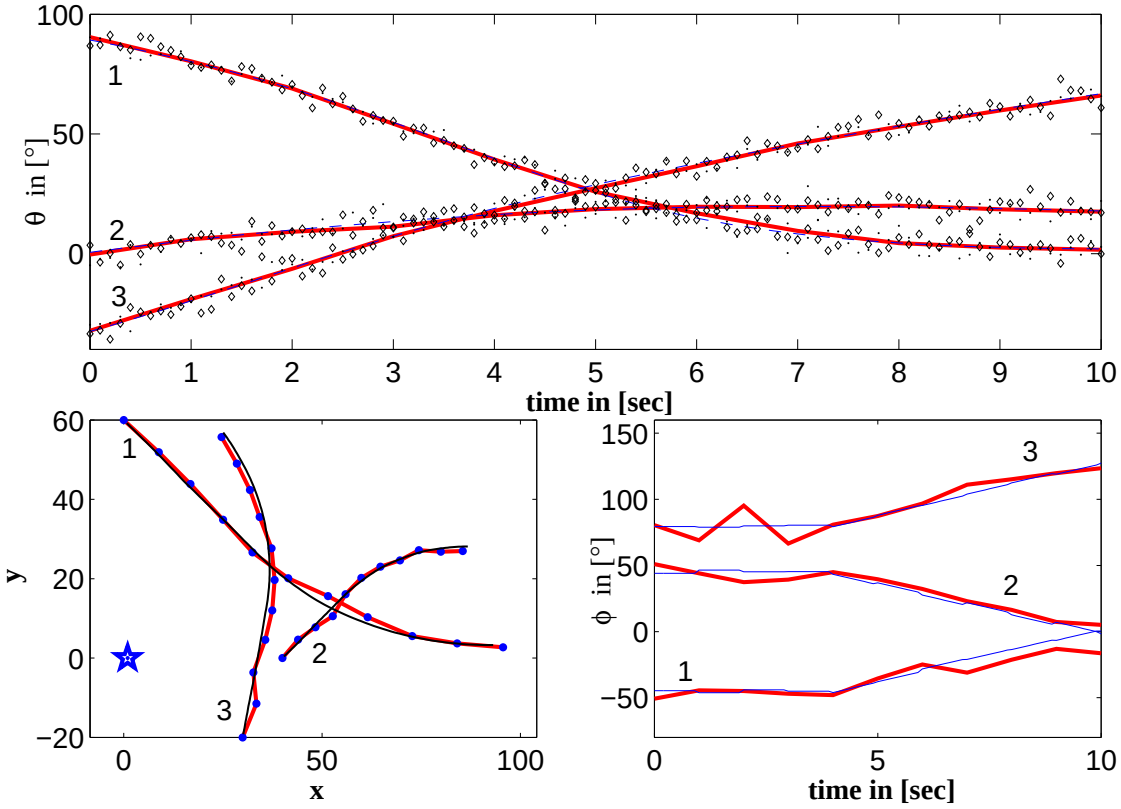


Figure 23: DOAs represented by diamonds and dots are generated by independent noise and are input to the filter unsorted.

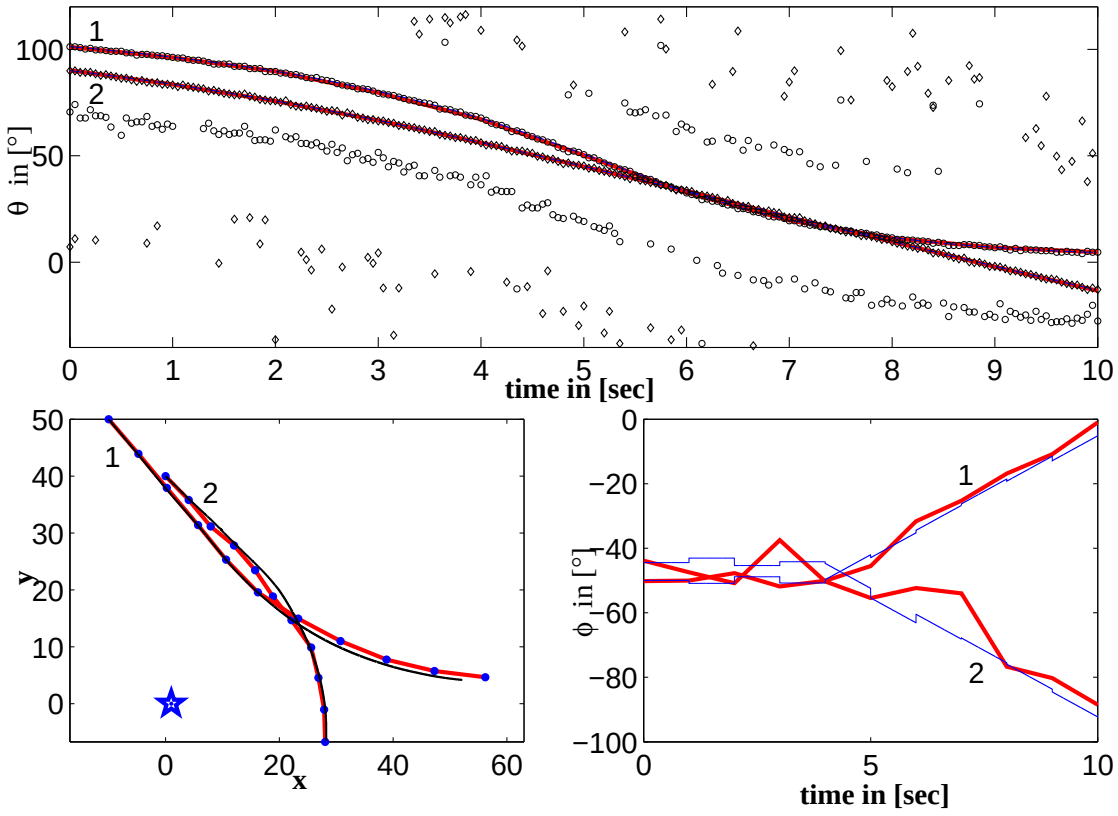


Figure 24: The DOA bias in the MVDR beamformer estimates is negligible, since the number of samples used for the estimation is small. The filter automatically separates DOAs with respect to the frequency planes defined by f_k in the state. This way the filter survives the period between times $t = 5$s and $t = 8$s where the motion vectors for both targets are very similar. Without the help of the frequency association, the filter confuses the targets. Note that due to the spacing of the sensors, there is spatial aliasing for target 2. That is why some of the spurious peaks (circles) form a track similar to the true target track.

tracking. The multiple target tracking performance can be further improved by altering the data likelihood so that a confusion probability is assigned to each data point while the targets are crossing.

CHAPTER IV

CALIBRATION: THE DUAL OF TARGET TRACKING

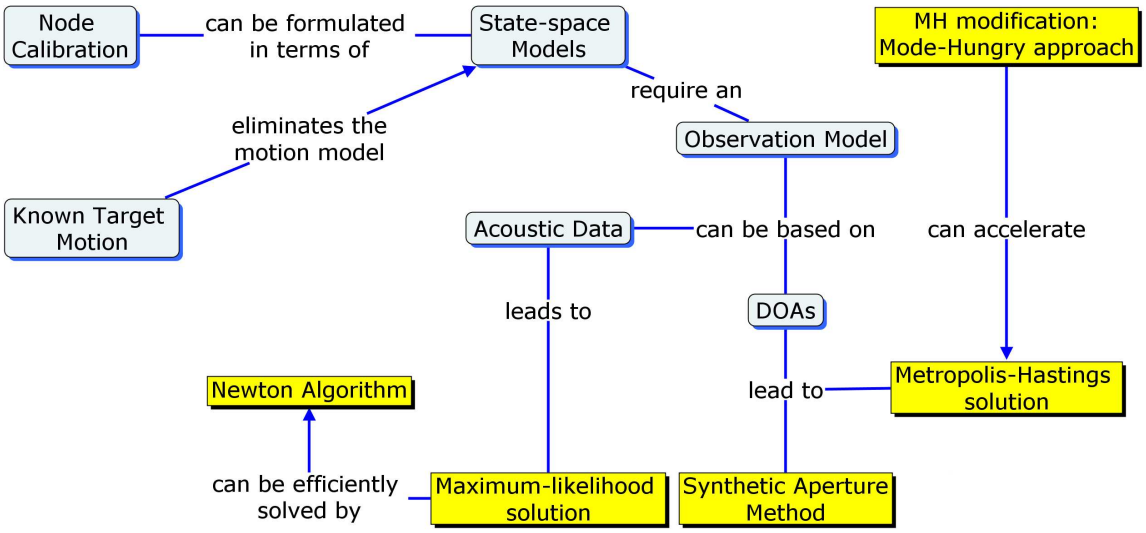


Figure 25: Chapter outline with the new contributions shown in rectangular boxes.

4.1 Introduction

If an acoustic node is defined to be an array of omnidirectional microphones whose relative positions are known with respect to each other, then the node calibration problem consists of determining the unknown array center and array orientation. This problem differs from the problem of calibrating the individual microphone positions previously considered in the literature [56,57]. In [57], accurate localization of individual microphone positions was done by considering the effects of the calibration on the array manifold matrix, which affects the DOA estimate. On the other hand, the methods described in this chapter calibrate one or more nodes in which the individual microphones have fixed relative positions. Multiple nodes would then be used to estimate target position via triangularization as shown in Fig. 26.

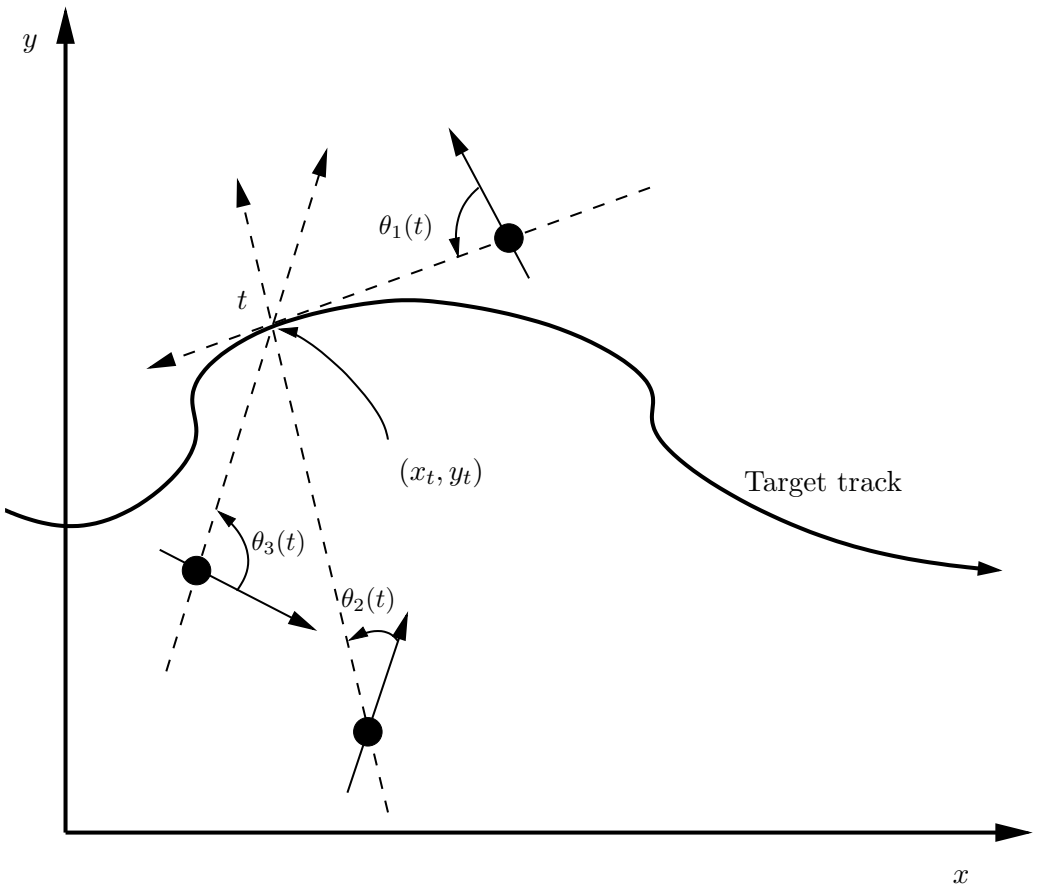


Figure 26: Black dots are the centers of acoustic nodes (arrays) and the solid arrows through the nodes represent their reference orientations for the local DOA estimates. The DOAs are measured counterclockwise from the reference. If the node centers and orientations are known, then it is possible to triangulate the target position (x_t, y_t).

There has been some previous work in calibrating acoustic nodes using beacons or moving sources [20, 52, 53]. The scenario considered in this chapter is similar to one considered in [20], where a moving calibration source is available. Here, it is assumed that the calibration source can report its position, e.g., acquired via the global positioning system (GPS). The acoustic nodes themselves are assumed to not have GPS capability due to battery limitations or jamming susceptibility (justifications can be found in [53].) The imperfect GPS position estimates of the moving source will be modelled as noisy, and the effects of the GPS noise on the estimation performance of the node positions will be treated in Section 4.5.1.

Several calibration methods are proposed in this thesis, including a Maximum Likelihood (ML) solution that works directly with the acoustic microphone array output signals, and two DOA-based algorithms that first process the microphone outputs to estimate the angle to the moving source, and then use simple geometry to calibrate from that angle information. The DOA-based algorithms include a Synthetic Aperture Calibration Method and a Metropolis-Hastings Calibration Method. The maximum-likelihood estimator of the node-center location is derived using an acoustic array data model. Then, a Newton search algorithm is employed to avoid calculating the whole ML surface. For the Newton algorithm, an initial node-center location and node orientation close to the true values must be provided. These initial estimates can be provided by any method, even one of the DOA calibration algorithms. We also give the Cramér-Rao performance bounds for the calibration problem for the different estimation methods and consider the sufficiency of these auxiliary DOA estimates.

The organization of the chapter is as follows. Section 4.2 formulates the problem and presents the ML solution along with performance bounds and the Newton search algorithm. Section 4.3 describes the DOA calibration algorithms that use auxiliary DOA estimates and geometrical arguments. Section 4.4 demonstrates a relative calibration algorithm based on a constant velocity calibration target. Computer simulations of a typical scenario are provided in Sect. 4.5.

4.2 A Maximum-Likelihood Solution for the Calibration Problem

In this section, we will present the ML solution for the node position and orientation, given a moving calibration source and local measurements of its sound at the node array microphones. It is assumed that the calibration source has a narrow-band time-frequency signature and its position estimates at each time are supplied by a GPS device. The GPS errors are modelled as *i.i.d.* Gaussian; their effects on calibration performance are considered in a later section. An example calibration source is a helicopter, which also has the capability to deploy the nodes in the field. The agility of such a calibration source introduces non-stationarity problems not often treated in the standard array model for beamforming. The ML estimate can handle this non-stationarity, but the DOA-based methods given in Section 4.3 require motion compensation to achieve the accuracy needed for calibration.

For the ML solution, it is necessary to have GPS estimates and acoustic measurements that are time-synchronized. However, since the calibration source is assumed to lie in the far-field of the sensor, its sounds will not arrive instantaneously at the acoustic node, and the acoustic data will be delayed relative to the earlier arrival of the GPS, which would be transmitted electronically. This acoustic propagation delay must be taken into account by estimating the actual time delays along with the node position and orientation. It is likely that more iterations of the ML algorithm will be needed to obtain correct automatic time synchronization from the data[1].

4.2.1 ML Solution

We define $\boldsymbol{\xi}$ as the vector consisting of the unknown node-center position (x, y) and the unknown node orientation φ in the 2D plane:

$$\boldsymbol{\xi} = [x, y, \varphi]^T \tag{80}$$

[1]Even if the sensor network has built-in global timing, e.g., by time-stamping the GPS data transmitted from the moving calibration source, there is no guarantee that the clocks in the moving source and the network are perfectly synchronized, so the time synchronization issue must be addressed during the processing.

The known (noisy) track of the moving source supplied by the GPS is $\boldsymbol{\chi}_t = [x_T(t), y_T(t)]^T$. Using $\boldsymbol{\xi}$ and $\boldsymbol{\chi}_t$, the node-to-source bearing angle θ_t (measured counterclockwise with respect to the x-axis) and range R_t at time t are

$$\begin{aligned} \theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t) &\triangleq -\varphi + \tan^{-1}\left(\frac{y - y_T(t)}{x - x_T(t)}\right), \\ R_t &\triangleq \|\boldsymbol{\xi}_{x,y} - \boldsymbol{\chi}_t\|. \end{aligned} \tag{81}$$

If the node estimate $\boldsymbol{\xi}$ is known, there is a one-to-one correspondence between $\boldsymbol{\chi}_t$ and (θ_t, R_t).

The acoustic signals at the microphones will be modeled according to the standard model:

$$\mathbf{y}(t) = \mathbf{a}(\theta_t)s(t) + \mathbf{n}_\mathbf{a}(t), \tag{82}$$

where $\mathbf{n}_\mathbf{a}(t)$ is additive noise. Section 2.2.3 discussed the assumptions that underlie the derivation of this model. Under the *i.i.d.* Gaussian assumption on the array noise $\mathbf{n}_\mathbf{a}(t)$, the probability density function (pdf) of the observed data is given by

$$p(\mathbf{Y}_K|\boldsymbol{\xi}, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t) = \prod_{t=0}^{K-1} \frac{1}{\pi^P \sigma^{2P}} \exp\left[-\frac{1}{\sigma^2}\|\mathbf{y}(t) - \mathbf{a}_t s(t)\|^2\right], \tag{83}$$

where σ^2 is the array noise variance, K is the total number of observations at the sampling frequency F_s, P is the number of microphones in the node, $\mathbf{a}_t \triangleq \mathbf{a}(\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t))$ is the steering vector, and $\mathbf{Y}_K$ is the aggregate data vector formed by stacking all the observed data as follows:

$$\mathbf{Y}_K = \begin{bmatrix} \mathbf{y}(t) \\ \mathbf{y}(t+\tau) \\ \vdots \\ \mathbf{y}(t+(K-1)\tau) \end{bmatrix}, \qquad \text{where } \tau = \frac{1}{F_s}. \tag{84}$$

Equation (83) models the data likelihood that we are seeking to maximize with respect to $\boldsymbol{\xi}$ to find the ML solution. First, the negative log-likelihood function is obtained

$$L^- \doteq KP\log(\pi\sigma^2) + \frac{1}{\sigma^2}\sum_{t=0}^{K-1}\|\mathbf{y}(t) - \mathbf{a}_t s(t)\|^2 \tag{85}$$

where $\doteq$ denotes equality up to a constant. Since the array steering vector $\mathbf{a}_t$ depends on $\boldsymbol{\xi}$, the optimal solution for $\boldsymbol{\xi}$ (node-center position and orientation) is the one that best aligns

the microphone output signals $\mathbf{y}(t)$ with shifted versions of the common source signal $s(t)$, given the source GPS track and the relative microphone positions.

The ML estimate (maximizing the log-likelihood function) is equivalent to minimizing L^-. We can eliminate $s(t)$ and σ^2 from the minimization. First of all, we fix $\boldsymbol{\xi}$ and $s(t)$ and minimize L^- with respect to σ^2 to find the ML noise variance. Next, the signal estimate is found by taking the variation of (85) with respect to $s(t)$, and setting it equal to zero. The results are:

$$\begin{aligned} \sigma^2_{ML} &= \frac{1}{KP} \sum_{t=0}^{K-1} \|\mathbf{y}(t) - \mathbf{a}_t(\boldsymbol{\xi}) s(t)\|^2 \\ s_{ML}(t) &= \tfrac{1}{P}\, \mathbf{a}_t^H(\boldsymbol{\xi}) \mathbf{y}(t) \end{aligned} \tag{86}$$

Substituting these ML estimates into L^-, we can rewrite the second term in (85) as an ML cost function that depends on $\boldsymbol{\xi}$ alone:

$$J(\boldsymbol{\xi}) = \sum_{t=0}^{K-1} \operatorname{tr}\left\{\mathbf{P}_t \hat{\mathbf{R}}_t\right\} = \sum_{t=0}^{K-1} \operatorname{tr}\left\{(\mathbf{I} - \tfrac{1}{P}\, \mathbf{a}_t(\boldsymbol{\xi}) \mathbf{a}_t^H(\boldsymbol{\xi})) \hat{\mathbf{R}}_t\right\} = \sum_{t=0}^{K-1} J_t(\boldsymbol{\xi}) \tag{87}$$

where $\mathbf{P}_t = \mathbf{I} - \frac{1}{P}\, \mathbf{a}_t(\boldsymbol{\xi}) \mathbf{a}_t^H(\boldsymbol{\xi})$ is the projection onto the null space of $\mathbf{a}_t^H(\boldsymbol{\xi})$, and $\hat{\mathbf{R}}_t = \mathbf{y}(t)\mathbf{y}^H(t)$ is the one-sample autocorrelation estimate. Finally, the ML estimate would be obtained by minimizing the cost function in (87)

$$\boldsymbol{\xi}_{ML} = \arg\min_{\boldsymbol{\xi}} J(\boldsymbol{\xi}). \tag{88}$$

4.2.2 Newton Search Algorithm

The solution to (88) might require the evaluation of the cost function $J(\boldsymbol{\xi})$ over the entire domain of the parameter vector $\boldsymbol{\xi}$ to obtain the global minimum, but this would be computationally expensive and unnecessary in most cases. The function $J(\boldsymbol{\xi})$ can have multiple local minima. However, if an initial estimate can be found that is reasonably close to the global minimum, then the cost function can be approximated via the expansion:

$$J(\boldsymbol{\xi} + \delta\boldsymbol{\xi}) \approx J(\boldsymbol{\xi}) + \mathbf{g}^T(\boldsymbol{\xi}) \delta\boldsymbol{\xi} + \tfrac{1}{2}\, \delta\boldsymbol{\xi}^T \mathbf{H}(\boldsymbol{\xi}) \delta\boldsymbol{\xi}, \tag{89}$$

where $\mathbf{g}$ and $\mathbf{H}$ are the gradient and the Hessian of $J(\boldsymbol{\xi})$, respectively. The necessary gradient of the cost function (87) can be calculated using the chain rule:

$$\mathbf{g} \triangleq \frac{\partial J(t)}{\partial \boldsymbol{\xi}} = \frac{1}{M} \sum_{t=0}^{M-1} V(t) \mathrm{diag}(\nabla_\theta J_t), \tag{90}$$

where

$$V(t) \triangleq \frac{\partial \theta_t}{\partial \boldsymbol{\xi}} \tag{91}$$

and

$$\nabla_\theta J_t = -2\mathrm{Re}\left\{ \mathbf{a}_t^\dagger \hat{\mathbf{R}}_t \mathbf{P}_t \frac{\partial \mathbf{a}_t}{\partial \theta_t} \right\}. \tag{92}$$

The Hessian $\mathbf{H} \triangleq \dfrac{\partial^2 J(t)}{\partial \boldsymbol{\xi} \partial \boldsymbol{\xi}^H}$ is then given by

$$\mathbf{H} = \frac{1}{M} \sum_{t=0}^{M-1} \left\{ [\nabla^2_{\theta\theta} J_t \otimes 1] \odot [V(t)^H V(t)] + [\mathrm{diag}(\nabla_\theta J_t \otimes I)] \odot \Upsilon(t) \right\}, \tag{93}$$

where $\Upsilon(t)$ is the Hessian of θ_t with respect to $\boldsymbol{\xi}$, and $\mathbf{1}$ and I denote a matrix of all ones and the identity matrix of appropriate dimensions. Symbols $\otimes$ and $\odot$ denote the Kronecker and Schur products, respectively. To guarantee the positive definiteness of $\mathbf{H}$, the term containing $\Upsilon(t)$ in (93) is ignored while calculating the Hessian as discussed in [77]. The following derivative also needs to be approximated while calculating $\nabla^2_{\theta\theta} J_t$:

$$\frac{\partial^2 J_t}{\partial \theta(t) \partial \theta(t)} \simeq 2\mathrm{Re}\{\gamma^H(t)\gamma(t)\}, \tag{94}$$

where we define $\gamma(t) = \dfrac{\partial \mathbf{P}_t \mathbf{y}(t)}{\partial \theta_t}$.

Newton's method can be employed to find the global minimum via the iteration:

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k - \mu_k \mathbf{H}_k^{-1} \mathbf{g}_k, \tag{95}$$

where $\mathbf{g}_k$ and $\mathbf{H}_k$ are the k^{th} step estimates of the gradient and the Hessian, and μ_k is a non-negative step size. It is well-known that Newton's method exhibits quadratic convergence when starting sufficiently close to the optimum point.

At this point, it is important to recall that the ML solution above does not include the effects of errors in the GPS track ($\boldsymbol{\chi}_t$) on the estimation performance of finding $\boldsymbol{\xi}$. This is addressed in the next subsection, where it is shown that such errors are usually negligible.

4.2.3 Effects of the GPS Errors on the Estimation Performance

In the 2D problem under consideration, the GPS outputs $\boldsymbol{\chi}_t$ have components in the x and y directions, and if we model errors in the GPS as zero mean *i.i.d.* Gaussian noise, $\mathbf{n}_\chi \sim \mathcal{N}(\mathbf{0}, \sigma_\chi^2 \mathbf{I})$, then we can compare the relative effects of GPS noise and microphone noise on the final calibrated node-position estimate. The size of the GPS errors is generally on the order of 1–5 meters. Rather than finding the variance of the final calibrated node-position estimates directly, we develop a formula for the array output $\mathbf{y}(t)$ that includes a perturbation due to GPS noise. Then the size of this perturbation term is compared to the array noise to derive the condition under which GPS noise is negligible.

The array output $\mathbf{y}(t)$ depends on the angle to the source θ_t, which, in turn, depends on the source location. With GPS noise there is uncertainty in the source location, which translates into a perturbation of the angle and finally into additive noise in the array output. The first step is to find the effect of GPS noise on the auxiliary variable θ_t defined in (81). Expanding in a Taylor series, the first-order perturbation can be modeled as follows:

$$\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t + \mathbf{n}_\chi(t)) \approx \theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t) + n_\chi^x(t) \frac{\partial \theta_t}{\partial n_\chi^x(t)}\bigg|_{n_\chi^x(t)=0} + n_\chi^y(t) \frac{\partial \theta_t}{\partial n_\chi^y(t)}\bigg|_{n_\chi^y(t)=0}. \tag{96}$$

By taking the necessary derivatives of (81) and noting that the noise $\mathbf{n}_\chi(t)$ is independent in the x and y directions, and that the second-order terms are very small, we can approximate the DOA θ_t as Gaussian:

$$\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t + \mathbf{n}_\chi(t)) \sim \mathcal{N}\left(\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t), \frac{1}{R_t^2}\sigma_\chi^2\right), \tag{97}$$

with the correct mean $\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t)$ and a variance that is inversely proportional to the square of the range, R_t^2. This leads to an intuitive result: when the GPS errors are very small compared to the target range, the position errors will translate into an approximate angle error of $\tan^{-1}(\sigma_\chi / R_t) \approx \sigma_\chi / R_t$, which is tiny when R_t is large.

The next step is to derive the effect of GPS noise on the steering vector. Once again, a first-order approximation can be used:

$$\mathbf{a}(\boldsymbol{\xi}, \boldsymbol{\chi}_t + \mathbf{n}_\chi(t)) \approx \mathbf{a}(\boldsymbol{\xi}, \boldsymbol{\chi}_t) + \frac{\partial \mathbf{a}}{\partial \theta} n_\theta. \tag{98}$$

As an example, we explicitly derive the effect for a narrow-band calibration source, whose center frequency is f_0. To simplify the mathematical expressions, assume that the p^{th} microphone position is given in polar coordinates (ρ_p, ϕ_p), and the array steering vector (50) derived in the Chapter 2 can be written as

$$[\mathbf{a}(\boldsymbol{\xi}, \boldsymbol{\chi}_t)]_p = \exp\left[j\frac{2\pi f_0 \rho_p}{c}\sin(\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}_t) + \phi_p)\right], \tag{99}$$

where c is the speed of sound and $[\mathbf{a}]_p$ indicates the p^{th} element of the vector $\mathbf{a}$. The derivative of (99) becomes

$$\frac{\partial \mathbf{a}}{\partial \theta} = \frac{j2\pi f_0}{c}\mathbf{a}(\boldsymbol{\xi}, \boldsymbol{\chi}_t)\boldsymbol{\lambda}(\theta),$$

where $\boldsymbol{\lambda}(\theta) = \text{diag}\{[\rho_1 \cos(\theta + \phi_1), \ldots, \rho_P \cos(\theta + \phi_P)]\}$. If we define $\boldsymbol{\Lambda}(\theta) = \boldsymbol{\lambda}(\theta) \odot \boldsymbol{\lambda}(\theta)$, then the array outputs for a target signal with constant envelope magnitude of one can be shown to obey the following Gaussian distribution

$$\mathbf{y}(t) \sim \mathcal{N}\left(\mathbf{a}(\boldsymbol{\xi}, \boldsymbol{\chi}_t), \boldsymbol{\Sigma}(\boldsymbol{\xi}, \boldsymbol{\chi}_t)\right), \tag{100}$$

where the autocorrelation matrix $\boldsymbol{\Sigma}$ is a function of the array noise as well as the GPS noise:

$$\boldsymbol{\Sigma}(\boldsymbol{\xi}, \boldsymbol{\chi}_t) = \sigma^2\mathbf{I} + \left(\frac{2\pi f_0 \sigma_\chi}{cR_t}\right)^2 \mathbf{a}(\theta_t)\boldsymbol{\Lambda}(\theta_t)\mathbf{a}^H(\theta_t). \tag{101}$$

The second term in (101) is the perturbation due to GPS errors. If $\sigma \gg \frac{2\pi f_0 \sigma_\chi}{cR_t}$, then it can be argued that GPS errors have a small impact on the estimation performance because the data likelihood (83) is not affected. For most cases of interest in our work, this is reasonable because the narrow-band frequencies of the source are usually less than 100 Hz, the GPS error standard deviation σ_χ is on the order of a few meters, and the range R_t is a kilometer or more.

4.2.4 Cramér-Rao Lower Bound for the Estimate of $\boldsymbol{\xi}$

The Cramér-Rao lower bound (CRLB) is an information theoretic inequality, which provides a lower bound for the variances of the unbiased estimators. If an estimator achieves the CRLB, then it is also a solution of the likelihood equation. However, it is not always true that the ML solution will achieve the CRLB (at least, for finite sample sizes) or that it will

be unbiased [63]. The CRLB is still a useful metric with which to compare the performance of an algorithm, and is derived for the calibration problem in this section.

First, we derive an expression for the Fisher information matrix (FIM). Assume that the noise variance σ^2 is known. The log-likelihood function (83) for the parameter vector $\boldsymbol{\xi}$ simplifies to the following relation:

$$L(\boldsymbol{\xi}) \doteq -\frac{1}{\sigma^2} \sum_{t=0}^{K-1} \|\mathbf{y}(t) - \mathbf{a}_t s(t)\|^2 \tag{102}$$

where $\mathbf{a}_t = \mathbf{a}_t(\theta_t(\boldsymbol{\xi}, \boldsymbol{\chi}))$ is the steering vector from the node position to the calibration source. The $(i,j)^{\text{th}}$ element of the FIM is given by partial derivatives of (102) with respect to the i^{th} and j^{th} parameters of the vector $\boldsymbol{\xi}$

$$F_{i,j} = E_{\mathbf{y}} \left\{ \frac{\partial^2 L_\chi(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}_i \partial \boldsymbol{\xi}_j} \right\} = -\frac{2}{\sigma^2} \sum_t \Re \left\{ \left(\frac{\partial \mathbf{a}_t}{\partial \boldsymbol{\xi}_i} \right)^H \frac{\partial \mathbf{a}_t}{\partial \boldsymbol{\xi}_j} \right\} \tag{103}$$

where $\boldsymbol{\xi}_i = [\boldsymbol{\xi}]_i$ and $E_{\mathbf{y}}\{\cdot\}$ denotes the expected value with respect to the data distribution (see [57] for a similar derivation). The Cramér-Rao lower bound is the inverse of this expression [63]. Interestingly, the CR bound depends on the source's narrow-band frequency: the higher the frequency, the lower the localization bound. In Sect. 4.5 on simulations, Fig. 32 shows this dependence for a specific scenario.

4.3 DOA-Based Calibration Algorithms

In the previous section, the calibration problem was introduced and the maximum-likelihood solution presented. The remainder of the chapter treats another set of methods called *DOA-Based calibration algorithms* that exploit the geometry of the problem defined by the GPS (carried by the source) along with estimated DOAs (at the node). These methods rely on the fact that the DOA is a sufficient statistic from which it is possible to determine the node-center position and orientation. We first explain the angle matching idea used by the DOA-based calibration methods. Then, two DOA-based calibration methods are studied: the first uses the synthetic aperture concept from radar, and the second, a Metropolis-Hastings type of Monté-Carlo algorithm. For both DOA calibration algorithms, performance bounds are derived and examples are included.

4.3.1 Angle Matching

The DOA-based calibration algorithms use a simple angle matching idea coming from the geometry of the problem, illustrated by Fig. 27. As the target moves, the node estimates a DOA track $\theta(t)$ with respect to its orientation by using the target's acoustic data, independent of the node position. The node can also calculate a node-to-GPS angle track $\psi(t)$ as a function of the node position (x, y) and the GPS track $(x_T(t), y_T(t))$ transmitted by the target (Fig. 27):

$$\psi(x, y, t) = \tan^{-1}\left(\frac{y - y_T(t)}{x - x_T(t)}\right) = \theta(t) + \varphi. \tag{104}$$

Then, by assuming that the DOA estimation errors are zero mean Gaussian random variables, a maximum-likelihood solution can be found for $\boldsymbol{\xi}$ by minimizing the following cost function:

$$\boldsymbol{\xi}_{ML} = \arg\min_{\boldsymbol{\xi}} \sum_t \frac{(\psi(x, y, t) - \theta(t) - \varphi)^2}{\sigma^2_{\theta_t}}, \tag{105}$$

where $\sigma^2_{\theta_t}$ is the DOA estimation variance, which will be discussed later.

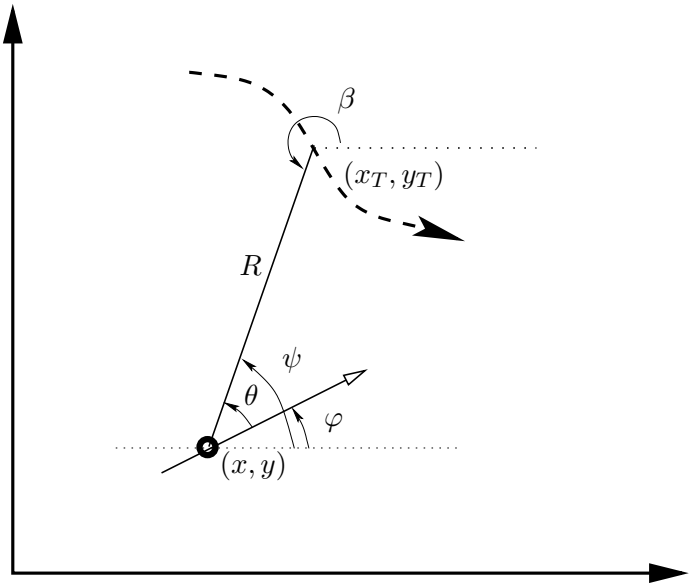


Figure 27: The node position is shown with a circle at the position (x, y). As the target moves (dashed line), the node can calculate two angles $\theta(t)$ and $\psi(t)$ that should be matched to determine the unknown parameter vector $\boldsymbol{\xi}$. The parameter β is called the synthetic aperture radar (SAR) angle, as explained in the next section.

In reality, the above solution will calibrate the array only moderately well because it does not consider the target acoustic data propagation time. When the target reports

its GPS position electronically at some position $\boldsymbol{\chi}_t$, the GPS information arrives at the node with the speed of light. However, the acoustic data corresponding to GPS-reported position arrives at the node with the speed of sound, which is significantly slower. Hence, time synchronization is required to match the estimated DOAs to the node-to-GPS angles.

Denote $\tau(t)$ as the arrival time at the node for a sound emitted at the "source" at time t. Then,

$$\tau(t) = t + ||\boldsymbol{\xi}_{x,y} - \boldsymbol{\chi}_t||/c, \tag{106}$$

where c is the speed of sound. Hence, the angle matching equation (104) as well as the ML solution to it should be modified accordingly

$$\psi(x, y, t) = \theta(\tau(t)) + \varphi \tag{107}$$

$$\boldsymbol{\xi}_{ML} = \arg\min_{\boldsymbol{\xi}} \sum_t \frac{(\psi(x, y, t) - \theta(\tau(t)) - \varphi)^2}{\sigma_{\theta_t}^2}. \tag{108}$$

Another Newton type of algorithm can be formulated for the solution of the above equation. The solution will not be discussed here because it is trivial and a similar ML solution was considered in the previous section. Instead, we will explain two novel solutions, one based on the synthetic aperture radar concept, and the other based on Monté Carlo simulation methods.

4.3.2 Synthetic Aperture Calibration Method

The synthetic aperture concept is the idea of creating a large aperture size from a small moving physical aperture to obtain better angular resolution. In the node calibration problem, if the problem is reformulated by time-reversing the events such that the fixed node is considered to be the sound source and the moving calibration source the receiver(s), then it is possible to apply the synthetic aperture concept. That is, we will create a moving pseudo-receiver defined by the calibration source's GPS track and assume that the signals are coming from the acoustic node (Fig. 28). The moving pseudo-receiver can be grouped into synthetic subarrays, from which we can estimate the DOAs to the fixed node, and then calibrate the node position by the triangulation geometry.

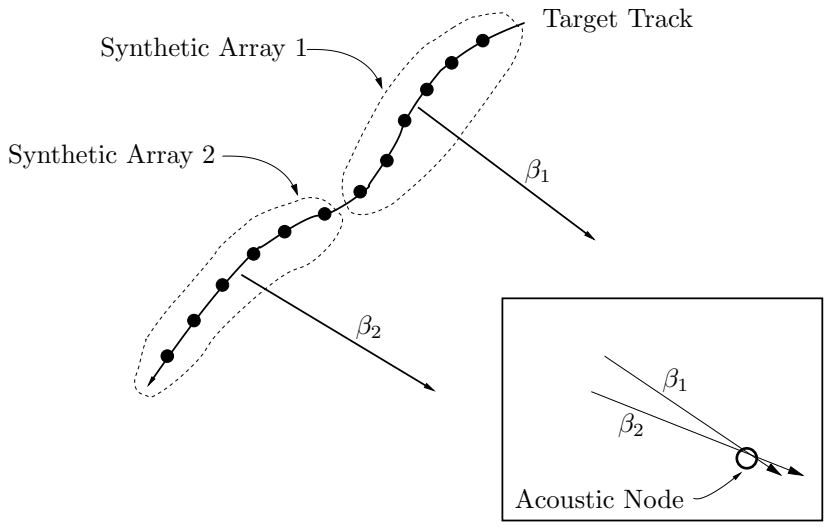


Figure 28: The moving calibration source can be interpreted as a moving pseudo-receiver that creates a synthetic aperture. Pseudo-receiver positions can then be grouped into subarrays and used to estimate DOAs assuming the signals are coming from the fixed sensor node. This does not require additional transmission of the recorded target sounds from the node, since this estimation can be done at the node.

A single *synthetic receiver* position will consist of a fixed number of signal samples M and will be assigned a fixed position, even though the *real receiver* is actually moving while the M samples are acquired. The number of samples M will determine the intersensor spacing for the synthetic array, so spatial aliasing of the acoustic signals must be taken into account. The distance traveled by the calibration source moving at a velocity v during M samples (sampled at F_s) is Mv/F_s. In conventional array processing, it is well-known that if the sensor spacing is less than half the wavelength of the signal of interest, then spatial aliasing can be avoided [42]. Hence, an upper bound on M is:

$$M < \frac{cF_s}{2f_0 v} \tag{109}$$

where $\lambda/2 = c/(2f_0)$, f_0 is the narrow-band center frequency, and c is the speed of sound. As a rule of thumb, we recommend using the largest M that satisfies (109).

The total number of synthetic receiver positions used in forming a subarray is subject to the following trade-off: longer subarrays would give better DOAs, but would provide fewer DOAs to do triangulation. Surprisingly, it appears that the node position estimation accuracy was not affected much by this trade-off when tested on synthetic data.

We now describe the details of the synthetic aperture calibration method. The acoustic data recorded at each microphone of the acoustic node is temporally partitioned into M-sample data sets, and the midpoint time is used to define the position for each synthetic receiver along the path of the moving calibration source, $\boldsymbol{\chi}_t$. Two issues arise from this definition of the synthetic receiver. First, each sample in the block of M data samples comes from the neighborhood of the defined receiver position through which the source was moving. We can model this as a non-stationary effect in the received data. Second, since the GPS positions are usually supplied at a much lower rate than the acoustic data sampling rate, the synthetic receiver positions must be estimated by using a straight-line approximation between the given GPS data points. For example, if $F_s = 1024$ Hz, and $M = 64$, then the synthetic receiver positions must be determined 16 times per second, requiring interpolation since the GPS sampling rate was slower.

A subarray is formed by grouping together Q synthetic receiver positions. The intrasensor spacing is Mv/F_s meters, assuming v is constant. Simple beamforming such as MUSIC or MVDR will give based DOA estimates if the calibration source is moving fast (e.g., a helicopter). This bias is caused by the non-stationarity of the synthetic array data mentioned above, but previous work in [77] has addressed this same issue and shown how to calculate the bias values theoretically. Therefore, we propose the following ML cost function, derived similarly to (87), for obtaining unbiased synthetic DOA estimates β:

$$J(\beta) = \sum_{t=1}^{M} \mathrm{tr}\left\{ \left[\mathbf{I} - \frac{1}{Q}\mathbf{a}_t(\beta)\mathbf{a}_t^H(\beta) \right] \hat{\mathbf{R}}_z(t) \right\}, \tag{110}$$

where $\mathbf{a}_t(\beta)$ is the steering vector corresponding to aperture points and β is defined as the synthetic aperture radar angle (Fig. 27). The time dependence in $\mathbf{a}_t(\beta)$ is caused by the data samples at each synthetic sensor being from a neighborhood of the synthetic sensor position. The angle β is calculated with respect to the center of the synthetic aperture of size MQ samples, and $\hat{\mathbf{R}}_z(t)$ is the one-sample (outer product) autocorrelation estimate of the synthetic array. Equation (110) differs from (87) in the time dependence of the steering vectors. In (87), the microphone positions are fixed with respect to t; whereas, in (110), the SAR angle β is fixed, and the position where the data is collected is changing along the

aperture.

Equation (110) can be used to determine the position of each individual microphone within the node. Since there are P microphones in the node, it is possible to come up with P independent estimates of the node position using the individual microphone outputs, assuming that the additive noise is spatially white at the acoustic node. Then, the P estimated microphone positions can be averaged to obtain an estimate of the node-center position. The node orientation can then be estimated separately. To estimate the node orientation, the acoustic node DOA estimates can be used along with the node-center position estimate to determine the orientation. Once again, it is important to recognize the array non-stationarity issue caused by rapid target movement. The ML cost function to be minimized at the acoustic node in this case is

$$J(\theta) = \sum_{t=1}^{M} \mathrm{tr}\left\{ \left[\mathbf{I} - \frac{1}{P}\mathbf{a}(\theta_t)\mathbf{a}^H(\theta_t) \right] \hat{\mathbf{R}}_y(t) \right\}, \tag{111}$$

where

$$\theta_t = \tan^{-1}\left\{ \frac{F_s \sin\theta + qt\cos\phi}{F_s \cos\theta + qt\sin\phi} \right\}, \tag{112}$$

where θ is the DOA at the beginning of the batch, $q = v/R$, and ϕ is the approximate target orientation for the estimation batch. Equation (112) can be derived by a straight-line approximation for the calibration source during a batch period. Given the node-center position estimate, if the target DOAs are estimated using (111), one can estimate the node orientation by simply calculating the bearings from the node position and GPS estimates, and then finding the difference between the mean values of these estimates.

When (110) is used for the SAR angles, the cost function will display two minima corresponding to two different candidate SAR angles. This is attributed to the cone of ambiguity problem for microphone arrays. Even when there is no spatial aliasing, the estimator (110) will result in two DOA estimates that are symmetric with respect to the array axis (calibration source orientation angle)[2]. This issue has been addressed for uniform linear arrays in [42]. The solution to this problem is to also track the DOA estimates and impose a

[2]The average of these two DOA candidates can be used to check the consistency of the target orientation angle.

constraint that the DOA estimates not change too much from one aperture to another. This fact also helps reduce the computational load while finding the minimum of the cost function in (111), since previous estimates will be close to the sought minima. In turn, two node positions will be triangulated using the two DOA tracks, one corresponding to the node and the other corresponding to a shadow. The real node can easily be determined by picking the node position with the least minimum-mean squared error, since it actually corresponds to a physical position.

Once the SAR angles (β_i's $i = 1, \ldots, L$) are calculated for each synthetic array with respect to the array centers $(x_T^{(i)}, y_T^{(i)})$, determination of the node position requires one more step. Figure 28 suggests an intuitive solution based on determining the intersection of the lines created by the aperture positions and the SAR angles. This leads to the following overdetermined system of equations to determine the node position:

$$\begin{bmatrix} \tan(\pi - \beta_1) & -1 \\ \vdots & \vdots \\ \tan(\pi - \beta_L) & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \tan(\pi - \beta_1) x_T^{(1)} - y_T^{(1)} \\ \vdots \\ \tan(\pi - \beta_L) x_T^{(L)} - y_T^{(L)} \end{bmatrix} \tag{113}$$

This equation is based on the following geometrical relationship between the node DOA θ_i, its orientation φ, and the SAR angle β_i, as illustrated by Fig. 27:

$$\beta_i = \pm\pi + \theta_i + \varphi = \pm\pi + \tan^{-1}\left(\frac{y - y_T^{(i)}}{x - x_T^{(i)}}\right) \tag{114}$$

Equation (114) defines an under-determined system for $\boldsymbol{\xi}$ for one index i, hence making it analytically intractable to derive a joint probability density function $p(x, y)$, since the required Jacobians for the coordinate transformation from β to (x, y) are not defined[3]. Equation (113) provides the least-squares solution for the node position. However, the estimated DOAs become approximately Gaussian distributed as the sampling frequency of the data increases. Based on this observation, similar to the ML solution using the

[3]This is because the coordinate transformation defined by (114) is not one-to-one.

node-to-GPS angles ψ shown in the previous section, we propose the following

$$\begin{aligned}\boldsymbol{\xi}_{x,y} &= \arg\min_{\boldsymbol{\xi}_{x,y}} J \\ J &= \sum_{i=1}^{L} \frac{\left(\beta_i - \tan^{-1}\left(\frac{y-y_T^{(i)}}{x-x_T^{(i)}}\right) \pm \pi\right)^2}{\sigma_{\beta_i}^2},\end{aligned} \tag{115}$$

where the sign in front of π must be determined from the geometry and the SAR angle error variance $\sigma_{\beta_i}^2$ should be calculated from the data. A possible expression will be given in the later sections. This estimator can be refined by weighting the terms in the summation by the estimated noise variances per index. The following sections give analytical equations for these variance estimates. Equations (113) and (115) are biased estimators (shown by simulations); however, the bias is difficult to analyze analytically.

A Newton-based search can again be employed to solve (115). The first and second-order derivatives of the cost function J are given in (116) and (117) for a Newton-based search algorithm:

$$\begin{aligned}\frac{\partial J}{\partial x} &= 2\sum_{i=1}^{L}\left\{\beta_i - \tan^{-1}\left(\frac{y-y_T^{(i)}}{x-x_T^{(i)}}\right) + \pi\right\}\left(\frac{y-y_T^{(i)}}{R_i^2}\right) \\ \frac{\partial J}{\partial y} &= -2\sum_{i=1}^{L}\left\{\beta_i - \tan^{-1}\left(\frac{y-y_T^{(i)}}{x-x_T^{(i)}}\right) + \pi\right\}\left(\frac{x-x_T^{(i)}}{R_i^2}\right)\end{aligned} \tag{116}$$

$$\frac{\partial^2 J}{\partial x^2} = 2\sum_{i=1}^{L}\left\{\left(\frac{y-y_T^{(i)}}{R_i^2}\right)^2 + \left(\beta_i - \tan^{-1}\left(\frac{y-y_T^{(i)}}{x-x_T^{(i)}}\right) + \pi\right)\left(\frac{-2(x-x_T^{(i)})(y-y_T^{(i)})}{R_i^4}\right)\right\}$$

$$\frac{\partial^2 J}{\partial y^2} = 2\sum_{i=1}^{L}\left\{\left(\frac{x-x_T^{(i)}}{R_i^2}\right)^2 + \left(\beta_i - \tan^{-1}\left(\frac{y-y_T^{(i)}}{x-x_T^{(i)}}\right) + \pi\right)\left(\frac{2(x-x_T^{(i)})(y-y_T^{(i)})}{R_i^4}\right)\right\}$$

$$\frac{\partial^2 J}{\partial x \partial y} =$$

$$-2\sum_{i=1}^{L}\left\{\frac{(x-x_T^{(i)})(y-y_T^{(i)})}{R_i^4} + \left(\beta_i - \tan^{-1}\left(\frac{y-y_T^{(i)}}{x-x_T^{(i)}}\right) + \pi\right)\left(\frac{(y-y_T^{(i)})^2 - (x-x_T^{(i)})^2}{R_i^4}\right)\right\} \tag{117}$$

These derivatives are useful for estimating the angles since the SAR angles β_i are very close to each other. Hence, the previous estimate β_{i-1} can be refined using these derivatives to estimate β_i. This is conditioned on the fact that the target is not maneuvering too much, because the Newton search can become trapped at a local minima. We recommend

calculating J in (115) first on a rough grid nearby β_{i-1} (e.g., a grid of angles $\beta \in [\beta_{i-1} - 15°, \beta_{i-1}+15°]$ with $1°$ spacing), then using the minima over that rough grid and the Newton algorithm to refine the estimate for β_i.

4.3.3 Metropolis-Hastings Calibration Method

The objective of the Metropolis-Hastings (MH) algorithm is to distribute particles (discrete state samples $\boldsymbol{\xi}_i$) according to a target distribution π. Hence, at each iteration k, the algorithm recursively redistributes its states so that, asymptotically, the resulting Markov chain is distributed according to the target distribution. The MH scheme [24] is depicted in Sect. 5.2 of the next chapter.

For the calibration problem, the target function is actually the exponential of the negative ML cost function defined earlier in (107):

$$\pi(\boldsymbol{\xi}) \propto \exp\left\{-\frac{(\psi(x,y,t) - \theta(\tau(t)) - \varphi)^2}{2\sigma_{\theta_t}^2}\right\}, \tag{118}$$

where ψ is the node-to-GPS angle, θ_t is the DOA estimate from the acoustic data using (111), and φ is the node orientation angle. Time synchronization is easily incorporated into this solution because it is possible to calculate $\tau(t)$ given a proposed node position and the target GPS.

Moreover, it is easy to see that (118) peaks at the correct node position when there is no noise. Intuitively, this is the complement of the synthetic aperture solution, where the angles are calculated on the node side, and performs similarly to the estimator proposed in (115). As for the proposal function, we employ the symmetric random walk in (131), as explained in Chapter 5, where the walk variances should be picked subjectively. For example, a few meters works well for the walk variances on $\boldsymbol{\xi}_{x,y}$, and we use a few degrees for the orientation φ.

The MH scheme, as it is presented, takes a notoriously long time to converge. Hence, it is necessary to speed it up for a real-time application. Pseudo code is given for the classical MH algorithm for the calibration problem. In Chapter 5 and [19], we outline a Mode-Hungry MH algorithm for accelerated convergence, which fits well to this problem

Table 7: Pseudo Code for the MH Calibration

- At time k, for each particle i $(i = 1, \ldots, N)$, $\boldsymbol{\xi}_i^{(k)}$:
  - i. generate a candidate $\boldsymbol{\gamma}_i$ using $q(\boldsymbol{\xi}_i, \boldsymbol{\gamma}_i)$ given in (131)
  - ii. estimate the time-reference frame for data synchronization using the proposed position, the source GPS track, and the speed of sound c
  - iii. calculate the DOAs, $\theta(t)$, using the motion compensated beamformer (111)
  - iv. calculate the acceptance ratio, where the target distribution $\pi(.)$ is as given in (118)

$$\alpha(\boldsymbol{\xi}_i, \boldsymbol{\gamma}_i) = \min\left(\frac{\pi(\gamma_i)}{\pi(\boldsymbol{\xi}_i)}, 1\right)$$

  - v. sample $u \sim \mathcal{U}(0, 1)$
  - vi. if $u \leq \alpha(\boldsymbol{\xi}_i, \boldsymbol{\gamma}_i)$, set $\boldsymbol{\xi}_i^{(k+1)} = \boldsymbol{\gamma}_i$, else, $\boldsymbol{\xi}_i^{(k+1)} = \boldsymbol{\xi}_i^{(k)}$.
- Repeat until convergence is reached.

since only one node position is searched. A grid-based initialization of this algorithm is also possible, as discussed in [31].

4.3.4 Performance of the DOA Calibration Algorithms

The DOA calibration algorithms use a sequence of estimated DOAs corresponding to a batch size of M samples, together with geometrical arguments, to estimate $\boldsymbol{\xi}$. In other words, the DOA at index i is estimated using M array samples corresponding to the i^{th} synthetic receiver, sampled at F_s. To evaluate the performance of the DOA calibration algorithms, the DOA estimation performance should first be related to the node array signal-to-noise (SNR) ratio. Note that the DOAs need to be estimated using (111) if the calibration source is moving relatively fast. Classical DOA estimators such as MUSIC and MVDR result in biased estimates due to the non-stationarity of the data caused by rapid target motion [18, 77]. Using the likelihood function in (102), we write the Fisher information for

the DOA at the beginning of the batch as

$$\begin{aligned} F_{\theta_1} &= E\left\{\left(\frac{\partial L(\theta_1)}{\partial \theta_1}\right)^2\right\} \\ &= -\frac{2}{\sigma_a^2}\sum_{t=1}^{M}\mathrm{Re}\left[\left(\frac{\partial \mathbf{a}_t}{\partial \theta_t}\frac{\partial \theta_t}{\partial \theta_1}\right)^H\left(\frac{\partial \mathbf{a}_t}{\partial \theta_t}\frac{\partial \theta_t}{\partial \theta_1}\right)\right], \end{aligned} \tag{119}$$

where $\frac{\partial \theta_t}{\partial \theta_1}$ is calculated from (112). The inverse of F_θ bounds the best achievable performance by an unbiased estimator, but in most cases, DOA estimation performance will be close to this bound for large M. Hence, it is reasonable to approximate the noise variance on the estimated DOAs as

$$\sigma_\theta^2 \approx -\frac{\sigma_a^2}{2}\left\{\sum_{t=1}^{M}\mathrm{Re}\left[\left(\frac{\partial \mathbf{a}_t}{\partial \theta_t}\frac{\partial \theta_t}{\partial \theta_1}\right)^H\left(\frac{\partial \mathbf{a}_t}{\partial \theta_t}\frac{\partial \theta_t}{\partial \theta_1}\right)\right]\right\}^{-1}. \tag{120}$$

This estimated noise variance (120) can be also used in the solution (115) to weight the SAR angles. As a rule of thumb, one can use $\sigma_{\theta_t}^2 \propto R_t$ for the field examples. Now, if we assume the DOA noise is independent from batch to batch and is Gaussian with zero mean and with variance of (120), then the DOA likelihood from (118) becomes

$$L(\theta|\boldsymbol{\xi}) \doteq -\frac{1}{2}\sum_{i=1}^{L}\frac{1}{\sigma_{\theta_i}^2}\left\{\theta_i + \varphi - \tan^{-1}\left(\frac{y - y_T^{(i)}}{x - x_T^{(i)}}\right)\right\}^2 \tag{121}$$

with $\theta = [\ \theta_1,\ \ldots,\ \theta_i,\ \ldots,\ \theta_L\ ]$, where i is the estimation batch index and $\sigma_{\theta_i}^2$ is estimated using (120). Then, the estimation bound on the parameter $\boldsymbol{\xi}$ becomes

$$CRB_{\boldsymbol{\xi}} = \left(\sum_{i=1}^{L}\boldsymbol{\gamma}_i\boldsymbol{\gamma}_i^T\right)^{-1} \tag{122}$$

where

$$\boldsymbol{\gamma}_i = -\frac{1}{\sigma_{\theta_i}^2}\left[1,\ \frac{y - y_T^{(i)}}{R_i^2},\ -\frac{x - x_T^{(i)}}{R_i^2}\right]^T \tag{123}$$

In the simulations section, we show that the above CRLB (122) is quite close to the bound that is calculated directly from the data likelihood. This, in turn, raises the question of the sufficiency of the DOA estimates in the calibration problem, which is covered in the next subsection.

4.3.5 Sufficiency of the Auxiliary DOA Estimates in the Calibration Problem

The parameter vector $\boldsymbol{\xi}$ affects the distribution of the observations $\mathbf{Y}_M$ defined by (84) through the distribution $p(\mathbf{Y}_M|\boldsymbol{\xi}, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t)$ given in (83). Therefore, in the calibration problem, the statistical behavior of the acoustic data constitutes the only information about the parameter $\boldsymbol{\xi}$ when there is a lack of a prior distribution on $\boldsymbol{\xi}$. Hence, if knowing β_i removes any dependence on $\boldsymbol{\xi}$ from the data distribution (83), then it can be said that β_i contains all the relevant information in the data that is useful in estimating the parameter $\boldsymbol{\xi}$ [63]. Hence, the objective is to prove the following equality:

$$p(\mathbf{Y}_M|\boldsymbol{\xi}, \beta_0, \ldots, \beta_L, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t) = p(\mathbf{Y}_M|\beta_0, \ldots, \beta_L, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t) \tag{124}$$

At this point, some assumptions need to be reiterated. The SAR angles β_i are only given at every other MQ samples of data (a total of L times), whereas the data likelihood depends on the parameter at every sample. This is not a problem if the assumption of straight-line motion (112) is true between GPS data points. Thus, the data likelihood can be written as

$$\begin{aligned} &p(\mathbf{Y}_M|\boldsymbol{\xi}, \beta_0, \ldots, \beta_L, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t) = \\ &\prod_{t=0}^{K-1} \frac{1}{\pi^P \sigma^{2P}} \exp\left[-\frac{1}{\sigma^2}\left\|\mathbf{y}(t) - \mathbf{a}(\pi + \beta^{(t)} - \varphi)s(t)\right\|^2\right], \end{aligned} \tag{125}$$

where

$$\begin{aligned} &\beta^{(t)} + \pi - \varphi = \tan^{-1}\left\{\frac{F_s \sin\theta_{t-1} + q^*(t-1)\cos\phi^*}{F_s \cos\theta_{t-1} + q^*(t-1)\sin\phi^*}\right\}, \\ &\text{with} \quad \theta_{t-1} = \beta^{(t-1)} + \pi - \varphi. \end{aligned} \tag{126}$$

Parameter ϕ^* is the approximate heading direction of the calibration source at time t and is calculated from the GPS data. Parameter q^* is the only parameter that depends on the node position (x, y) and can be approximated by

$$q^* \approx \left|\frac{F_s \sin(\beta_i - \beta_{i-1})}{(MQ-1)\sin(\beta_t - \phi^*)}\right|. \tag{127}$$

Hence, given the synthetic DOAs β_i, it is possible to remove the dependence of the data on the node position, but not on the node orientation. That is,

$$p(\mathbf{Y}_M|\boldsymbol{\xi}, \beta_0, \ldots, \beta_L, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t) = p(\mathbf{Y}_M|\varphi, \beta_0, \ldots, \beta_L, \boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_t) \tag{128}$$

Equation (128) implies that the estimated synthetic DOAs form a sufficient statistics for the acoustic node position. However, to also remove the dependence on the orientation angle φ, at least one local estimate of θ_t is required from the node.

4.4 Special Case

If the calibration source is moving at a constant speed, then an acoustic node can estimate its orientation angle given at least three DOA measurements. In addition, if the range to the calibration source is known at any time t, then it is possible to track the calibration source's position in space given only DOA measurements (Fig. 29). As far as array calibration is concerned, if the calibration source is moving along a straight-line, then given one absolute distance, it is possible to calibrate multiple nodes relative to one reference node by using only geometrical arguments. Figure 30 demonstrates this relative calibration concept.

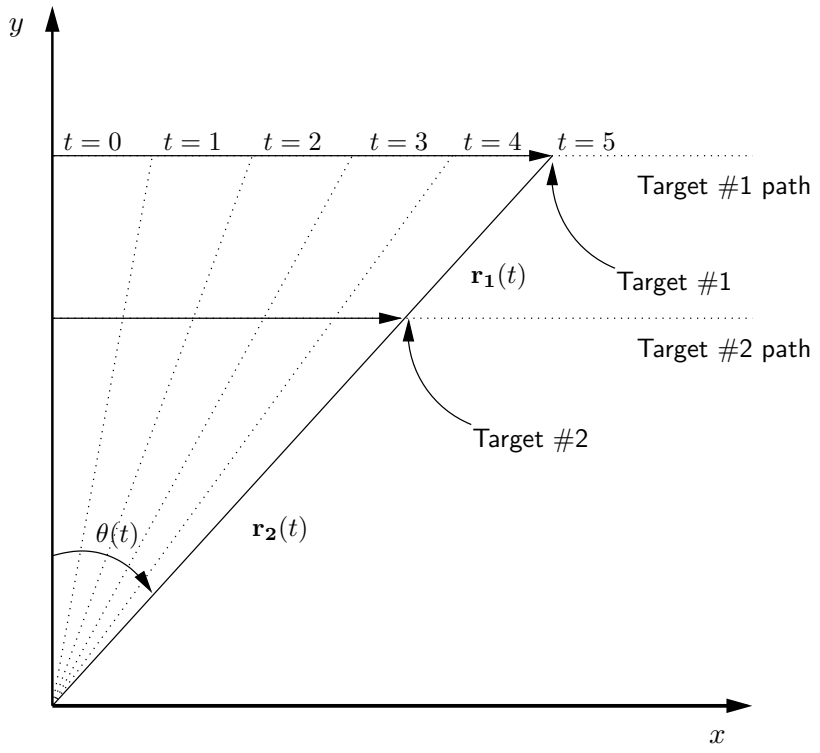


Figure 29: Two different targets may have the same DOA at all times. It is impossible to distinguish target #1 from target #2 given the DOA measurements alone. Hence, a range estimate is needed to track the target position correctly.

In Fig. 30, A_0B_0 and A_1B_1 are the track estimates of the reference node and node 1,

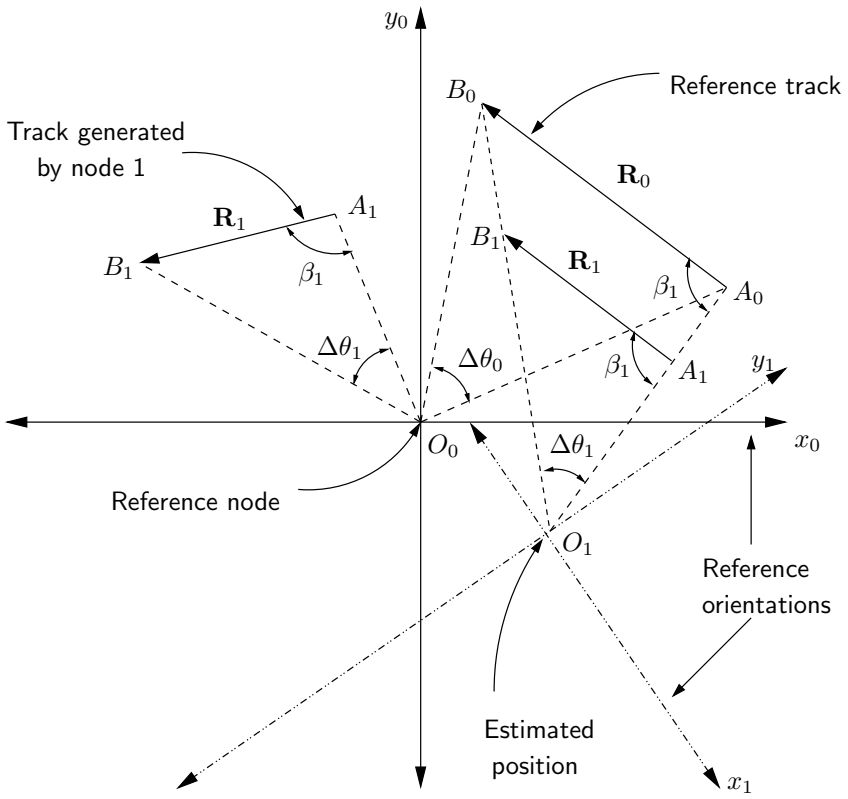


Figure 30: The track generated by node 1 is superposed on the reference node coordinate system. Note that both nodes are tracking scaled versions of the original calibration target track. The unknown node orientation, relative to the reference node, is simply the difference between the orientation estimates both nodes estimate. This is equivalent to finding the rotation angle that aligns tracks $\mathbf{R}_0$ and $\mathbf{R}_1$. Triangle $\Delta O_0 A_1 B_1$ is similar to $\Delta O_1 A_0 B_0$ because of the scaling property of the track estimates. Since the angles $\Delta\theta_1$ and β_1 estimated by node 1 are known, O_1 can be determined geometrically. The whole system is scalable and a reference distance is required for absolute estimates.

respectively. Since the track $\mathbf{R}_1$ and $\Delta\theta_1$ are known, the angle β_1 can be determined. Then, on the reference track $\mathbf{R}_0$, one can triangulate the sensor position O_1 by intersecting two lines from A_0 (A_0A_1) and B_0 (B_0B_1) using the angles ϕ_1 and $\pi - \phi_1 - \Delta\theta_1$, respectively. To find the reference angle for node 1, the difference of the orientation angles of the tracks $\mathbf{R}_0$ and $\mathbf{R}_1$ is used. Hence, the solution of the problem only requires estimation of the orientation angles of the tracks as well as one absolute distance. Some analytical relations are available in the literature to estimate the target orientation angle given three DOA estimates [77].

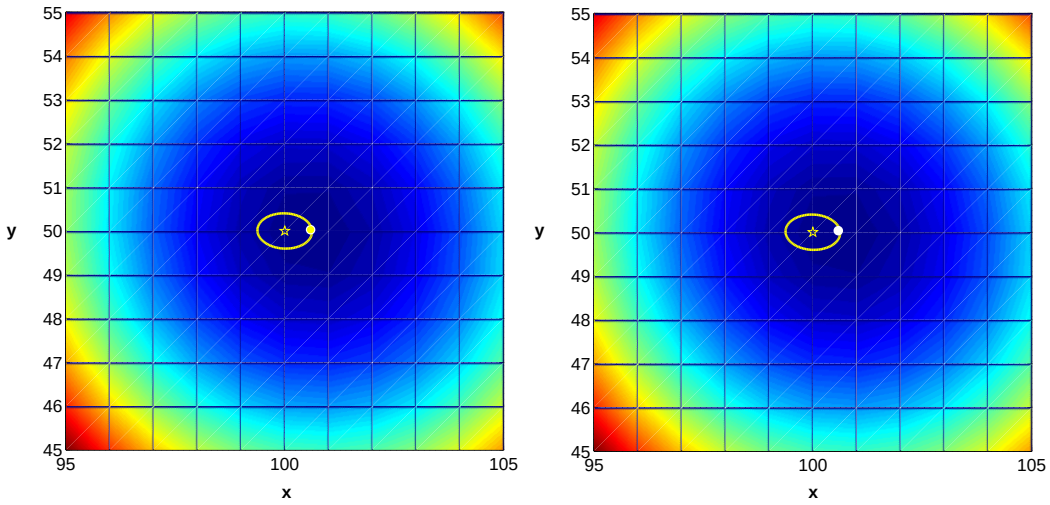
4.5 Simulations

Using computer simulations, our objectives are to (i) show the effects of the GPS errors on the estimation performance of the various algorithms, and (ii) demonstrate the ML and DOA calibration algorithms and compare their performance. All the simulation examples below use a calibration target circling the origin at a range of 600m with a constant speed of 75 mph. The acoustic node is a uniform circular array with $P = 6$ omnidirectional microphones with a radius of 1.22m. The total estimation time for the calibration is 120s and the GPS error standard deviation is $\sigma_\chi = 1$m. Depending on the scenario, different node positions, sampling rates, target narrow-band center frequencies, and node SNRs are used.

4.5.1 Effects of the GPS Errors on the Estimation Performance

It is interesting to plot the ML surface near the true node position (Fig. 31). The smooth convex shape of the surface justifies the argument that a Newton or gradient-descent type of search algorithm can be used to estimate the node parameters instead of calculating the whole surface. In Fig. 31, we try to determine the position of the acoustic node placed at [100, 50]m. The constant center frequency used for the calibration is $f_0 = 20$ Hz, the node sampling rate is $F_s = 128$ Hz, and the acoustic node SNR is 7 dB.

The similarity of Figs. 31 (a) and (b) confirms that the GPS errors have no significant effect on the ML surface used for the estimation. In Fig. 31, the estimated positions, with and without GPS errors, differ by less than a centimeter. Another interesting observation



(a) (b)

Figure 31: (a). The likelihood surface J with GPS errors. The GPS error standard deviation is $\sigma_\chi = 1$m and it is circularly symmetric in the x and y directions. The star indicates the true node position at $[\,100,\, 50\,]$m , whereas the dot is the estimated position $[\,100.5899,\, 50.0289\,]$m using the Newton algorithm. The ellipse is the Cramér-Rao bound on the position estimates. (b). The likelihood surface J without any GPS errors. The Newton method estimates the node position as $[\,100.5811,\, 50.0360\,]$m . The solutions are almost identical.

is that the estimated position lies rather close to the estimated CRLB for the problem. In Fig. 31, the bound is represented by an ellipse whose axes sizes are determined by the square root of the diagonal elements of the CRLB. The ellipse orientation is determined by the correlation terms in the CRLB.

For the Newton search algorithm, a constant step size of $\mu = 0.125$ is used, and the approximations suggested in Sect. 4.2.2 are implemented. As stopping conditions, the algorithm is run up to a specified number of iterations (e.g., 1000 iterations) or until the following difference condition is met

$$\left|\boldsymbol{\xi}_k - \boldsymbol{\xi}_{k-1}\right| < 10^{-5} \tag{129}$$

In the simulations performed, the algorithm never reached the iteration limit when initialized within 5 meters of the true position.

4.5.2 Performance Comparisons

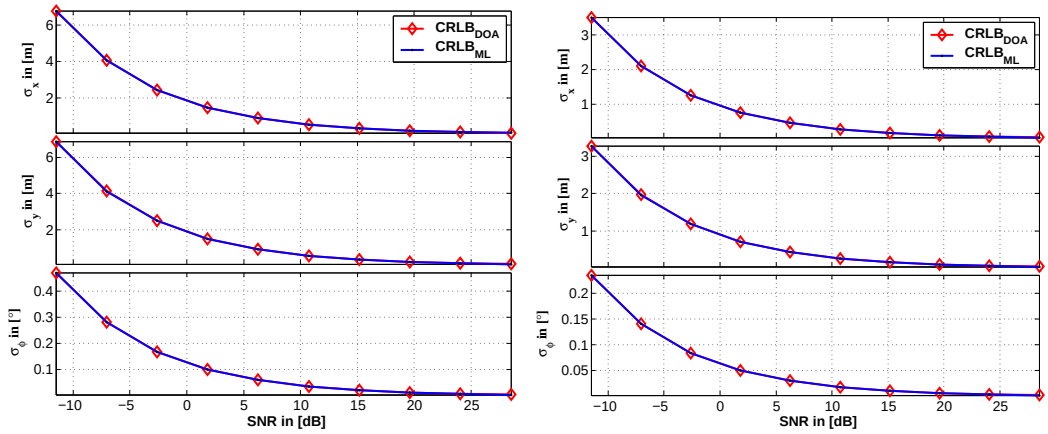
In this section, we first compare the two CRLBs derived directly from the array model and through the DOA estimation. Figure 32 shows that both bounds are very close. Intuitively, this may suggest a similar performance for the ML calibration algorithm and the DOA calibration algorithms. However, Fig. 33 shows that this is not the case. In Fig. 33, the synthetic aperture method (solid curve) performs better than the ML calibration algorithm (dashed curve).

An efficient unbiased estimator has a performance bound defined by the CRLB. It is also known that if an estimator achieves the CRLB, then it is the ML estimator. In our problem, the ML estimation algorithm achieves the CRLB as the number of data points are increased. However, the synthetic aperture method beats the bound in many cases (at lower SNRs). This implies that it is a biased estimator whose total minimum mean squared error (including the bias term) is smaller than the ML estimator. Such estimators do exist but are hard to find [63]. The bias is introduced by (114) since the relationship is not one-to-one. The analysis of this bias is difficult because there is not a proper probability density function for the node positions given (114).

The CRLBs in Fig. 32 also demonstrate the dependency of the calibration performance to the target center frequency. As the target center frequency increases, the performance is expected to get better since the decreased wavelength results in better DOA estimations.

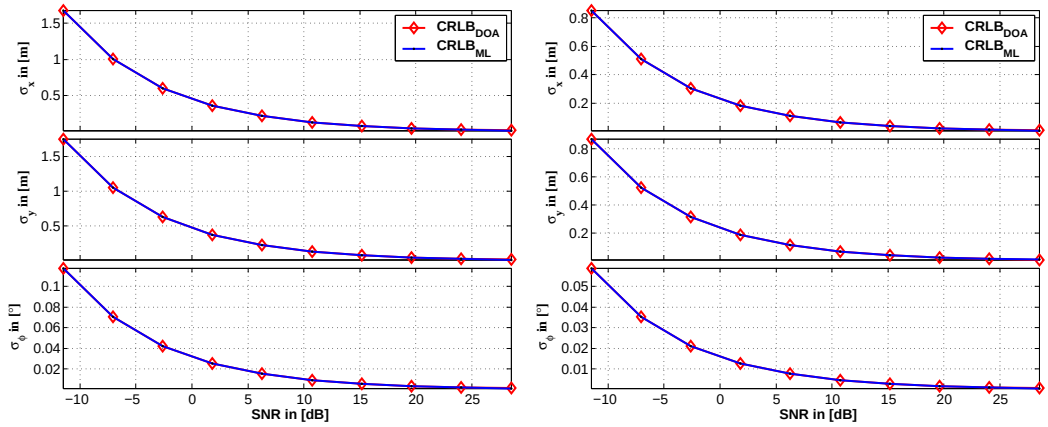
4.6 Field Data Results

We applied the Monté-Carlo calibration scheme on field data from a small acoustic array. A helicopter flew sorties around the acoustic node for the calibration purposes (Fig. 34). For the experiment, the array was hand-emplaced so that the true true location and orientation would be known. The acoustic node has six omnidirectional microphones placed uniformly on a circle with a radius of 1.219m, which also corresponds to the inter-microphone distance. The spatial aliasing frequency is around 135Hz corresponding to the half-wavelength (equal to the radius of the array). For the MVDR beamforming results, we tracked the ten highest peaks using the short time Fourier transform (Fig. 35) with their respective heights, and



(a) $f_0 = 20$ Hz

(b) $f_0 = 40$ Hz



(c) $f_0 = 80$ Hz

(d) $f_0 = 160$ Hz

Figure 32: The Cramér Rao lower bounds' dependency on the calibration target center frequency. The calculated CRLB from the ML formulation is shown with the dotted line; the CRLB from the DOA calibration formulations is shown with the diamonds. The bounds are exceptionally close even though they are derived from different formulations, as indicated earlier.

averaged the estimated DOAs accordingly. For the time synchronization, third-order b-splines were used to interpolate the irregular time-grid for the algorithm's proposed positions for each particle.

The GPS track of the helicopter used for the calibration is shown in Fig. 34. Figure 33 demonstrates the results of the Metropolis-Hastings scheme with the Mode-Hungry modification in Chapter 5. A listing of the pseudo-code for the MH scheme is given. To establish a baseline, we also determine the CRLB to approximate the lower bounds for the variances for the parameter vector as $[\sigma_x, \sigma_y, \sigma_\theta] = [1.3931\text{m}, 1.3597\text{m}, 0.13296°]$.

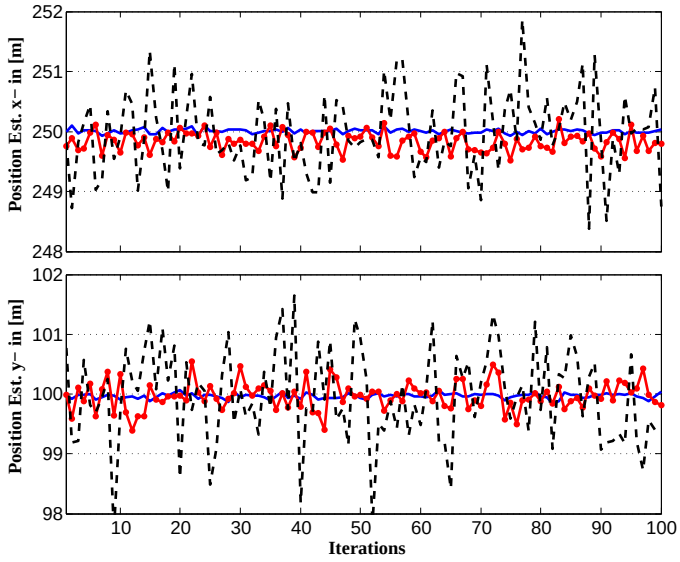


Figure 33: The node is situated at $[\,250,\,100\,]$m. The calibration is repeated 100 times with different noise realizations at SNR $= 8.5$ dB. The solid line is the performance of the synthetic aperture method, where (115) is used to solve for the position. The dotted solid line is also the synthetic aperture method, except the least squares (113) approach is used for the solution. Full ML estimates using the acoustic data are shown with the dashed line.

Note that the approximate bound assumes the single sinusoid array model for the observations. This assumption is not satisfied for the helicopter signal. In Fig. 33, the DOAs estimated with the field data between $t = 50$s and $t = 120$s violates the Gaussian assumption since they have the incorrect mean. Without post-processing of the data, it is difficult to detect this incorrect mean since the orientation estimates can also bias the DOA distribution. The algorithm still managed to do well because the target distribution weights the calculated DOAs according to their estimated range. When the target is far away from the node, the estimates are expected to get worse. Hence, the algorithm puts less importance on the DOA estimates corresponding to large target ranges. Lastly, Fig. 36 also demonstrates that the calibration results are significantly worse without the time synchronization step. Interestingly, for this test run, the acoustic node also used low-cost a GPS system to determine its position. The GPS estimated position had a 7m error, implying that the MH scheme performed within the ballpark.

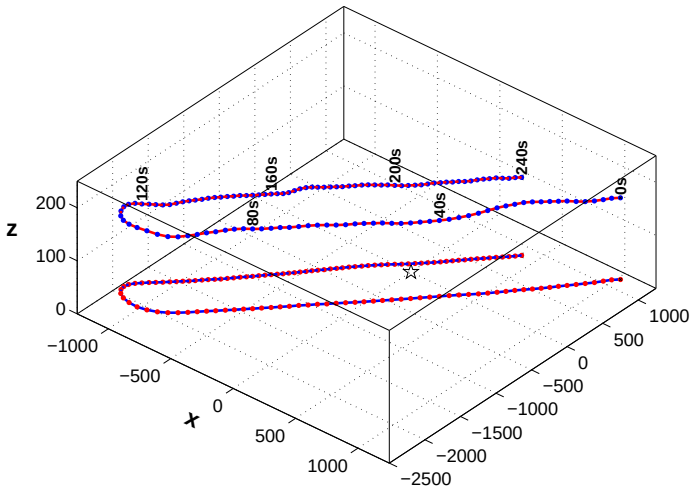


Figure 34: The acoustic node (star) is situated at the origin. The calibration helicopter completes two sorties around the node for the calibration, corresponding to a four-minute run. The actual helicopter track as well as its projection on the x-y plane are shown.

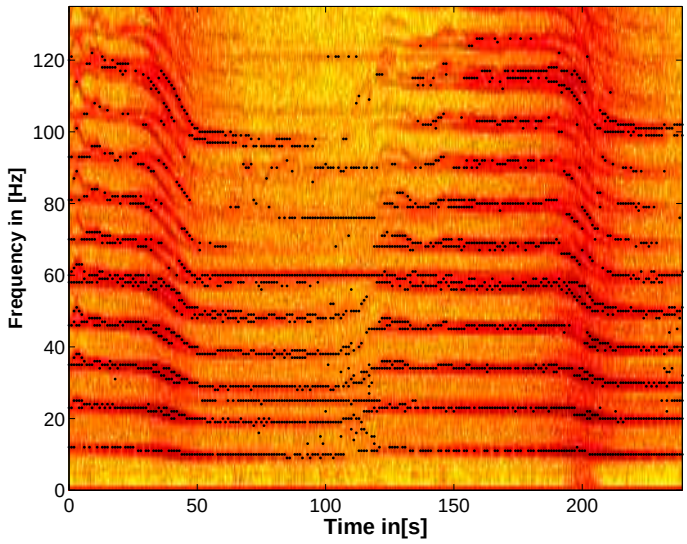


Figure 35: The helicopter spectrum displays strong harmonic lines. The ten highest peaks in the time-frequency plane are picked using the magnitude of the Fourier transform once per second. These frequencies as well as their time-frequency amplitudes are used in determining the DOA estimates.

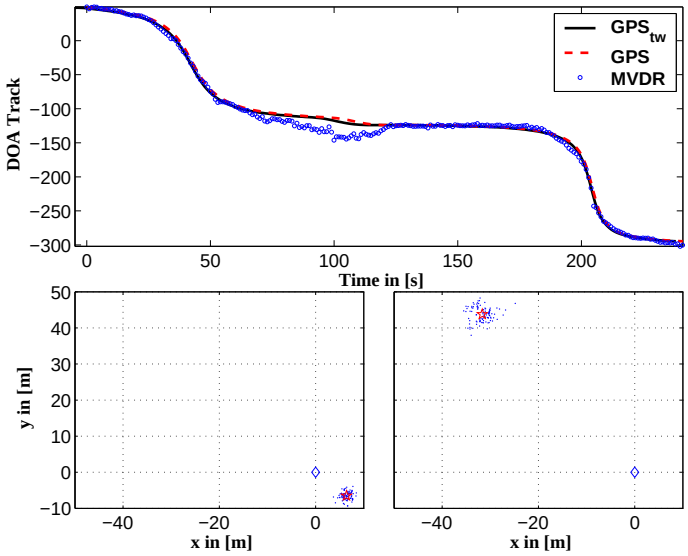


Figure 36: Top figure shows the GPS track coming from the helicopter (dashed line), the time-warped GPS track using (106), and the MVDR beamformer estimates of the field data. The bottom left plot is the resulting MHMH algorithm distribution with estimate $\xi = [\ 6.43,\ -6.52\ ]$m using time synchronization, whereas, at the bottom right, the distribution with estimate $\xi = [\ -31.66,\ 43.77\ ]$m is the MHMH result without time synchronization. The estimated orientation for both cases is 1°. The true node location is shown with the diamond.

4.7 Conclusions

Various node calibration algorithms were demonstrated given a calibration target with a GPS system. The maximum-likelihood solution was demonstrated and compared to the biased synthetic aperture method, which has a lower minimum mean squared error. The DOA calibration algorithms can also compensate for array non-stationarity problems using the methods outlined. Among the calibration methods, the MH calibration algorithm is the most flexible and suitable for field data processing because it can incorporate time synchronization as well as motion compensation directly into the calibration. The proposed algorithm performances are compared, with simulation examples.

CHAPTER V

FAST INITIALIZATION OF PARTICLE FILTERS USING A MODIFIED METROPOLIS-HASTINGS ALGORITHM: MODE-HUNGRY APPROACH

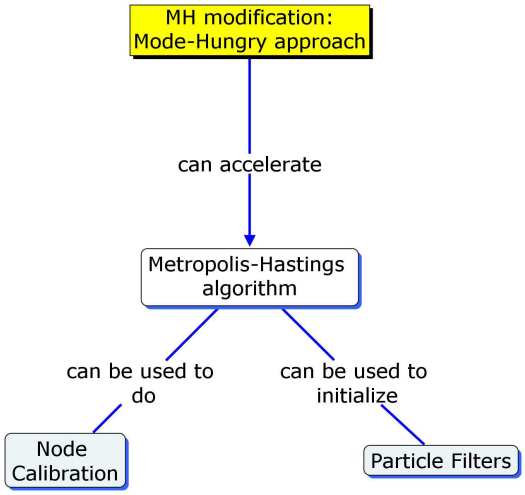


Figure 37: Chapter outline with the new contributions shown in rectangular boxes.

5.1 Introduction

Particle filters use Bayes' rule to update an arbitrary posterior recursively as the observations arrive. In the filter mechanics, posteriors are represented by particles that are discrete realizations distributed according to the underlying distribution (either directly or by weighting). Hence, the particle filter can estimate any statistics of the posterior, and the estimation accuracy can be improved up to the theoretic bounds by increasing the number of particles. Particle filters are becoming widely popular in signal processing due to (i) advances in computer systems, and (ii) their easy implementation [5, 25].

As a recursive algorithm, the particle filter needs initialization (i.e., initial samples) to track a state vector as the observations arrive in sequence. The algorithm's estimation

performance not only depends on how it is constructed around the data models, but also on its initial samples. Hence, it is necessary to generate good initial samples from the data itself. If the initial samples do not sufficiently cover the state space, important features of the distribution (such as multimodality) can be missed. This can significantly degrade the estimation performance of the filter.

Monté Carlo Markov chain (MCMC) simulation techniques offer attractive solutions to this type of sampling problem. In MCMC terms, the initialization problem becomes the estimation of a posterior density $\pi(x)$ by simulating a Markov chain $x^{(1)}, x^{(2)}, \ldots, x^{(t)}$ whose stationary distribution is the target posterior π as $t \to \infty$. Among the MCMC sampling techniques, the Metropolis-Hastings (MH) algorithm is used in this chapter to demonstrate the concepts and the results.

The MH scheme [37, 51] provides a basis from which other well-known sampling algorithms such as Acceptance-Rejection (AR) and Gibbs sampling can be derived as special cases [24, 75]. The algorithm assumes that it is possible to assign probabilities from the distribution to a given realization x. These probabilities need not be exact; they can also be given up to a proportionality constant. Note that calculating probabilities given the particles is different (and much easier) than generating particles directly from the (possibly intractable and multivariate) distribution itself.

At each iteration of the algorithm, a candidate generating density $q(x, y) : x \to y$ is used to propose moves (or *jumps*) within the state space. These moves are accepted or rejected with probability $\alpha(x, y)$ so that the following reversibility condition is satisfied:

$$\pi(x)q(x, y)\alpha(x, y) = \pi(y)q(y, x) \tag{130}$$

The reversibility condition is also known as *detailed balance*, *microscopic reversibility*, or *time reversibility* and is a necessary condition the chain must satisfy to have a stationary distribution π [24].

The convergence analysis of the MH algorithm is an open problem. Many methods have been proposed to detect the convergence of this scheme [15, 31, 32]. Although these convergence detection methods are useful, the MH algorithm usually takes a large number of

iterations to converge, and hence it is usually not considered for real-time applications in its original form. To make the algorithm more attractive, there has been some effort on making the algorithm more efficient. Two fundamentally different approaches have been proposed to speed up the MH scheme: One approach adaptively alters the jumping rules (i.e., the jump size) defined by the candidate generating density. The other approach uses a mode-search algorithm (usually a grid search) to locate the modes and samples from a mixture of suitable distributions located at these modes to generate a prior for the algorithm. The first method slightly improves the convergence speed, but not enough for real-time initialization. The latter method incurs a high computational cost to begin, and is therefore not suitable for our purposes.

In this chapter, we propose a new approach called Mode-Hungry MH (MHMH) to speed up the original scheme for the initialization of particle filters. The new method modifies the jumping rules according to how the state of the Markov chain is distributed and concentrates the particles on high probability regions. The algorithm quickly converges on the multimodal regions of the posterior. The estimated posterior is an approximation to the true posterior around the mode. However, it can be shown that any discrepancies between the estimated and true posteriors can be handled by the particle filter's built-in weighting structure [5, 25]. This property of the particle filters and the convergence speed makes this algorithm attractive for particle filtering applications.

The organization of this chapter is as follows. Section 5.2 provides the background for the MH algorithm. Section 5.3 describes the modification of the basic algorithm and gives pseudo-code for the MHMH algorithm. Then, an MH algorithm is described for particle filter initialization for DOA tracking in Sect. 5.4. Computer simulations are given in Sect. 5.5.

5.2 Background

The Metropolis-Hastings scheme [24] depicted in Fig. 38 attempts to distribute particles (discrete state samples) according to a target distribution π shown at the top. At each iteration, algorithm recursively redistributes its states so that, asymptotically, the resulting

Markov chain is distributed according to the target distribution. In Fig. 38, the Markov chain at iteration t is represented by $x^{(t)}$. The new chain candidates y are generated by the proposal function $q(x, y)$, which is often taken to be the spherically symmetric random walk:

$$q(x, y) = q(|x - y|) \propto \exp\left\{-\frac{(x-y)^2}{2\sigma^2}\right\}. \tag{131}$$

Once the new candidates are generated, the algorithm accepts the candidates or keeps the current state according to the acceptance ratio $\alpha(x, y)$ derived from the reversibility condition:

$$\alpha(x, y) = \min\left\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\right\}. \tag{132}$$

In Fig. 38, the acceptance ratios are represented by the height of the boxes for each candidate. To accept or reject a new candidate, a random number generator is used to generate uniform random numbers in $(0, 1)$, $u \sim \mathcal{U}(0, 1)$, represented by the black dots in the figure. If u is less than the acceptance ratio for the specific particle, the move is accepted, otherwise, it is rejected. The *acceptance rate* at iteration t is defined to be the number of accepted moves divided by the chain size. Visually, it is the number of arrows in the last stage in Fig. 38 divided by the number of particles. Finally, the chain moves to $x^{(t+1)}$ and the scheme is repeated.

Note that the candidate generating (or proposal) function $q(x, y)$ has a significant impact on the efficiency of the algorithm. It should be constructed so that the generated candidates display most of the structural dependence between the different dimensions. σ in (131) is defined as the jump size and is the other important variable affecting the algorithm speed. If σ is too small, the algorithm takes longer to converge since the chain moves very slowly along the target distribution. On the other hand, if σ is too large, the algorithm mostly rejects the new candidates and stays frozen. The current MCMC literature concentrates on these two important factors in the algorithm's efficiency [31].

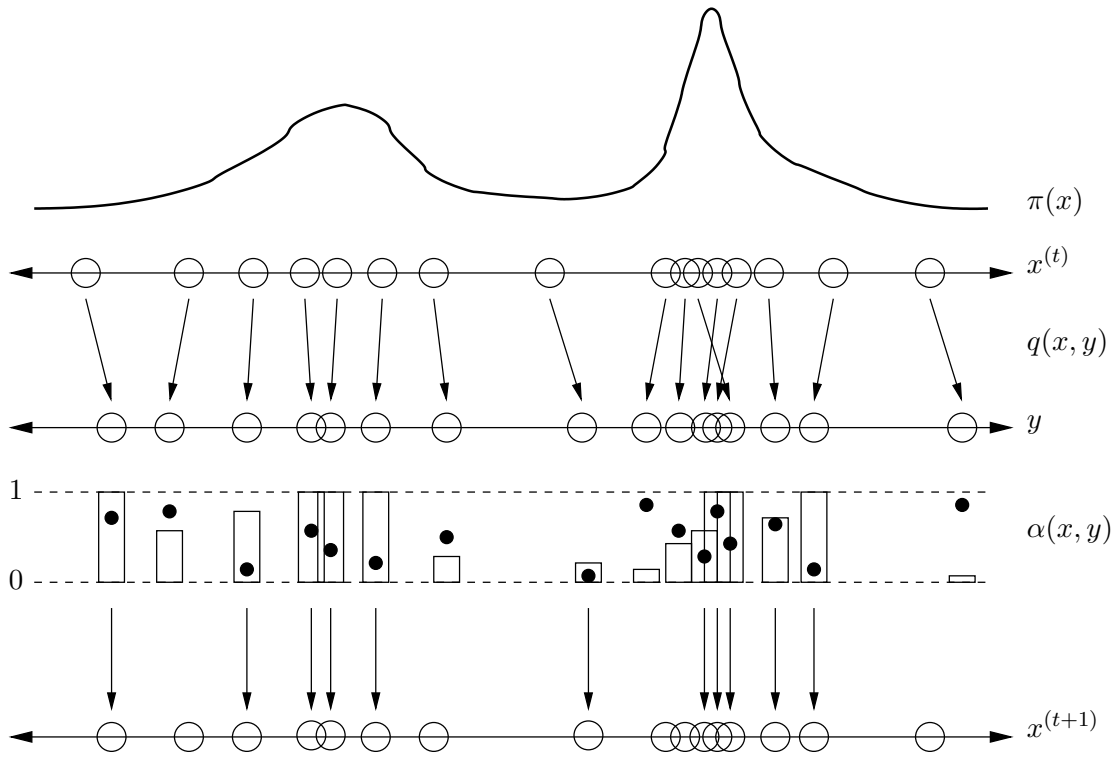


Figure 38: The Metropolis-Hastings scheme is demonstrated. Each circle represents a sample from the chain in the respective state space. The algorithm uses its current state to generate new candidates for its next state using a candidate generating function q. The new candidates are accepted or rejected in a way that the Markov chain asymptotically converges to the target posterior π.

5.3 Mode-Hungry Approach

The MH scheme is naturally rather slow since it tries to explore the whole parameter space. Hence, any relevant information should be used to increase the convergence speed. The proposal functions should incorporate any dimensional dependence, and the jump sizes should be modified by monitoring the acceptance rate of the algorithm. In this section, we will modify the MH algorithm to concentrate around the modes of the target posterior. We call this modification the *Mode-Hungry* approach. Interestingly, note that the adaptive modification of the simulation gives rise to a non-Markov chain since the transition probabilities depends on the previous iteration results. However, the resulting performance justifies their use.

In most cases, the target distribution is multimodal with a significant number of small modes and a few large modes. The small modes can be due to (i) the nature of the problem,

(ii) modelling errors, and (iii) an approximation of the actual target density π used in the simulation. In most particle tracking problems, the large modes are of interest since the small modes get rejected by resampling (or weighting) after a few iterations of the filter. Moreover, the small modes tend to trap the chain and increase the convergence time. It takes the MH scheme a notoriously high number of iterations to break the particles from these small modes and concentrate them on the higher ones [26]. Hence, when generating the target density π, there is a trade-off between how closely it can be represented versus how long the initialization takes to converge.

With this observation, we suggest that if the objective of the initialization is to cover only the high probability modes, the particles around the small modes should be redistributed accordingly during the MH iterations. This, in turn, will result in faster coverage of the high modes and improve the convergence speed. Hence, at time t, the chain $x^{(t)}$ consisting of N particles is separated into two partitions after being sorted with respect to likelihoods. Denote P_1 as the set corresponding to the $N - M_t$ high-probability particles, and P_2 as the M_t low-probability particles. During the algorithm execution, a rule needs to be determined to distribute the particles from P_2 around P_1 so that the algorithm does not get stuck at low probability modes. We propose using randomly chosen particles from P_1 as the candidates y for the particles from P_2, and accept their moves with probability 1.

Figure 39 illustrates this Mode-Hungry approach. The target particles from P_1 can be chosen in a variety of ways, two of which are considered here: proportional to their probabilities and uniformly. If we insist that the cross-jumps from P_2 to P_1 should be such that particles around one mode should not be quickly consumed by another similar size mode, then it is not prudent to pick particles from P_1 proportional to their probabilities. In this case, the highest mode quickly accumulates all the particles and the algorithm results in a single mode distribution. Hence, this rule is recommended if the target posterior is known to be unimodal.

If the particles from P_2 are uniformly distributed over P_1, the algorithm quickly finds the multiple modes; however, the distributions around the modes can be skewed if they are far apart from each other, as illustrated in the next section. This is still acceptable, since

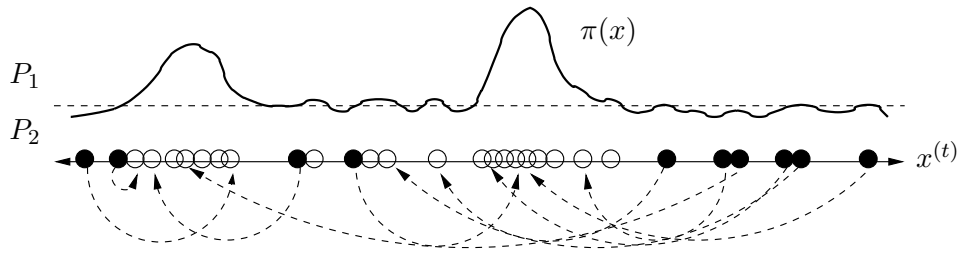


Figure 39: Particles corresponding to the partition P_2 are shown in black. In the figure, the particles in P_2 are distributed over the states defined by P_1 uniformly.

any imbalance of the particle distribution will be automatically corrected by the particle filter's weighting mechanism. Note that a good local distribution around the mode is still required to prevent sample degeneracy [5]. The particle filter attaches a weight w_i to each particle x_i coming from the MH scheme. Then, it estimates an expected value $E[.]$ of the posterior as

$$E[f(x)] = \sum_i w_i f(x_i), \tag{133}$$

assuming that the weights are normalized. Hence, as long as the region around the modes are covered correctly, the weights can correct the imbalances introduced by the initialization since they also depend on the target density π. Note that the original MH scheme also has similar issues in multimodal problems; however, the proposed cross-jumps enable the algorithm to cover the high modes faster. The frequency of these jumps can be determined by monitoring the acceptance rates or how tightly the particles distribute themselves around the modes; however, a fixed period (T_{jump}) also seem to work. The pseudo-code gives the details of the proposed algorithm in Table 8.

5.4 Acoustic Target Tracker Initialization

In this section, we give an example MH initialization algorithm that can initialize the particle tracker discussed in Chapter 3. The problem is to generate discrete samples for the following state vector used in the acoustic tracking problem:

$$x(t) = [\ \theta(t)\ ,\ Q(t)\ ,\ \phi(t)\ ]^T \tag{134}$$

where $\theta(t)$ is the target direction-of-arrival (DOA) measured clockwise with respect to the y-axis, and $\phi(t)$ is the target heading direction measured counterclockwise with respect to the

Table 8: Mode-Hungry Metropolis-Hastings

- At time t, decide if cross-jumping is needed for $x^{(t)}$, i.e., every T_{jump} iterations
- If not, then for each particle x_i, use the MH scheme:
  - generate a candidate y_i using $q(x_i, y_i)$
  - calculate the acceptance ratio

$$\alpha(x_i, y_i) = \min\left\{1, \frac{\pi(y_i)q(y_i, x_i)}{\pi(x_i)q(x_i, y_i)}\right\}$$

  - sample $u \sim \mathcal{U}(0, 1)$
  - if $u \leq \alpha(x_i, y_i)$, set $x_i^{(t+1)} = y_i$, else, $x_i^{(t+1)} = x_i^{(t)}$.
- If yes, then use the Mode-Hungry scheme:
  - determine a subpartition of size $M_t < N$
  - order the current particles according to their probabilities in descending order: $x_i \rightarrow x_j^*$ where x^* is the ordered particle set
  - generate candidates $y^*(1)$ for $x^*(1) = \{x_j^* | j : j = 1, 2, \ldots, N - M_t\}$ using $q(\cdot, \cdot)$
  - calculate the acceptance ratio $\alpha(x^*(1), y^*(1))$ and set $x_j^{(t+1)}$ to $x_j^*(1)$ or $y_j^*(1)$ accordingly for $j = 1, 2, \ldots, N - M_t$
  - distribute M_t candidates $y^*(2)$ from $x^*(1)$ uniformly
  - set $x_j^{(t+1)}$ to $y^*(2)$ for $j = N - M_t + 1, \ldots, N$

x-axis. The parameter $Q(t)$ is the logarithm of the target velocity-range ratio. The state-update was derived using constant velocity motion assumption, and geometrical arguments in Chapter 2:

$$\begin{aligned}\mathbf{x}(t+T) &= h(\mathbf{x}(t), T) \\ &= \begin{bmatrix} \tan^{-1}\left(\frac{\sin\theta(t)+e^{Q(t)}T\cos\phi(t)}{\cos\theta(t)+e^{Q(t)}T\sin\phi(t)}\right) \\ Q(t) - 1/2\log\left[1+2e^{Q(t)}T\sin\left(\theta(t)+\phi(t)\right)+(e^{Q(t)}T)^2\right] \\ \phi(t) \end{bmatrix}\end{aligned} \tag{135}$$

Let $\mathbf{y}(t)$ be a vector of the measured DOAs:

$$\mathbf{y}(t) = \left[\ \hat{\theta}(t)\ ,\ \hat{\theta}(t+T)\ ,\ \ldots\ ,\ \hat{\theta}(t+(M-1)T)\ \right]^T \tag{136}$$

where $M \geq 3$ is an integer and T is the period at which the DOAs are measured. Then, by using Bayes' rule, the target distribution is given by

$$\pi(\mathbf{x}(t)) = p(\mathbf{x}(t)|\mathbf{y}(t)) \propto p(\mathbf{y}(t)|\mathbf{x}(t)) \tag{137}$$

when there is no prior information on the state vector.

Assuming that $p(\mathbf{y}(t)|\mathbf{x}(t))$ is available[1], the MH algorithm to generate the initial state distribution is given in Table 9. This formulation is very similar to the one by Orton [59].

5.5 Examples

In this section, our objectives are (i) to demonstrate that the proposed method results in faster convergence, and (ii) to show that the algorithm can handle multimodal distributions. In the simulations, the initial chain values are generated from a uniform distribution $x_i \sim \mathcal{U}(-50, 50)$ in each dimension.

[1]Chapters 2 and 3 have derived the analytical relationships for this pdf.

Table 9: Metropolis-Hastings Initialization Algorithm

---

i. Sample $\mathbf{x}_1^{(k)}(t) \sim \mathcal{N}\left(\mathbf{x}_1^{(k-1)}(t), \sigma_\theta^2\right)$

ii. Sample $\mathbf{x}_3^{(k)}(t) \sim \mathcal{N}\left(\mathbf{x}_3^{(k-1)}(t), \sigma_\phi^2\right)$

iii. Sample $u_\theta \sim \mathcal{N}\left(0, \sigma_\theta^2\right)$ and set

$$\mathbf{x}_2^{(k)}(t) = \log\left|\frac{\sin(h\left(\mathbf{x}_1^{(k-1)}(t), (M-1)T\right) + u_\theta - \mathbf{x}_1^{(k)}(t))}{(M-1)T\cos(\mathbf{x}_3^{(k)}(t) + u_\theta + \mathbf{x}_1^{(k-1)}(t))}\right|$$

iv. Calculate the acceptance ratio

$$\alpha(\mathbf{x}^{(k)}(t), \mathbf{x}_1^{(k-1)}(t)) = \min\left\{1, \frac{\pi(\mathbf{x}^{(k)}(t))q\left(\mathbf{x}_2^{(k-1)}(t)|\mathbf{x}_1^{(k-1)}(t), h\left(\mathbf{x}^{(k)}(t), (M-1)T\right), \mathbf{x}_3^{(k-1)}(t)\right)}{\pi(\mathbf{x}^{(k-1)}(t))q\left(\mathbf{x}_2^{(k)}(t)|\mathbf{x}_1^{(k)}(t), h\left(\mathbf{x}_1^{(k-1)}(t), (M-1)T\right), \mathbf{x}_3^{(k)}(t)\right)}\right\}$$

where

$$q\left(\mathbf{x}_2^{(k)}(t)|\mathbf{x}_1^{(k)}(t), h\left(\mathbf{x}_1^{(k-1)}(t), (M-1)T\right), \mathbf{x}_3^{(k)}(t)\right) =$$
$$\frac{(Ts-1)T}{\mathbf{x}_2^{(k)}(t)\sqrt{2\pi\sigma_\theta}} e^{-\frac{u_\theta^2}{2\sigma_\theta^2}} \frac{\cos^2(\mathbf{x}_3^{(k)}(t) + u_\theta + \mathbf{x}_1^{(k-1)}(t))}{\left|\cos(\mathbf{x}_3^{(k)}(t) + \mathbf{x}_1^{(k)}(t))\right|}$$

- sample $u \sim \mathcal{U}(0,1)$
- if $u \leq \alpha(\mathbf{x}^{(k)}(t), \mathbf{x}^{(k-1)}(t))$, accept the move, else, $\mathbf{x}^{(k)}(t) = \mathbf{x}^{(k-1)}(t)$.

5.5.1 Single Mode Gaussian Example

$N = 1000$ particles are simulated to obtain a three-dimensional Gaussian distribution: $\pi(\mathbf{x}) \sim \mathcal{N}(\mu, \mathbf{\Sigma})$. The mean and covariance of the distribution are the following:

$$\mu = \begin{bmatrix} 20 \\ -15 \\ 45 \end{bmatrix} \quad \mathbf{\Sigma} = \begin{bmatrix} 1 & 0 & 0.5 \\ 0 & 2 & -.5 \\ 0.5 & -.5 & 0.5 \end{bmatrix} \tag{138}$$

For the Mode-Hungry MH (MHMH) algorithm, M_t is chosen to be 333. An independent random walk is used for the proposal function:

$$q(\mathbf{x}, \mathbf{y}) \propto \exp\left\{ -\frac{1}{2} \sum_{i=1}^{3} \frac{(x_i - y_i)^2}{\sigma_i^2} \right\}, \tag{139}$$

with $\sigma_1^2 = \sigma_2^2 = \sigma_3^2 = 6$. The particles belonging to P_2 are distributed over P_1 at every other iteration of MHMH ($T_{jump} = 2$) until the variances of the distribution come close to the search variances of the proposal function. Fig. 40 displays the estimation results of both algorithms. In the figure, it should be noted that MHMH achieves convergence around the modes much faster than the original MH scheme. Moreover, both algorithms take approximately $30 - 50$ iterations to achieve the correlation structure in (138) after convergence around the modes.

5.5.2 multimodal Example

To demonstrate efficacy of the proposed modifications on multimodal distributions, the following target density is used:

$$\begin{aligned} \pi(\mathbf{x}) \propto & \left(e^{-(x_1-20)^2/2} + \frac{1}{2} e^{-(x_2+20)^2/2} \right) \\ & \times \left(\frac{1}{2} e^{-(x_1-30)^2/10} + e^{-(x_2+45)^2/10} \right) \end{aligned} \tag{140}$$

The same set of cross-jumping rules are used as in the previous example ($N = 1000$, $M_t = 333$, and $T_{jump} = 2$). In this example, the random walk proposal is used with $\sigma_1^2 = 5$ and $\sigma_2^2 = 5$ as the respective variances in the x_1 and x_2 directions. The resulting distribution after 30 iterations of MHMH is shown in Fig. 41.

It should be noted in Fig. 41 that the local distributions of the particles around the modes are correct. However, the number of particles attracted to each mode does not obey

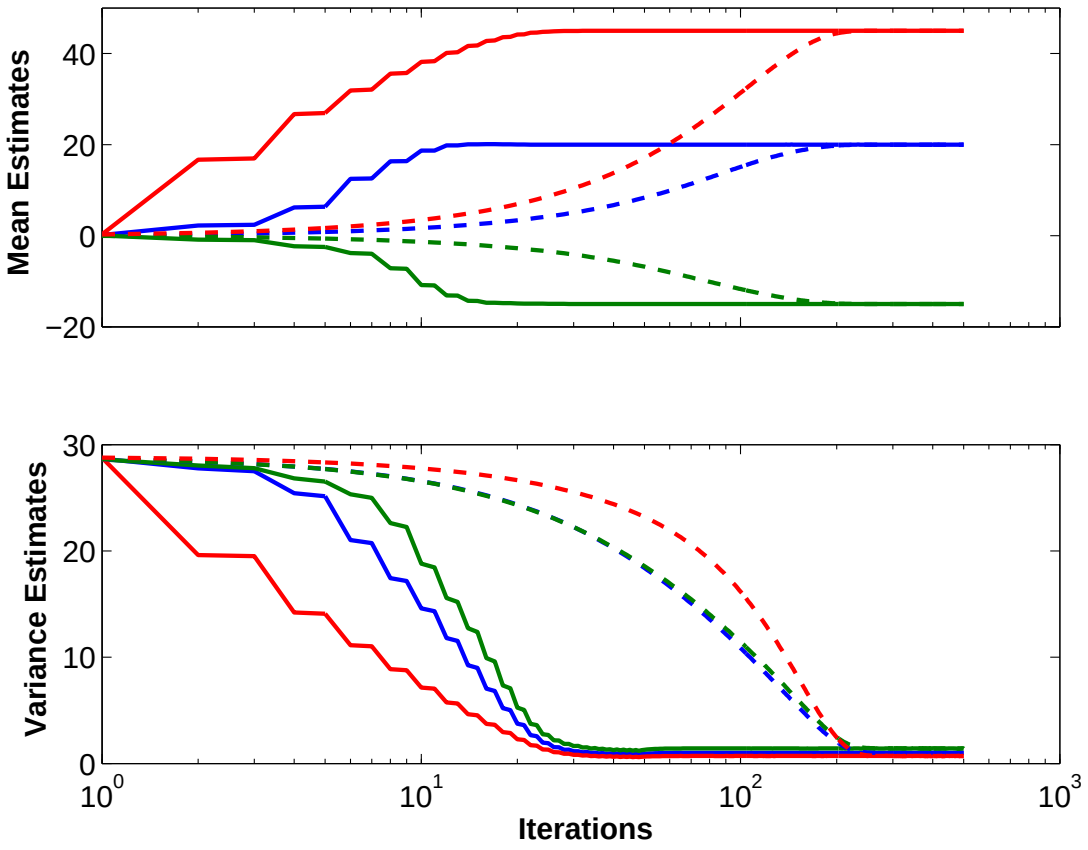


Figure 40: MHMH (solid) and MH (dashed) algorithms are run 100 times and the averaged estimates are displayed. MHMH is about an order of magnitude faster in this case. The jumps in the MHMH curves are attributed to the redistribution of the low probability particles over the high ones.

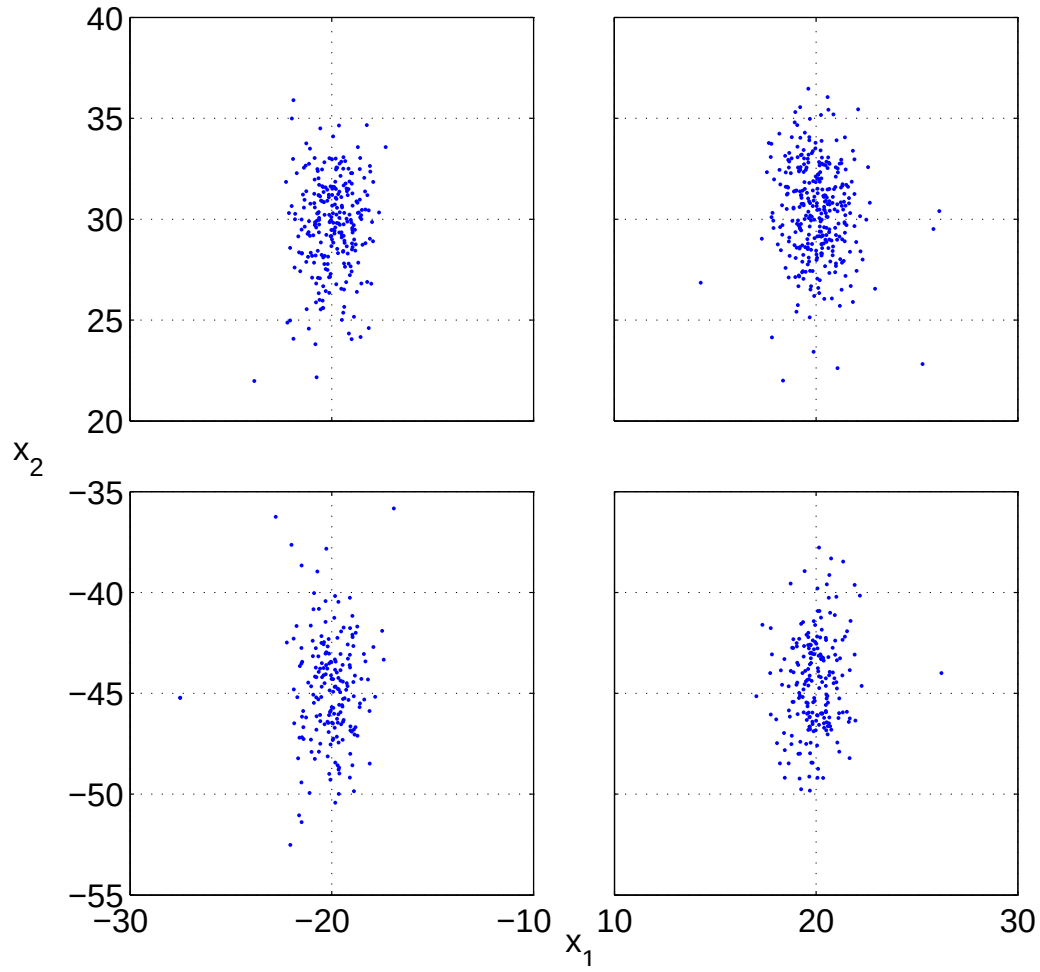


Figure 41: Output distribution of MHMH after 30 iterations. A similar distribution can be obtained by the MH scheme using approximately 100 iterations.

the global distribution (two of the modes are higher than the others; hence, they should attract more particles.) This is also a common issue in the MH scheme, where it takes a high number of iterations for the MH algorithm to break particles from different modes to correctly concentrate them on different modes. This output is still suitable for a particle filter because of the correct local distributions. The inherent weighting of the particles would take care of any discrepancies in the global distribution.

5.6 Conclusions

In this chapter, the MH algorithm is modified to achieve faster convergence for the initialization of particle filters. The modification depends on the important fact that the particle

filter needs only the distribution around the current state estimates. The MHMH algorithm is shown to converge faster than the MH scheme and is well suited for particle filtering initializations. The effect of the state dimension on the convergence speed of the proposed MHMH algorithm is being studied. In the simulation examples, stopping conditions for the cross jumps depend on monitoring the distribution and seeing if the particles have converged around the modes. The stopping condition is extremely important for MHMH when the target distribution is multimodal. If the algorithm is allowed to run further, the highest mode naturally consumes the particles around the other modes. This issue can be handled by adaptively changing M_t and T_{jump}.

CHAPTER VI

PROPOSAL STRATEGIES FOR JOINT STATE-SPACE TRACKING WITH PARTICLE FILTERS

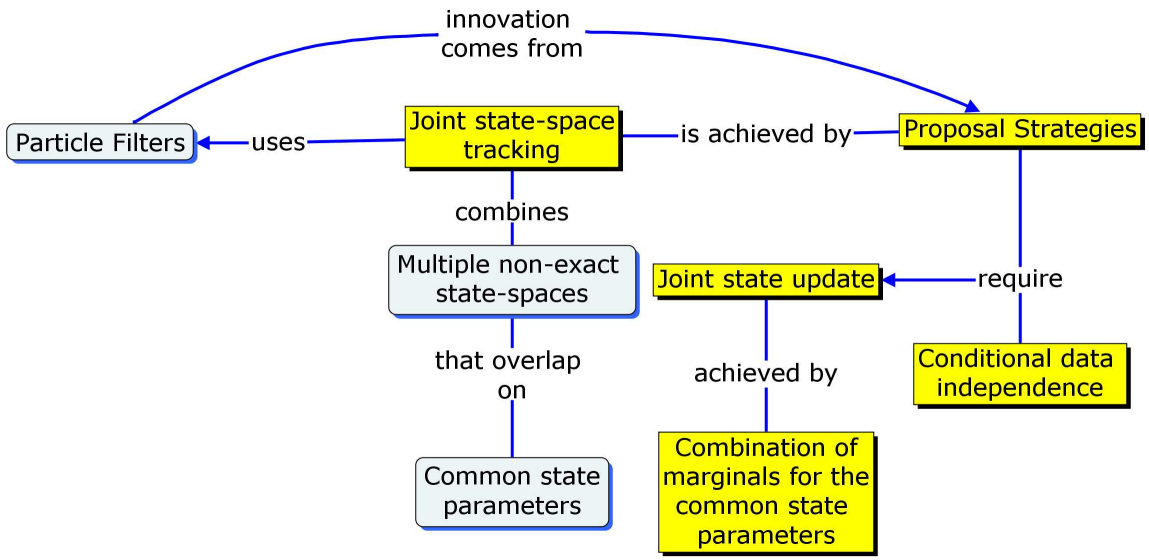


Figure 42: Chapter outline with the new contributions shown in rectangular boxes.

6.1 Introduction

State-space models are mathematical relations used for describing a system's evolution and have extensive applications in many practical problems in control theory, signal processing, and telecommunications. Since exact state-space models of real systems are extremely rare, approximate models are used. Hence, during analytical modelling of some natural phenomena, the emphasis is usually placed on choosing a minimum set of variables that completely describe a system's internal status relevant to the problem at hand. In this way, satisfactory results can still be achieved despite incomplete modelling of a system due to ignorance or lack of knowledge [14, 44].

Once a system's state-space is described in a probabilistic fashion, sequential Monté-Carlo methods, also known as *particle filters*, can be used to track the state vector as the

observations arrive in sequence. In the filter mechanics, the posteriors describing the state vector are represented by randomly distributed discrete state realizations, called particles, along with some weights. A *proposal function* determines the internal distribution of the particles and directly affects the efficiency of the filter. Given the random particle support, the particle filter can estimate any statistics of the posterior by proper weighting, and the estimation accuracy can be improved up to theoretic bounds by increasing the number of particles [25, 48].

Recently, there has been much interest in combining multiple tracking algorithms described by different state-spaces with overlapping state parameters. The motivation for joint estimation is basically two-fold: (i) to improve the performance of the estimates by merging different information streams and (ii) to maintain adequate robustness of the estimates in the face of unexpected model variations or noisy data. For joint state-space estimation problems, the particle filter is a natural choice because it propagates the probability density function (pdf) of the state vectors. Hence, it allows for heuristic combination methods, which may be problem specific [23], or a general probabilistic framework for combining information.

In this chapter, a general framework is described for tracking a single joint state vector by merging two overlapping state-space models using the particle filter. It should be stressed that combining two particle filters for different state spaces is different from formulating one filter that will track them jointly [47]. A proposal strategy is described that carefully combines the proposal strategies optimal for the individual state-spaces such that the random support of the particle filter is concentrated where the final posterior of the joint state-space lies. The resulting filter can have better estimation accuracy with the same number of particles as the individual filters.

The joint proposal strategy assumes that the state-space formulations are not exact. If there is an exact relation governing the system parameters, then a single consistent state space can be formulated easily from the individual state-spaces. Loosely speaking, each state space locally explains the underlying phenomenon and the different independent observations can be used to assist the estimation in the individual state-spaces. For example,

an acoustic tracker with a constant velocity motion model can assist a visual tracker using a random-walk model through an occlusion scenario if the acoustic propagation path from the target is not blocked.

The chapter is organized as follows. Section 6.2 sets up the state spaces and describes the assumptions in mathematical terms. Section 6.3 provides the derivation of the joint proposal strategy for particle filters. Computer simulations are given in Section 6.4.

6.2 State-Space Assumptions

Two state-spaces $\mathcal{S}_1$ and $\mathcal{S}_2$ described below are used to demonstrate the framework. The state update and observation functions of $\mathcal{S}_1$ and $\mathcal{S}_2$ are assumed to be time-invariant; however, the results can be generalized to time-varying systems including nuisance parameters. It is also assumed that the state dimensions are constant even if the system is time-varying. Define

$$\begin{aligned} \mathcal{S}_{\mathrm{i}}: \qquad x_{i,t} &= \begin{bmatrix} \chi_t \\ \psi_{i,t} \end{bmatrix} \sim q_i(x_{i,t}|x_{i,t-1}) \\ y_{i,t} &\sim f_i(y_{i,t}|x_{i,t}), \end{aligned} \tag{141}$$

where the observed data in each space is represented by $\{y_{i,t}, i = 1, 2\}$ and the overlapping, possibly multi-dimensional, state parameters are represented by χ_t. The state transition density functions $q_i(\cdot|-)$ are assumed known or, in the general time-varying case, they can be determined dynamically. The observations are explained through the density functions $f_i(\cdot|-)$. The observation sets y_i are modelled as statistically independent given the state through conditionally independent observation densities, e.g., one of them might be the acoustic observations and the other one video. This assumption is justified in many cases but may be hard to verify mathematically for a specific problem at hand [39, 47].

To track the joint state vector $x_t = [\chi_t, \psi_{1,t}, \psi_{2,t}]$ with a particle filter, the following target posterior should be determined:

$$\begin{aligned} p(x_t|x_{t-1}, y_{1,t}, y_{2,t}) &\propto p(y_{1,t}, y_{2,t}|x_t)p(x_t|x_{t-1}) \\ &= \pi_t(y_{1,t}, y_{2,t})\pi_{t-1}(x_t), \end{aligned} \tag{142}$$

where $\pi_s(\cdot) = p(\cdot|x_s)$. In (142), the Markovian property is implicitly assumed on the state-spaces. That is, given the previous state and the current data observation, the current state distribution does not depend on the previous state track and the previous observations.

Equation (142) allows the target posterior to be calculated up to a proportionality constant, where the proportionality is independent of the current state x_t. The first pdf on the right hand side of (142) is called the joint-data likelihood and can be simplified using the conditional independence assumption on the observations:

$$\pi_t(y_{1,t}, y_{2,t}) = f_1(y_{1,t}|x_{1,t}) f_2(y_{2,t}|x_{2,t}). \tag{143}$$

The last pdf in (142), corresponding to a joint state update, requires a little bit more finesse. State-spaces $\mathcal{S}_1$ and $\mathcal{S}_2$ may have different updates for the common parameter set since they are not exact. This poses a challenge in terms of formulating the common state update for x_t. Instead of assuming a given analytical form for the joint state update as in [47], we combine the individual state update marginal pdfs for the common state parameter by, in effect, convolving the models as follows:

$$\pi_{t-1}(\chi_t) = c p_1(\chi_t)^{o_1} p_2(\chi_t)^{o_2} r(\chi_t)^{o_3}, \tag{144}$$

where $c \geq 1$ is a constant, $p_i(\chi_t) \triangleq p(\chi_t|x_{i,t-1})$ is the marginal density, the probabilities o_i for $i = 1, 2$ ($\sum_i o_i = 1$) define an ownership of the underlying phenomenon by the state models, and $r(\chi_t)$ is a (uniform/reference) prior in the natural space of parameter χ_t [11] to account for unexplained observations by the state models. If we define D as the Kullback-Leibler distance, then

$$D(\alpha(\chi_t)||\pi_{t-1}(\chi_t)) = -\log c + \sum_i o_i D(\alpha(\chi_t)||p_i(\chi_t)) \tag{145}$$

where α is the unknown true χ_t distribution. Hence, $D(\alpha||\pi_{t-1}) \leq \max_i\{D(\alpha||p_i)\}$. This implies that (144) minimizes the worst case divergence from the true distribution [3]. Hence, this proves that the one of the trackers does assist the other in this framework.

The ownership probabilities, o_i, can be determined using an error criteria. For example, one way is to monitor how well each each partition $x_{i,t}$ in x_t explains the information streams $y_{i,t}$ through their state-observation equation pair defined by $\mathcal{S}_i$, (141). Then, the

respective likelihood functions can be aggregated with an exponential envelope to solve for the o_i's recursively. In this case, the target posterior will be dynamically shifting towards the better self-consistent model while still taking into account the information coming from the other, possibly incomplete, model, which might be unable temporarily to explain the temporary data stream.

If one believes that both models explain the underlying process equally likely regardless of their self-consistency, one can set $o_1 = o_2 = 1/2$ to have the marginal distribution of χ_t resemble the product of the marginal distributions imposed by both state spaces. The proposal strategy in this chapter is derived with this assumption on the ownership probabilities, because, interestingly, it is possible to show that assuming equal ownership probabilities along with (144) leads to the following conditional independence relation on the state spaces:

$$\pi_{t-1}(x_{1,t})\pi_{t-1}(x_{2,t}) = q_1(x_{1,t}|x_{1,t-1})q_2(x_{2,t}|x_{2,t-1}). \tag{146}$$

Equation (146) decouples the partition $x_{i,t}$ distributions in the joint update for x_t and finally results in the following update equation:

$$\begin{aligned} \pi_{t-1}(x_t) &= \pi_{t-1}(\psi_{1,t}, \psi_{2,t}|\chi_t)\pi_{t-1}(\chi_t) \\ &= \pi_{t-1}(\psi_{1,t}|\chi_t)\pi_{t-1}(\psi_{2,t}|\chi_t)\pi_{t-1}(\chi_t) \\ &= \frac{\pi_{t-1}(x_{1,t})\pi_{t-1}(x_{2,t})}{\pi_{t-1}(\chi_t)} \\ \Rightarrow \pi_{t-1}(x_t) &= \frac{q_1(x_{1,t}|x_{1,t-1})q_2(x_{2,t}|x_{2,t})}{\pi_{t-1}(\chi_t)}, \end{aligned} \tag{147}$$

$$\text{where} \quad \pi_{t-1}(\chi_t) \propto \left[\iint q_1(x_{1,t}|x_{1,t-1})d\psi_{1,t}q_2(x_{2,t}|x_{2,t})d\psi_{2,t}\right]^{1/2}. \tag{148}$$

6.3 Proposal Strategy

A proposal function, denoted as $g(x_t|x_{t-1}, y_t)$, determines the random support for the particle candidates to be weighted by the particle filter. Two very popular choices are (i) the state update $g \propto q_i(x_t|x_{t-1})$ and (ii) the full posterior $g \propto f_i(y_t|x_t)q_i(x_t|x_{t-1})$. The first one is attractive because it is analytically tractable. The second one is better in the sense that it incorporates the latest data while proposing particles, and it results in less variance

in the importance weights of the particle filter since, in effect, it directly samples the posterior [25, 48]. Moreover, it can be analytically approximated for faster particle generation by using local linearization techniques (see [25]), where the full posterior is approximated by a Gaussian. An example proposal function obtained by local linearization of the posterior is given by

$$g(x_t|x_{t-1}, y_t) \approx \mathcal{N}(\mu(z) + z, \Sigma(z)), \tag{149}$$

$$\text{where} \quad \Sigma(z) = -\left[l''_f(z) + l''_q(z)\right]^{-1}, \quad \mu(z) = \Sigma(z)\left[l'_f(z) + l'_q(z)\right],$$

and where l' and l'' are the gradients and the Hessians of the respective log-likelihood functions. Also, z is judiciously chosen as the mode of the posterior, which can be calculated. Hence, by either way of proposing particles, one can assume that an analytical relation for g_i, defining the support of the actual posterior for each state space, can be obtained.

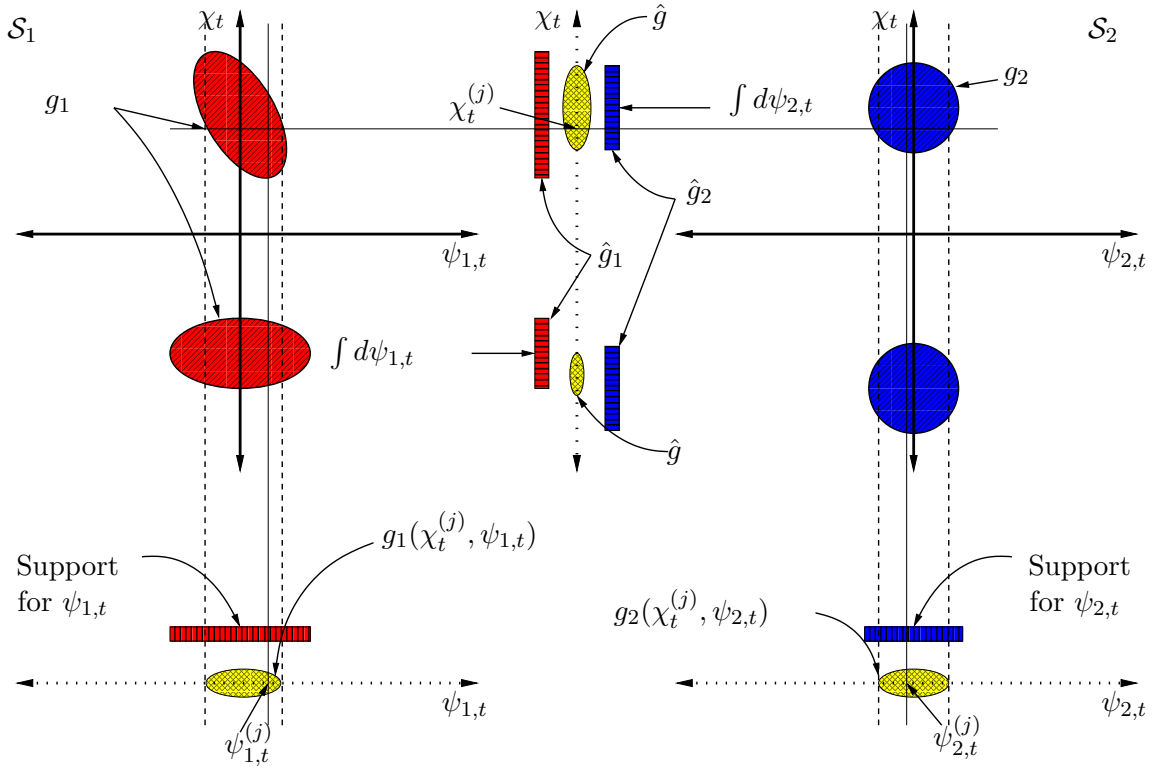


Figure 43: The supports, g_i's, for the posterior distribution in each state space, $\mathcal{S}_i$, are shown on the axes χ_t vs. $\psi_{i,t}$. Particles for the joint state are generated by first generating χ_t's from the combined supports of the marginal distributions of χ_t. $\psi_{i,t}$'s are then sampled from g_i's as constrained by the given χ_t realization.

Figure 43 describes the proposal strategy used for the joint state space. It is assumed

that each state space has a proposal strategy described by the analytical functions $\{g_i, i = 1, 2\}$ defined over the whole state-spaces. Then, the proposal functions of each state g_i are used to propose particles for the joint space by carefully combining the supports of the individual posteriors. First, marginalize out the parameters $\psi_{i,t}$:

$$\hat{g}_i(\chi_t|x_{i,t-1}, y_{i,t}) = \int g_i(x_{i,t}|x_{i,t-1}, y_{i,t})d\psi_{i,t}. \tag{150}$$

The functions, $\hat{g}_i$, describe the random support for the common state parameter χ_t and can be combined in the same way as the joint state update (144). Hence, the following function

$$\hat{g}(\chi_t|x_{t-1}, y_{1,t}, y_{2,t}) \propto [\hat{g}_1(\chi_t|x_{1,t-1}, y_{1,t})\hat{g}_2(\chi_t|x_{2,t-1}, y_{2,t})]^{1/2} \tag{151}$$

can be used to generate the candidates $\chi_t^{(j)}$ for the overlapping state parameters. Then using $\chi_t^{(j)}$, one can generate $\psi_{i,t}^{(j)}$ from $g_i(\chi_t^{(j)}, \psi_{i,t}|x_{i,t-1}, y_{i,t})$ and form $x_t^{(j)} = [\chi_t^{(j)}, \psi_t^{(j)}, \varphi_t^{(j)}]$.

In general, Monté-Carlo simulation methods can be used to simulate the marginal integrals in this section [66]. Here, we show how to calculate the marginal integrals of the state models. Simulation of the other integrals are quite similar. Given $\chi_t^{(j)}$, draw M samples using $\psi_{i,t}^{(m)} \sim g_i(\chi_t^{(j)}, \psi_{i,t}|x_{i,t-1}, y_{i,t})$[1]. Then,

$$\int q_1(\chi_t^{(j)}, \psi_{i,t}|x_{1,t-1})d\psi_{i,t} \approx \frac{1}{M}\sum_{m=1}^{M} \frac{q_1(\chi_t^{(j)}, \psi_{i,t}^{(m)}|x_{1,t-1})}{g_1(\chi_t^{(j)}, \psi_{i,t}^{(m)}|x_{1,t-1}, y_{1,t})}. \tag{152}$$

Pseudo-code for the joint strategy is given in Table 10. Finally, the importance weights for the particles generated by the joint strategy described in this section can be calculated as follows:

$$w^{(j)} \propto \frac{p(x_t^{(j)}|x_{t-1}, y_{1,t}, y_{2,t})\hat{g}(\chi_t^{(j)}|x_{t-1}, y_{1,t}, y_{2,t})}{g_1(\chi_t^{(j)}, \psi_{1,t}^{(j)}|x_{1,t-1}, y_{1,t})g_2(\chi_t^{(j)}, \psi_{2,t}^{(j)}|x_{2,t-1}, y_{2,t})}. \tag{153}$$

6.4 Examples

The objective of this section is to demonstrate the proposal strategy with an analytical example to emphasize the following: (i) the joint tracker (called $\mathcal{J}$) has less RMS error in tracking when compared to the trackers (also called $\mathcal{S}_i$) formulated using the individual

[1]It is actually not necessary to draw the samples directly from $g_i(\chi_t^{(j)}, \psi_{i,t}|-)$. An easier distribution function approximating only q_i can be used for simulating the marginalization integral (152).

Table 10: Pseudo Code for Joint Proposal Strategy

---

i. Given the state update q_i and observation relations f_i for the individual state spaces $\{\mathcal{S}_i, i = 1, 2\}$, determine analytical relations for the proposal functions g_i's. For the individual proposal functions g_i, it is important to approximate the true posterior as close as possible because these approximations are used to define the random support for the final joint posterior. For this purpose, Gaussian approximation of the posterior (149) or linearization of the state equations can be used [25].

ii. Determine the support for the common state parameter χ_t using (151). The expression for $\hat{g}$ may have to be approximated or simulated to generate candidates $\chi_t^{(j)}$, $j = 1, 2, \ldots, N$ where N is the number of particles.

iii. Given $\chi_t^{(j)}$,

- calculate the marginal integrals by using (152),
- generate $\psi_{i,t}^{(j)} \sim g_i(\chi_t^{(j)}, \psi_{i,t} | x_{i,t-1}, y_{i,t})$,
- form $x_t^{(j)} = [\chi_t^{(j)}, \psi_{1,t}^{(j)}, \psi_{2,t}^{(j)}]$, and
- calculate the importance weights, $w^{(j)}$'s, using (153).

---

state spaces, and (ii) the joint tracker manifest robustness in the common parameter estimates even if one of the models start to diverge. For the analytic example, the ownership parameters for the state spaces are fixed to $o_i = 1/2, i = 1, 2$ and the prior component $r(\cdot)$ in (144) is not used.

Consider the following state-space descriptions

$$\mathcal{S}: \quad \begin{bmatrix} \chi_t \\ \psi_t \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \chi_{t-1} + \cos(\omega t)\psi_t \\ \psi_{t-1} \end{bmatrix}, \begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix} \right) \tag{154}$$

$$\begin{aligned} \mathcal{S}_1: \quad & \begin{bmatrix} \chi_t \\ \psi_t \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \chi_{t-1} + \psi_t \\ \psi_{t-1} \end{bmatrix}, \begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix} \right) \\ & y_t \sim \mathcal{N}\left(\begin{bmatrix} \chi_t \\ \psi_t \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \right) \end{aligned} \tag{155}$$

$$\begin{aligned} \mathcal{S}_2: \quad & \chi_t \sim \mathcal{N}\left(\chi_{t-1}, \lambda_3^2\right) \\ & \theta_t \sim \mathcal{N}\left(\tan^{-1}\left(\frac{\chi_t}{W}\right), \sigma_3^2\right) \end{aligned} \tag{156}$$

where $\mathcal{S}$ is the true state model with a deterministic parameter ω. $\mathcal{S}_1$ and $\mathcal{S}_2$ are different formulations trying to explain the phenomenon generated by $\mathcal{S}$ and have independent observations of $\mathcal{S}$. In this case, the joint state space is included in the state space of $\mathcal{S}_1$. Hence, the objective is to do a better job in tracking χ_t given the aggregate observations.

Since the data-likelihood and the state-likelihood functions of $\mathcal{S}_1$ are linear Gaussian, the full posterior can be analytically determined. In this case, the proposal function g_1 will be the actual posterior for $\mathcal{S}_1$:

$$g_1(\chi_t, \psi_t | \chi_{t-1}, \psi_{t-1}, y_t) \sim \mathcal{N}(\mu_1, \Sigma_1), \tag{157}$$

where

$$\Sigma_1 = \begin{bmatrix} \frac{1}{\sigma_1^2} + \frac{1}{\lambda_1^2} & -\frac{1}{\lambda_1^2} \\ -\frac{1}{\lambda_1^2} & \frac{1}{\sigma_2^2} + \frac{1}{\lambda_1^2} + \frac{1}{\lambda_2^2} \end{bmatrix}^{-1}, \tag{158}$$

$$\mu_1 = \Sigma_1 \left\{ \begin{bmatrix} y_{t,1}/\sigma_1^2 \\ y_{t,2}/\sigma_2^2 \end{bmatrix} + \begin{bmatrix} \frac{1}{\lambda_1^2} & -\frac{1}{\lambda_1^2} \\ -\frac{1}{\lambda_1^2} & \frac{1}{\lambda_1^2} + \frac{1}{\lambda_2^2} \end{bmatrix} \begin{bmatrix} \chi_{t-1} + \psi_{t-1} \\ \psi_{t-1} \end{bmatrix} \right\}. \tag{159}$$

The marginal distribution of χ_t from $\mathcal{S}_1$ is calculated using its state update. The resulting marginal distribution is

$$p(\chi_t | \chi_{t-1}, \psi_{t-1}) \sim \mathcal{N}\left(\chi_{t-1} + \psi_{t-1}, \lambda_1^2 + \lambda_2^2\right). \tag{160}$$

Moreover, (150), and (157) lead to

$$\hat{g}_1(\chi_t | \chi_{t-1}, \psi_{t-1}, y_t) \sim \mathcal{N}(\mu_1(1), \Sigma_1(1,1)). \tag{161}$$

For $\mathcal{S}_2$, the data-likelihood is given by

$$L_\theta \doteq -\frac{1}{2\sigma_3^2} \left[\theta_t - \tan^{-1}\left(\frac{\chi_t}{W}\right)\right]^2. \tag{162}$$

The proposal function for $\mathcal{S}_2$, which approximates the full posterior is obtained by local linearization of the model as follows:

$$g_2(\chi_t | \chi_{t-1}, \theta_t) = \hat{g}_2(\chi_t | \chi_{t-1}, \theta_t) \sim \mathcal{N}(\mu_2, \Sigma_2) \tag{163}$$

$$\Sigma_2 = \left[\frac{1}{\sigma_3^2} + \left(\frac{W}{W^2 + \chi_t^2}\right)^2 \frac{1}{\lambda_3^2}\right]^{-1}, \tag{164}$$

Table 11: RMS errors for χ_t and ψ_t. Large numbers in the table are caused by tracking divergence. The underlying model $\mathcal{S}$ is closer to $\mathcal{S}_1$ when ω is small, whereas it is closer to $\mathcal{S}_2$ when ω is large.

ω	$\mathcal{J}(\chi_t)$	$\mathcal{S}_1(\chi_t)$	$\mathcal{S}_2(\chi_t)$	$\mathcal{J}(\psi_t)$	$\mathcal{S}_1(\psi_t)$
1/500	44.38	80.24	6669.5	21.43	24.83
1/50	42.99	101.05	1511.7	26.55	27.68
1/5	235.7	1134.7	48.96	222.59	134.21

$$\mu_2 = \Sigma_2 \left[\frac{\chi_{t-1}}{\sigma_3^2} + \frac{W}{W^2+\chi_t^2} \frac{1}{\lambda_3^2} \left(y_{t,2} - \tan^{-1}\left(\frac{\chi_{t-1}}{W}\right) + \frac{W\chi_{t-1}}{W^2+\chi_t^2} \right) \right]. \tag{165}$$

Hence, to generate χ_t, substitute (161) and (163) into (151) to obtain

$$\hat{g}(\chi_t|\chi_{t-1}, \psi_{t-1}, y_t, \theta_t) \sim \mathcal{N}(\mu, \Sigma) \tag{166}$$

$$\Sigma = 2\left(\frac{1}{\Sigma_1(1,1)} + \frac{1}{\Sigma_2}\right)^{-1}, \tag{167}$$

$$\mu = \left(\frac{1}{\Sigma_1(1,1)} + \frac{1}{\Sigma_2}\right)^{-1} \left(\frac{\mu_1(1)}{\Sigma_1(1,1)} + \frac{\mu_2}{\Sigma_2}\right). \tag{168}$$

Given $\chi_t^{(j)}$, (157) can be used to generate ψ_t as follows

$$g_1(\chi_t^{(j)}, \psi_t|\chi_{t-1}, \theta_t) \sim \mathcal{N}\left(\mu_1(2) + \frac{\rho_1\left(\chi_t^{(j)} - \mu_1(1)\right)}{\Sigma_1(1,1)}, \Sigma_1(2,2)\right), \tag{169}$$

where ρ_1 is the correlation coefficient of Σ_2.

Three particle filters ($\mathcal{J}$,$\mathcal{S}_1$,$\mathcal{S}_2$) are implemented each with 100 particles. $\mathcal{J}$ uses the joint proposal strategy whereas $\mathcal{S}_1$ and $\mathcal{S}_2$ use (157) and (166) as proposal functions. Each filter also has a resampling stage where particles are resampled with replacement according to their probabilities. Simulation parameters are $\lambda_1^2 = 1$, $\lambda_2^2 = .5$, $\sigma_1^2 = 2$, $\sigma_2^2 = 1$, $\lambda_3^2 = 9$, $\sigma_3^2 = (1^\circ)^2$, $W = 50$, and various ω's. The simulation is done for 50 iterations total. Table 11 demonstrates the RMS error averages of the Monté-Carlo simulation where model $\mathcal{S}$ is simulated 100 times with different ω parameters.

It should be noted that when neither $\mathcal{S}_1$ nor $\mathcal{S}_2$ diverges, their performances are comparable. In those cases, the joint filter has less than one-fourth of their combined RMS error because the particles are not wasted in redundant areas. Table 11 also shows that the joint filter demonstrates improved performance as well as robustness. Even though one of the models diverges, the joint filter still does a good job, because it is, in effect, assisted by the

presence of the other built-in model, which can still track the phenomenon. The filter's performance may be further improved by implementing the adaptive ownership probabilities. Lastly, the joint tracker's robustness vs. time is shown in Fig. 44.

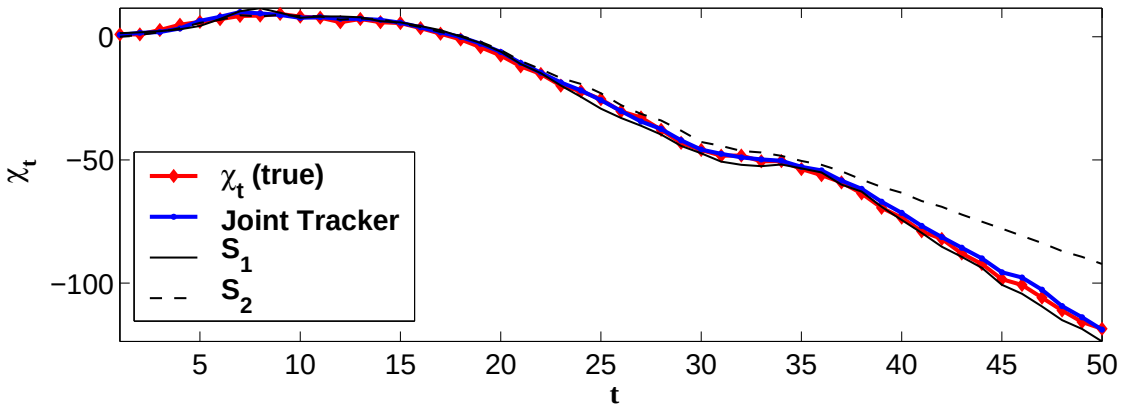


Figure 44: Example tracking realization with $\omega = 1/50$: Even though the particle filter designed using $\mathcal{S}_2$ is unable to track after $t = 35$s, the joint tracker still does a good job since it uses the information from coming both state-space models.

6.5 Conclusions

In this chapter, a general proposal strategy is demonstrated for joint state-space tracking with particle filters. The framework is quite general and it allows different state-spaces to assist each other in tracking a common parameter. The framework is demonstrated using an analytical example that mimics a joint acoustic image tracker.

CHAPTER VII

CONCLUSIONS

The performance of any tracking/detection algorithm will be inherently limited by its ability to accurately model reality. In Chapter 2, we introduced the independent partition particle filter using the narrow-band observation model, and then we have also developed a new motion model that incorporates an acceleration component along the heading direction of the target. We have shown that the target motion parameters can be considered part of a more general feature set for target tracking, e.g., target frequencies, which may be unrelated to the target motion, can be used to improve the tracking performance. To include the frequency variable, a new array steering vector was defined for the direction-of-arrival (DOA) estimation problems.

Although analytically very convenient, the narrow-band observation model is not suitable for tracking targets with harmonic time-frequency signatures. This is because the harmonics of a single target may confuse the tracker's ability to correctly count the actual number targets present in the field. To decrease the dependence on the observation model we have reworked the acoustic tracker using a flexible observation model based on an image tracking approach. This new acoustic tracker was presented in Chapter 3. The new observation model not only enables the particle filter to cope with more realistic scenarios, but also decreases its computational complexity by pre-processing the acoustic data to produce sufficient statistics for the tracking. The tracker is also formulated such that it is also robust against high observation noise that is commonly encountered in field data.

In Chapter 4, we first attacked the acoustic node calibration problem using the powerful maximum-likelihood (ML) method directly on the received acoustic data. We developed a Newton search method for its efficient solution and determined the inherent performance limits of the problem. We then showed that the same calibration performance can be achieved with less computation by building the calibration algorithms around the sufficient

statistics, an idea that we already used in Chapter 3. Interestingly, by doing so, we have also determined a biased estimator that out performs the ML method for calibration in minimum mean-squared error sense. The Metropolis-Hastings (MH) algorithm that we developed for the calibration problem was also tested with field data and its performance was compared to the existing solutions (i.e., GPS receiver on the nodes).

We then focused on how to accelerate the MH algorithm, which is a very general sampling algorithm. The modification was done so that the resulting sampling distribution has a certain shape: particles distributed around the modes of the posterior. Through simulations, we showed that the accelerated algorithm is an order of magnitude faster than the original algorithm. This accelerated Mode Hungry MH scheme can be used to initialize the acoustic particle filter trackers discussed in Chapters 2 and 3. It can also increase the speed of the MH scheme devised for the node calibration problem.

Finally, we have developed a general framework for the joint state-space tracking problem in Chapter 6. Challenges arise when one considers how to merge two or more trackers in a common framework to result in a single estimate. We considered merging the state vectors of the two trackers in one state vector. Then, the tracker merging problem was addressed in two parts. The first part, the consistent state-update problem, was solved by merging the marginal pdfs of the overlapping state space parameters. The second part, the proposal function of the particle filter, was developed by carefully placing the random support of the joint filter in the region where the final posterior is likely to lie. Computer simulations demonstrate improved performance and robustness of the joint state-space tracker when using the new particle proposal strategy.

REFERENCES

[1] Ackerberg, A., "A new use of importance sampling to reduce computational burden in simulation estimation." under revision at *Review of Economic Studies*, Available http://www.econ.ucla.edu/ackerber/smoonber2.pdf.

[2] Aidala, V. and Hammel, S., "Utilization of modified polar coordinates for bearings-only tracking," *IEEE Trans. on Automatic Control*, vol. AC-28, no. 3, pp. 283–294, March 1983.

[3] Ali, S. and Silvey, S., "A general class of coefficients of divergence of one distribution from another," *Journal of the Royal Statistical Society*, vol. 28, pp. 131–142, 1966.

[4] Allen, R. and Blackman, S., "Implementation of an angle-only tracking filter," in *SPIE Proc.*, vol. 1481, pp. 292–303, 1991.

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

[6] Bar-Shalom, Y., "Tracking methods in a multitarget environment," *IEEE Trans. Automatic Control*, vol. AC-23, pp. 618–626, Aug. 1978.

[7] Bar-Shalom, Y. and Fortmann, T., *Tracking and Data Association*. Academic-Press, 1988.

[8] Barrett, R. and Holdsworth, D., "Frequency line tracking using hidden Markov models," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 38, no. 4, pp. 586–598, April 1990.

[9] Barrett, R. and Holdsworth, D., "A frequency tracking using hidden Markov models with amplitude and phase information," *IEEE Trans. on Signal Processing*, vol. 41, no. 10, pp. 2965–2976, October 1993.

[10] Berger, J. and Bernardo, J., "On the development of reference priors," *Bayesian Statistics*, vol. 4, pp. 35–60, 1992.

[11] Bernardo, J., "Reference posterior distributions for Bayesian inference," *J.R. Statist. Soc. B*, vol. 41, pp. 113–147, 1979.

[12] Bernardo, J. and Ramon, J., "An introduction to Bayesian reference analysis: Inference on the ratio of multinomial parameters," *Bayesian Statistics*, vol. 47, pp. 101–135, 1998.

[13] Box, G. and Tiao, G., "On sequential simulation-based methods for Bayesian filtering," Tech. Rep. CUED/F-INFENG/TR.310, Department of Engineering, University of Cambridge, 2001.

[14] Brogan, W., *Modern Control Theory*. Prentice Hall, 1991.

[15] Brooks, S. and Gelman, A., "General methods for monitoring convergence of iterative simulations," *Journal of Computational and Graphical Statistics*, vol. 7, pp. 434–455, 1998.

[16] Capon, J., Greenfield, R., and Kolker, R., "Multidimensional maximum-likelihood processing of a large aperture seismic array," *Proc. IEEE*, vol. 55, pp. 192–211, Feb. 1967.

[17] Castella, G. and Robert, C., "Rao-Blackwellization of sampling schemes," *Biometrika*, vol. 83, pp. 81–94, 1996.

[18] Cevher, V. and McClellan, J. H., "General direction-of-arrival tracking with acoustic nodes." To appear in *IEEE Trans. on Signal Processing*.

[19] Cevher, V. and McClellan, J. H., "Fast initialization of particle filters using a modified Metropolis-Hastings algorithm: Mode-Hungry approach," in *ICASSP 2004*, (Montreal, CA), 17–22 May 2004.

[20] Cevher, V. and McClellan, J. H., "Sensor array calibration via tracking with the extended Kalman filter," in *Proc. of the the Fifth Ann. Fed. Lab. Symp. on Adv. Sensors*, (College Park, MD), pp. 51–56, 20-22 March 2001.

[21] Cevher, V. and McClellan, J., "2-D sensor position perturbation analysis: Equivalence to AWGN on array outputs," in *Second IEEE Sensor Array and Multichannel Signal Processing Workshop*, (Washington DC.), 2-4 August 2002.

[22] Chang, K. and Bar-Shalom, Y., "Joint probabilistic data association for multitarget tracking with possibly unresolved measurements," *IEEE Trans. Automatic Control*, vol. AC-29, pp. 585–594, July 1978.

[23] Chellappa, R., Qian, G., and Zheng, Q., "Vehicle detection and tracking using acoustic and video sensors," in *ICASSP 2004*, (Montreal, CA), 17-21 May 2004.

[24] Chib, S. and Greenberg, E., "Understanding the Metropolis-Hastings algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995.

[25] Doucet, A., Freitas, N., and Gordon, N., eds., *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[26] et. al., P. C., "Discussion on the meeting on the Gibbs sampler and other Markov chain Monte Carlo methods," *Journal of Royal Statistical Society*, vol. 55, no. 1, pp. 53–102, 1993.

[27] Farina, A., "Target tracking with bearings-only measurements," *Elsevier Signal Processing*, vol. 78, pp. 61–78, 1999.

[28] Fistas, N. and Manikas, A., "A new general global array calibration method," in *ICASSP*, vol. 4, pp. 73–76, May 1994.

[29] Gelfand, A. and Smith, A., "Sampling-based approaches to calculating marginal densities," *Journal of the American Statistical Association*, vol. 85, pp. 398–409, 1990.

[30] Gelman, A., "Iterative and non-iterative simulation algorithms," in *Proceedings of Computing Science and Statistics*, pp. 433–438, 1992.

[31] Gelman, A., Roberts, G., and Gilks, W., "Efficient Metropolis jumping rules," *Bayesian Statistics*, vol. 5, 1996.

[32] Gelman, A. and Rubin, D., "Inference from iterative simulation using multiple using multiple sequences," *Statistical Science*, vol. 7, pp. 457–511, 1992.

[33] Gershman, A. and Amin, M., "Wideband direction-of-arrival estimation of multiple chirp signals using spatial time-frequency distributions," *IEEE Signal Processing Letters*, vol. 7, no. 6, pp. 152–155, June 2000.

[34] Giovannelli, J., Idier, J., Boubertakh, R., and Herment, A., "Unsupervised frequency tracking beyond the Nyquist frequency using Markov chains," *IEEE Trans. on Signal Processing*, vol. 50, no. 12, pp. 2905–2914, December 2002.

[35] Goodman, N., "Statistical analysis based on a certain multivariate complex Gaussian distribution (an introduction)," *Annals of Mathematical Statistics*, pp. 152–177, March 1963.

[36] Goodridge, S. and Kay, M., "Multimedia sensor fusion for intelligent camera control," in *Proceedings of the 1996 IEEE International Conference on Multisensor Fustion and Integration for Intelligent Systems*, pp. 655–662, 1996.

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

[38] Hoelzer, H., Johnson, G., and Cohen, A., "Modified polar coordinates- the key to well behaved bearings-only ranging," Tech. Rep. IBM Rep. 78-M19-0001A, IBM Shipboard and Defense Systems, Manassas, VA, August 1978.

[39] II, M. L., Chong, C., Kadar, I., Alford, M., Vannicola, V., and Thomopoulos, S., "Distributed fusion architectures and algorithms for target tracking," *Proceedings of the IEEE*, vol. 85, pp. 95–107, Jan. 1997.

[40] Isard, M. and Blake, A., *Active Contours*. Springer, 2000.

[41] Isard, M. and MacCormick, J., "BraMBLe: A Bayesian multiple-blob tracker," in 8^{th} *International Conference on Computer Vision*, 2001.

[42] Johnson, D. and Dudgeon, D., *Array Signal Processing: Concepts and Techniques*. Prentice Hall, 1993.

[43] Johnson, G., Hoelzer, H., Cohen, A., and Harold, E., "Improved coordinates for target tracking from time delay information," in *Proc. of the Time Delay Estimation and Applications Conf.*, vol. 2, (Naval Postgraduate School, Monterey, CA), pp. M1–M32, May 1979.

[44] Kailath, T., Sayed, A., and Hassibi, B., *Linear Estimation*. Prentice Hall, 2000.

[45] Karlsson, R. and Gustafsson, F., "Monte Carlo data association for multiple target tracking," in *IEE Target Tracking: Algorithms and Applications*, (Netherlands), 16-17 Oct. 2001.

[46] Lanterman, A., "Schwarz, Wallace, and Rissanen: Intertwining themes in theories of model selection," *International Statistical Review*, vol. 69, pp. 185–212, 2001.

[47] Leichter, I., Lindenbaum, M., and Rivlin, E., "A probabilistic framework for combining tracking algorithms," in *CVPR 2004*, (WDC), June 27–July 2 2004.

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

[49] MacCormick, J. and Blake, A., "A probabilistic exclusion principle for tracking multiple objects," in 7^{th} *International Conference on Computer Vision*, pp. 572–578, 1999.

[50] MacCormick, J. and Isard, M., "Partitioned sampling, articulated objects, and interface-quality hand tracking," in *Proceedings of the Europen Conference on Computer Vision*, 2000.

[51] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E., "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1092, 1953.

[52] Moses, R., Krishnamurthy, D., and Patterson, R., "An auto-calibration method for unattended ground sensors," *EURASIP Journal on Applied Signal Processing*, vol. 4, pp. 348–358, 2003.

[53] Moses, R., Krishnamurthy, D., and Patterson, R., "An auto-calibration method for unattended ground sensors," in *ICASSP 2002*, vol. 3, (Orlando, FL), pp. 2941–2944, May 2002.

[54] Murphy, D., *Noisy bearings-only target motion analysis.* PhD thesis, Northeastern Univ., Boston, MA, 1970.

[55] Nardone, S. and Aidala, V., "Observability criteria for bearings-only target motion analysis," *IEEE Trans. on Aerospace and Electronic Systems*, vol. AES-18, pp. 162–166, July 1982.

[56] Ng, B. and Nehorai, A., "Active array sensor localization," in *ICASSP 1993*, vol. 4, pp. 21–24, 27-30 April 1993.

[57] Ng, B. and See, C., "Sensor array calibration using a maximum-likelihood approach," *IEEE Trans. on Antennas and Propagation*, vol. 44, pp. 827–835, June 1996.

[58] Ng, L. and Bar-Shalom, Y., "Multisensor multitarget time delay vector estimation," *IEEE Trans. ASSP*, vol. ASSP-34, pp. 669–677, Aug. 1986.

[59] Orton, M. and Fitzgerald, W., "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," Tech. Rep. CUED/F-INFENG/TR.403, Department of Engineering, University of Cambridge, 2001.

[60] Orton, M. and Fitzgerald, W., "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Trans. on Signal Processing*, vol. 50, no. 2, pp. 216–223, February 2002.

[61] Papoulis, A. and Pillai, S., *Probability, random variables and stochastic processes.* McGraw Hill, 2002.

[62] Pitt, M. and Shephard, N., "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, pp. 590–599, 1999.

[63] Poor, H., *An Introduction to Signal Detection and Estimation*. Springer-Verlag, 1994.

[64] Rabiner, L., "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.

[65] Rago, C., Willett, P., and Streit, R., "A comparison of the JPDAF and PMHT tracking algorithms," in *ICASSP 1995*, vol. 5, pp. 3571–3574, 1995.

[66] Ripley, B., *Stochastic Simulation*. John Wiley & Sons Inc., 1987.

[67] Rissanen, J., "Fisher information and stochastic complexity," *IEEE Trans. on Information Theory*, vol. 42, no. 1, pp. 40–47, January 1996.

[68] Smith, A. and Roberts, G., "Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society*, vol. 55, pp. 3–24, 1993.

[69] Sontag, E., "On the observability of autonomous discrete time nonlinear system," *Int. J. Contr.*, vol. 36, pp. 867–874, 1982.

[70] S.P.Brooks and Gelman, A., "General methods for monitoring convergence of iterative simulations," *Journal of Computational and Graphical Statistics*, vol. 7, pp. 434–455, 1998.

[71] Stoica, P. and Nehorai, A., "Music, maximum likelihood, and Cramér-Rao bound," *IEEE Trans. on ASSP*, vol. 37, no. 5, pp. 720–741, May 1989.

[72] Sword, C., Simaan, M., and Kamen, E., "Multiple target angle tracking using sensor array output," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, pp. 367–372, Mar. 1990.

[73] Takahashi, K. and Yamasaki, H., "Audio-visual sensor fusion system for intelligent sound sensing," in *Proceedings of the 1994 IEEE International Conference on Multisensor Fustion and Integration for Intelligent Systems*, pp. 493–500, Oct. 1994.

[74] Tanner, M. and Wong, W., "The calculation of posterior distributions by data augmentation," *Journal of the American Statistical Association*, vol. 82, pp. 528–549, 1987.

[75] Tierney, L., "Markov chains for exploring posterior distributions," *The Annals of Statistics*, vol. 22, no. 4, pp. 1701–1728, 1994.

[76] Wang, H. and Kaveh, M., "On the performance of signal-subspace processing-part I: Narrow-band systems," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, pp. 1201–1209, October 1986.

[77] Zhou, Y., Yip, P., and Leung, H., "Tracking the direction-of-arrival of multiple moving targets by passive arrays: Algorithm," *IEEE Trans. on Signal Processing*, vol. 47, no. 10, pp. 2655–2666, October 1999.

VITA

Volkan CEVHER was born in Ankara, Turkey, in August 1978. He received his B.S. Degree in Electrical Engineering from Bilkent University, Ankara, Turkey in 1999 as a valedictorian. During summer of 2003, He was employed by Schlumberger Doll Research. In Fall 2004, he was the co-recipient of the Center for Signal and Image Processing outstanding research award. He will receive his PhD Degree in Electrical Engineering from Georgia Institute of Technology in 2005. His research interests include Monte-Carlo Markov chain methods (specifically particle filters), target tracking models, adaptive filters, time-frequency distributions, fractional Fourier transform, and array signal processing. He is currently involved with the Advanced Sensors CTA sponsored by ARL.