Sequentially Structured Bayesian Solutions

Simon Maskell

Sequentially Structured Bayesian Solutions

Simon Maskell

February 21, 2004

draft date: February, 2004.

Typeset by the author with the LaTeX 2_ε Documentation System.

To Michelle

Declaration

This dissertation is submitted for the degree of Doctor of Philosophy. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text. This dissertation contains less than limits of 150 figures and 65,000 words; there are approximately 26 figures and 44,500 words.

Acknowledegements

I would like to thank my parents for their unending support. I would also like to thank my sisters and the rest of my family. I am also very grateful indeed to Michelle for putting up with me while I wrote this thesis and for all her continual encouragement.

I am also particularly gratefully for the award of an Industrial Fellowship by the Royal Commission for the Exhibition of 1851, which made this thesis possible. The projects constituting this thesis was undertaken in part under projects that were sponsored by the UK MOD Corporate Research Programme, within the Energy Guidance and Control, Communications Informaiton and Sensor Processing and Sensor and Electronic Warfare Technology Research Domains and their predecessors Technnology Groups 3, 10 and 9 respectively.

I would like to thank Professor Bill Fitzgerald of Cambridge University Engineering Department, CUED, and Dr Neil Gordon of the Defence Science and Technology Organisation, DSTO, Australia, for teaching me so well that I was able to undertake the work documented in this thesis. I would also like to thank Dr Arnaud Doucet of CUED and Dr Alan Marrs of QinetiQ for their support and guidance. Thank you also to Mr Mark Briers, Mr Rob Wright, Mr Martin Evans, Mr Richard Everitt, Dr David Salmond, Mr Kevin Weekes, Dr Andrew Webb, Dr Tim Field, Dr Colin Reed, Mr Nick Everett, Dr Malcolm Rollason, Dr Marcel Hernandez, Dr Paul Horridge, Dr Richard Glendinning, Dr Jon Salkeld, Mr Eric Knowles, Dr Steve Luttrell, Mrs Helen Newsholme, Mrs Jenny Green and Dr Arthur Williams of QinetiQ, Mr Matthew Orton, Dr Simon Godsill, Dr Jaco Vermaak and Professor Peter Rayner of CUED, Dr Christophe Andrieu of Bristol University, Dr Mahendra Mallick of Orincron, Dr Sanjeev Arulampalam of DSTO, Dr Thia Kirubarajan of MacMaster University and Dr Yaakov Bar-Shalom of Connecticut University for their invaluable support, guidance and assistance.

This version of the thesis is that corrected in response to the comments of the examiners following the viva. The contribution of the examiners, Dr Arnaud Doucet and Dr Jean-Pierre Le Cadre, in this respect is acknowledged.

Abstract

A large number of problems have sequential structure. These include but are not limited to problems that are often posed as sequential problems, such as the analysis of time series. So, this thesis begins by describing the context provided by existing sequential algorithms. Discussion then focuses on methods for developing the discrete-time dynamic models to be used in such algorithms from systems of stochastic continuous-time differential equations in such a way that the conversion itself does not impact performance. Next, the sequential problem of efficiently tracking a manoeuvring target governed by a semi-Markov model while robustly classifying the target is solved. An efficient mechanism is then proposed for conducting data association by summing over all possible assignments of measurements to targets in substantially less than exponential time. The approach exploits analogous structure in the assignment problem as is exploited when analysing time-series; sufficient statistics can be used to summarise the impact of the past assignments of some targets to measurements on the future assignments of the remaining targets to the measurements. As a final example, the analysis of images is considered as the sequential analysis of columns of data where each operation on the columns can itself be implemented using sequential algorithms on the pixels. The approach makes it viable to perform Bayesian inference on large images with no requirement for low noise conditions or need to resort to computationally expensive approaches such as MCMC. Finally some conclusions are drawn and avenues of further work proposed.

Contents

Introduction 1

The assertion made by this thesis is that there are a large number of problems which have sequential structure. By structuring solutions to these problems in such a way as to reflect and exploit this sequential structure, it is shown that a number of problems for which efficient solutions have evaded researchers can indeed be solved efficiently.

Chapter 2 aims to provide a thorough description and review of the literature on which the rest of the work builds and to do so through an appeal to intuition. While the chapter does describe the many variants of the many Kalman filter algorithms, the focus is on particle filtering. So, the bulk of the chapter is a modified version of [74] which in turn builds heavily on [2] and [79], which themselves are based on [73]. Section 2.5.3 describes a number of algorithms that fill the gap between the Kalman Filter and UKF and is joint work with Mark Briers and Robert Wright[19]. Section 2.6.4 describes a new minimum error resampling scheme and relates state-of-the-art particle filters to previous work on multiple hypothesis tracking with EKFs and is joint work with Arnaud Doucet. Section 2.7.3 discusses joint work with Alan Marrs and Yaakov Bar-Shalom[69].

Chapter 3 proposes an off-the-shelf method for converting systems of stochastic differential equations into the dynamic models used by trackers without the need for understanding the intricacies of Itô calculus and in such a way that the conversion process itself does not impact the potential efficiency of the algorithm. The approach is related to a previous approach developed by other authors that uses Itô calculus[92, 110] and it is shown that for simple systems the two approaches are the same. However, as the complexity and dimensionality increases, the proposed approach is demonstrated to be easier to apply. The work of this chapter has been submitted for publication[71]. The contributions of Tim Field in the initial stages of this work are acknowledged.

Chapter 4 shows how to track using Semi-Markov models, a generalisation of Markov models that explicitly model the state-dependent sojourn time distribution, the time for which the system remains in a given state. Markov models result in an exponentially distributed sojourn time, while semi-Markov make it possible to define the distribution explicitly. Such models can be used to describe the behaviour of manoeuvering targets and particle filtering can then facilitate tracking. Building on joint work with Alan Marrs and Yaakov Bar-Shalom[69] and with Thia Kirubarajan and Neil Gordon[43], an architecture is then proposed that enables particle filters to be both robust and efficient when conducting joint tracking and classification. It is demonstrated that this approach can be used to classify targets on the basis of their manoeuvrability. The chapter has been submitted for publication [76]. The author acknowledges the ccontributions of John Boyd and Dave Sworder during useful discussions on the subject of joint tracking and classification associated with the use of semi-Markov models and of Arnaud Doucet and Matthew

Orton during discussions of how particle filters can be used in such problems.

Chapter 5 considers multi-target tracking and presents a method that circumnavigates the combinatorial explosion often assumed to exist when summing probabilities of joint association events in a multiple target tracking context. The approach involves no approximations in the summation and while the number of joint events grows exponentially with the number of targets, the computational complexity of the approach is substantially less than exponential. Multiple target tracking algorithms that use this summation include mutual exclusion[66] in a particle filter context and the Joint Probabilistic Data Association Filter[38], a Kalman Filter based algorithm. The perceived computational expense associated with this combinatorial explosion has meant that such algorithms have been restricted to applications involving only a handful of targets. The approach presented here makes it possible to use such algorithms with a large number of targets. This approach has been published confidentially[72] and a patent filed[75].

Chapter 6 describes an approach for describing images as time series where each operation in the time series analysis is itself implemented using time series algorithms. Previous work has considered the use of time series algorithms to analyse images[59, 104], but this approach is substancially more efficient. The approach is demonstrated by generalising the Kalman Filter to two-dimensional time. This work was conducted jointly with Matthew Orton and Neil Gordon[80]. The authors of [80] acknowledge the helpful discussions with Michael Lavine and Maria De Iorio that were crucial in establishing the novelty of the approach.

Chapter 7 then draws a few overarching conclusions and recommends avenues of further work to be pursued.

Sequential Context 2

2.1 Introduction

The aim of this chapter is to introduce the analysis of sequential problems, sequential inference, in general and particle filtering in particular. The emphasis will be on tracking problems, the sequential estimation of the state of a number of targets. The intention is to compliment other introductions[27], tutorials[2] and books[10, 14] with a contribution that appeals to intuition and documents an understanding that demystifies what can appear to be a much more complex subject than it truly is. Rather than describing algorithmic recipes, the aim is to detail the make up of a generic algorithmic framework and then show how various algorithms are special cases of this framework. In so doing, the intention is that the plethora of acronyms that exist can be replaced with a single comprehensible generic description.

Sequential inference is the problem of updating some estimates about a time evolving system as measurements are received. Tracking is the task of doing this when the system is a collection of targets. The Kalman filter is the solution to the sequential inference problem when certain assumptions hold about the models for the targets' motion and the measurement process. These assumptions often hold (either exactly or approximately) in a tracking context. As a result, the use of Kalman filter based algorithms for tracking is widespread; such algorithms decompose the tracking of multiple manoeuvring targets with ambiguous data association in such a way that the subproblems are all amenable to analysis with (often approximate) Kalman filters.

Particle filtering is a new statistical technique for sequentially updating estimates about a time evolving system as measurements are received. The approach has been developed in parallel by a number of different researchers in a number of fields and so is also known as: the CONDENSATION algorithm[15], the bootstrap filter[42], interacting particle approximations[25] and survival of the fittest[56]. The approach opens the door to the analysis of time series using nonlinear non-Gaussian state-space models. While the linear Gaussian models behind the use of Kalman filters can be used to model a large variety of systems[46], nonlinear non-Gaussian models offer an even richer vocabulary with which to describe the evolution of a system and observations of this system.

Particle filtering is based on the idea that uncertainty over the value of a continuous random variable x can be represented using a probability density function, a pdf, $p(x)$. From this pdf, it is possible to deduce the estimates that are of interest; for example the mean, various quantiles and the covariance. It is rarely

possible to uniquely deduce the pdf from such estimates. So, to use nonlinear, non-Gaussian state-space models, one has to learn how to directly manipulate general pdfs. This chapter therefore describes how to use such models to analyse time-series by considering how to manipulate pdfs. The various Kalman filter based algorithms are described in this context since this probabilistic framework provides a common language with which it is straightforward to see how the different approaches compare.

Section 2.2 starts by describing the state-space model and its generic solution. The way that the Kalman Filter and Grid Based Filter then implement this generic solution are described in sections 2.3 and 2.4 respectively before section 2.5 describes how the Extended Kalman Filter can tackle nonlinear problems by approximating the models. Section 2.6 then describes particle filtering in the same context. There is then some discussion of existing tracking algorithms in section 2.7. Finally, section 2.8 draws some conclusions. Throughout much of the chapter, a simple example is used to focus discussion and aid explanation.

2.2 State-Space Model

2.2.1 Problem Definition

At time t, one gets some measurement, y_t, which is a function of the underlying state of the system, x_t. Sequential inference, or tracking, is the problem of describing the uncertainty over this underlying state of the system as a stream of measurements is received[10]. In the context of pdfs, at time t, this uncertainty is represented as $p(x_{1:t}|y_{1:t})$, where $x_{1:t} = \{x_1 \dots x_t\}$ is the history of states and $y_{1:t} = \{y_1 \dots y_t\}$ is the history of measurements. Hence $p(x_{1:t}|y_{1:t})$ is the uncertainty over the history of states that can be inferred from the history of measurements. From this, by the marginalisation theorem, one can obtain quantities that may be of more interest such as the pdf of the current state $p(x_t|y_{1:t})$ and the expected value of the current state $x_{t|t}$:

$$x_{t|t} \triangleq \int x_t p(x_{1:t}|y_{1:t})\, dx_t. \tag{2.1}$$

where $a \triangleq b$ denotes the definition of a.

The tracking problem is then to sequentially compute $p(x_{1:t}|y_{1:t})$. A filter is then an algorithm that provides a solution to this problem. The (fixed-interval) smoothing[34, 46, 126] problem is the estimation of the parameters of $p(x_{1:L}|y_{1:L})$ for some fixed L. Frequently, what is actually output is $p(x_t|y_{1:L})$ for every $1 \leq t \leq L$. However, because of the structure of the model, algorithms that generate such outputs can be modified trivially so that the fixed-interval smoothing problem can be solved in such a way that the parameters of $p(x_{1:L}|y_{1:L})$ are output. In any case, this is accomplished by first sequentially computing $p(x_t|y_{1:t})$ for each $t \leq L$ and then applying a backwards recursion to these pdfs. The details of the alternative schemes for computing the smoothed pdfs will be discussed later. Other versions of smoothing exist: the estimation of $p(x_{t-L}|y_{1:t})$ for some lag, L, is referred to as fixed-lag smoothing; the estimation of $p(x_t|y_{1:t+L})$ is fixed-point smoothing. Neither will be considered in detail here.

2.2.2 Model Definition

To be able to solve this problem, a model is required for the dynamics of the state and for the measurement process. Here, state-space models are used. These models are probabilistic and so represented using pdfs.

It is often assumed that the x_t process is Markov, so the state at a time step, x_{t-1}, is a sufficient statistic of the history of the process, $x_{1:t-1}$. Since the state captures all the information known about the system, the state at a time step is also assumed a sufficient statistic of the history of measurements, $y_{1:t-1}$. The current state, x_t, is therefore independent of the history of states and measurements if the previous state, x_{t-1}, is known:

$$p\left(x_t|x_{1:t-1}, y_{1:t-1}\right) = p\left(x_t|x_{t-1}, x_{1:t-2}, y_{1:t-1}\right) = p\left(x_t|x_{t-1}\right). \tag{2.2}$$

While in general, the measurement could be a function of the entire history of states, $x_{1:t}$, and the previous measurements, $y_{1:t-1}$, the case often encountered is that the measurement is independent of the history of states and the previous measurements:

$$p\left(y_t|x_{1:t}, y_{1:t-1}\right) = p\left(y_t|x_t, x_{1:t-1}, y_{1:t-1}\right) = p\left(y_t|x_t\right). \tag{2.3}$$

This is the form of the measurement model that will be considered unless explicitly stated otherwise. However, in some circumstances, correlations exist between the noise processes and these necessitate some minor modifications of the models. One can redefine the model such that the redefined state is the state of the system over some lag. However, this will introduce an increase in dimensionality which is likely to make any implementation inefficient.

An alternative, in the case of correlated measurement noise and process noise at one time step (which can come about as a result of the measurements being a function of the difference of the states at two time steps for example) is to express the measurement model as $p\left(y_t|x_{t-1:t}\right)$. This case will be revisited later in section 2.3.2.

If the measurement noises are correlated over some lag L, then the measurement model is expressed as $p\left(y_t|x_{t-L+1:t}, y_{t-L+1:t}\right)$. This phenomenon can arise if measurements are generated but then passed to the filter via some processing which processes a sliding window of measurements to generate the measurements seen by the filter. L is then the extent of the sliding window. While it may be preferable to remove this pre-processing step, such schemes are widespread in systems and often advocated as a mechanism to reduce noise and facilitate data compression. This kind of correlated noise structure is not discussed in detail here since it requires efficient fixed-lag smoothing algorithms (such as that described in [30]) to deal with this kind of model and such fixed-lag smoothers are not described here.

2.2.3 Prediction

To solve the tracking problem, each of the two components of the state-space model is considered in turn. In the prediction step, the dynamic model is used to manipulate the pdf from the previous iteration via the multiplication theorem:

$$p\left(x_{1:t}|y_{1:t-1}\right) = p\left(x_t|x_{1:t-1}, y_{1:t-1}\right) p\left(x_{1:t-1}|y_{1:t-1}\right) \tag{2.4}$$

substituting using (2.2):

$$p\left(x_{1:t}|y_{1:t-1}\right) = p\left(x_t|x_{t-1}\right) p\left(x_{1:t-1}|y_{1:t-1}\right). \tag{2.5}$$

$p\left(x_{1:t}|y_{1:t-1}\right)$ is then referred to as the predicted pdf.

Note that

$$p\left(x_t|y_{1:t-1}\right) = \int p\left(x_{1:t}|y_{1:t-1}\right) dx_{1:t-1} \tag{2.6}$$

$$= \int p\left(x_t|x_{t-1}\right) p\left(x_{t-1}|y_{1:t-1}\right) dx_{t-1} \tag{2.7}$$

so it is possible to express the prediction in terms of the updating of a pdf over a space that does not increase in dimensionality with t.

Here, the recursion starts with a proper prior that integrates to unity, $p\left(x_0\right) \triangleq p\left(x_{1:0}\right) \triangleq p\left(x_{1:0}|y_{1:0}\right)$ (ie. $p\left(x_{1:t-1}|y_{1:t-1}\right)$ with $t = 1$). This prior is assumed to be known and able to be sampled from. This does exclude from the discussion the possibility of using truly diffuse priors (with infinite variance) as in [34], but does not exclude the possibility of using very large variances which may for practical purposes be sufficient[1]. Some later discussion will describe the *information filter*, which makes it is possible to implement a Kalman filter in such a way that it can cope with infinite prior variances. However, it is non-obvious how to implement other sequential filters with such priors. It should be emphasised that the use of such diffuse priors can be thought to be suspect since the parameterisation of the state is often very informative in such cases; for example a uniform distribution over x-y position is informative with respect to range. As a result, the author asserts that such diffuse priors should be used with caution.

2.2.4 Update

Bayes theorem can then be used to manipulate this predicted pdf using the measurement model and the most recently received measurement:

$$p\left(x_{1:t}|y_{1:t}\right) = p\left(x_{1:t}|y_t, y_{1:t-1}\right) = \frac{p\left(y_t|x_{1:t}, y_{1:t-1}\right) p\left(x_{1:t}|y_{1:t-1}\right)}{p\left(y_t|y_{1:t-1}\right)} \tag{2.8}$$

substituting using (2.3):

$$= \frac{p\left(y_t|x_t\right) p\left(x_{1:t}|y_{1:t-1}\right)}{p\left(y_t|y_{1:t-1}\right)} \tag{2.9}$$

where the denominator is essentially just a normalising constant.

[1] It should be noted that the use of such priors will make pronounced any nonlinearity that is present; this in turn is likely to make it essential that care is taken in designing the filter. What nonlinearity means in this context and why this necessitates care in filter design is explained in the following sections.

As before, this can be expressed in terms of the single state, x_t

$$p\left(x_t|y_{1:t}\right) = p\left(x_t|y_t, y_{1:t-1}\right) = \frac{p\left(y_t|x_t\right) p\left(x_t|y_{1:t-1}\right)}{p\left(y_t|y_{1:t-1}\right)}. \tag{2.10}$$

So, (2.5) and (2.9) or equally (2.7) and (2.10) form a solution to the tracking problem. The question that remains is how to carry out the required calculations in practice.

2.2.5 Fixed-Interval Smoothing

There are two basic forms of fixed-interval smoother; the two-filter form[47] and the filter smoother[111]. Both these forms use as input the filtered distributions, $p\left(x_t|y_{1:t}\right)$. The approaches will be discussed in turn. Note that while fixed-interval smoothers are often implemented to calculate $p\left(x_t|y_{1:L}\right)$, the structure of the model dictates that:

$$p\left(x_{1:L}|y_{1:L}\right) = p\left(x_1|y_{1:L}\right) \prod_{t=2}^{L} p\left(x_t|x_{t-1}, y_{1:L}\right) \tag{2.11}$$

$$= p\left(x_1|y_{1:L}\right) \prod_{t=2}^{L} \frac{p\left(x_t, x_{t-1}|y_{1:L}\right)}{p\left(x_{t-1}|y_{1:L}\right)} \tag{2.12}$$

$$\tag{2.13}$$

As a result, the parameters of $p\left(x_t, x_{t-1}|y_{1:L}\right)$ provide a succinct and complete description of $p\left(x_{1:L}|y_{1:L}\right)$. This concept will be used extensively in chapter 6 where a similar Markov structure is exploited in the context of inference in images.

Two-Filter Form

This form of smoother consists of running two filters, one that uses the forward dynamics and one that uses the reverse time dynamics. The smoother then combines the two pdfs to deduce the smoothed distribution.

$$p\left(x_t|y_{1:L}\right) = \frac{p\left(x_t|y_{1:t}\right) p\left(y_{t+1:L}|x_t, y_{1:t}\right)}{p\left(y_{t+1:L}|y_{1:t}\right)} \tag{2.14}$$

$$= \frac{p\left(x_t|y_{1:t}\right) p\left(y_{t+1:L}|x_t\right)}{p\left(y_{t+1:L}|y_{1:t}\right)} \tag{2.15}$$

$$= \frac{p\left(x_t|y_{1:t}\right)}{p\left(y_{t+1:L}|y_{1:t}\right)} \int p\left(x_{t+1}, y_{t+1:L}|x_t\right) dx_{t+1} \tag{2.16}$$

$$= \frac{p\left(x_t|y_{1:t}\right)}{p\left(y_{t+1:L}|y_{1:t}\right)} \int \frac{p\left(x_t|x_{t+1}, y_{t+1:L}\right)}{p\left(x_t\right)} p\left(x_{t+1}, y_{t+1:L}\right) dx_{t+1} \tag{2.17}$$

$$= \frac{p\left(x_t|y_{1:t}\right)}{p\left(y_{t+1:L}|y_{1:t}\right)} \int \frac{p\left(x_t|x_{t+1}, y_{t+1:L}\right)}{p\left(x_t\right)} p\left(x_{t+1}|y_{t+1:L}\right) p\left(y_{t+1:L}\right) dx_{t+1} \tag{2.18}$$

$$\underbrace{p\left(x_t|y_{1:L}\right)}_{\text{Smoothed pdf}} \propto \underbrace{p\left(x_t|y_{1:t}\right)}_{\text{Filtered pdf}} \int \underbrace{\frac{1}{p\left(x_t\right)}}_{\text{Prior pdf}} \underbrace{p\left(x_t|x_{t+1}\right)}_{\text{Reverse dynamics}} \underbrace{p\left(x_{t+1}|y_{t+1:L}\right)}_{\text{Backwards filtered pdf}} dx_{t+1} \tag{2.19}$$

$p(x_t)$ is then needed to account for *double counting* and can be calculated as the integral of the prior and the dynamics up to the current time:

$$p(x_t) = \int p(x_0) \prod_{t=0}^{t-1} p(x_{t'+1}|x_{t'})\, dx_{0:t-1} \tag{2.20}$$

Note that, as a result of Bayes rule, the reverse dynamics also use the prior on the state:

$$p(x_t|x_{t+1}) = \frac{p(x_{t+1}|x_t)\, p(x_t)}{p(x_{t+1})} \tag{2.21}$$

So, $p(x_{t+1}|x_t)$ is not proportional to $p(x_t|x_{t+1})$ as is often incorrectly assumed both implicitly and explicitly. This will be explored in the context of a linear Gaussian model in section 2.3.3.

By a similar argument, a method for obtaining $p(x_{t-1:t}|y_{1:L})$ can be obtained, making it possible to deduce all the parameters of $p(x_{1:L}|y_{1:L})$ with a computational time that scales linearly with L:

$$p(x_{t-1:t}|y_{1:L}) = \frac{p(x_{t-1:t}|y_{1:t})}{p(y_{t+1:L}|y_{1:t})} \int p(x_{t+1}, y_{t+1:L}|x_{t:t-1})\, dx_{t+1} \tag{2.22}$$

$$\propto p(x_{t-1:t}|y_{1:t}) \int \frac{1}{p(x_t)} p(x_t|x_{t+1})\, p(x_{t+1}|y_{t+1:L})\, dx_{t+1} \tag{2.23}$$

$$\underbrace{p(x_{t-1:t}|y_{1:L})}_{\text{Smoothed pdf}} = \underbrace{p(x_{t-1:t}|y_{1:t})}_{\text{Filtered pdf}} \int \underbrace{\frac{1}{p(x_t)}}_{\text{Prior pdf}} \underbrace{p(x_t|x_{t+1})}_{\text{Reverse dynamics}} \int \underbrace{p(x_{t+1:t+2}|y_{t+1:L})}_{\text{Backwards filtered pdf}} dx_{t+2} dx_{t+1}. \tag{2.24}$$

The backwards filter operates much in the same way as a normal filter, but uses the reverse dynamics:

$$p(x_{t:L}|y_{t+1:L}) = p(x_t|x_{t+1})\, p(x_{t+1:L}|y_{t+1:L}) \tag{2.25}$$

$$p(x_{t:L}|y_{t:L}) = \frac{p(y_t|x_t)\, p(x_{t:L}|y_{t+1:L})}{p(y_t|y_{t+1:L})} \tag{2.26}$$

or equivalently:

$$p(x_t|y_{t+1:L}) = \int p(x_t|x_{t+1})\, p(x_{t+1}|y_{t+1:L})\, dx_{t+1} \tag{2.27}$$

$$p(x_t|y_{t:L}) = \frac{p(y_t|x_t)\, p(x_t|y_{t+1:L})}{p(y_t|y_{t+1:L})} \tag{2.28}$$

A similar alternative approach is to obtain the fixed lag pdf by combining the filtered pdf with a likelihood function[30] that is equivalent to the likelihood of the future measurements:

$$p(x_t|y_{1:L}) \propto p(x_t|y_{1:t})\, p(y_{t+1:L}|x_t, y_{1:t}) \tag{2.29}$$

$$= \underbrace{p(x_t|y_{1:t})}_{\text{Filtered pdf}} \underbrace{p(y_{t+1:L}|x_t)}_{\text{Likelihood function}} \tag{2.30}$$

Note that the likelihood is a function of x_t and so is not a pdf, but can be sequentially computed using the following reverse time recursion:

$$p(y_{t+1:L}|x_t) = \int p(y_{t+1}, y_{t+2:L}, x_{t+1}|x_t)\, dx_{t+1} \tag{2.31}$$

$$= \int p(x_{t+1}|x_t)\, p(y_{t+2:L}|x_{t+1}, x_t)\, p(y_{t+1}|y_{t+2:L}, x_{t+1}, x_t)\, dx_{t+1} \tag{2.32}$$

$$= \int p(x_{t+1}|x_t)\, p(y_{t+2:L}|x_{t+1})\, p(y_{t+1}|x_{t+1})\, dx_{t+1}. \tag{2.33}$$

Filter Smoother

The filter smoother avoids this need to calculate the reverse time dynamics and $p(x_t)$ by having a backwards recursion that sequentially computes the smoothed distributions:

$$p(x_t|y_{1:L}) = \int p(x_{t:t+1}|y_{1:L})\, dx_{t+1} \tag{2.34}$$

$$= \int p(x_{t+1}|y_{1:L})\, p(x_t|x_{t+1}, y_{1:L})\, dx_{t+1} \tag{2.35}$$

$$= \int p(x_{t+1}|y_{1:L})\, p(x_t|x_{t+1}, y_{1:t})\, dx_{t+1} \tag{2.36}$$

$$= \int p(x_{t+1}|y_{1:L})\, \frac{p(x_{t+1}|x_t, y_{1:t})\, p(x_t|y_{1:t})}{p(x_{t+1}|y_{1:t})} dx_{t+1} \tag{2.37}$$

$$\underbrace{p(x_t|y_{1:L})}_{\text{Smoothed pdf at } t} = \int \underbrace{p(x_{t+1}|y_{1:L})}_{\text{Smoothed pdf at } t+1} \frac{\overbrace{p(x_{t+1}|x_t)}^{\text{Dynamics}}\ \overbrace{p(x_t|y_{1:t})}^{\text{Filtered pdf at } t}}{\underbrace{\int p(x_{t+1}|x'_t)\, p(x'_t|y_{1:t})\, dx'_t}_{\text{Normalising constant}}} dx_{t+1}. \tag{2.38}$$

Again, it is possible to use this approach to deduce the parameters of $p(x_{t-1:t}|y_{1:L})$:

$$p(x_{t-1:t}|y_{1:L}) = \int p(x_{t+1}|y_{1:L})\, p(x_{t-1:t}|x_{t+1}, y_{1:t})\, dx_{t+1} \tag{2.39}$$

$$= \int p(x_{t+1}|y_{1:L})\, \frac{p(x_{t+1}|x_t)\, p(x_{t-1:t}|y_{1:t})}{\int p(x_{t+1}|x'_t)\, p(x_{t-1}, x'_t|y_{1:t})\, dx'_t} dx_{t+1} \tag{2.40}$$

$$\underbrace{p(x_{t-1:t}|y_{1:L})}_{\text{Smoothed pdf at } t-1:t} = \int\int \underbrace{p(x''_t, x_{t+1}|y_{1:L})}_{\text{Smoothed pdf at } t:t+1} dx''_t \frac{\overbrace{p(x_{t+1}|x_t)}^{\text{Dynamics}}\ \overbrace{p(x_{t-1:t}|y_{1:t})}^{\text{Filtered pdf at } t-1:t}}{\underbrace{\int p(x_{t+1}|x'_t)\, p(x_{t-1}, x'_t|y_{1:t})\, dx'_t}_{\text{Normalising constant}}} dx_{t+1}. \tag{2.41}$$

2.3 Linear Gaussian Models

2.3.1 Assumptions

The filtering calculations are (relatively) simple and exact when the models are linear and Gaussian. That is:

- The state space is continuous
- $p(x_0) = \mathcal{N}(x_0; m_0, C_0)$
- $p(x_t|x_{t-1})$ and $p(y_t|x_t)$ are linear Gaussian.

where $\mathcal{N}(x; m, C)$ is a Gaussian distribution for x parameterised by a mean, m, and a covariance, C.

2.3.2 Filtering

If these assumptions hold, it is possible to recursively calculate the parameters of $p(x_t|y_{1:t})$. The reason that this works is that the restriction on the model ensure that the resulting distribution is exactly Gaussian[48]. The problem then becomes one of deducing the parameters of $p(x_t|y_{1:t})$[55]. These parameters can be calculated by noting that multivariate normal distributions have the following property[46, pg 165]): if a state partitions $x = (x_a^T, x_b^T)^T$ with mean $\mu = (\mu_a^T, \mu_b^T)^T$ and covariance

$$\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \tag{2.42}$$

then, the distribution of x_a given x_b is also multivariate normal with mean

$$\mu_{a|b} = \mu_a + \Sigma_{aa}\Sigma_{bb}^{-1}(x_b - \mu_b) \tag{2.43}$$

and covariance

$$\Sigma_{aa|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba} \tag{2.44}$$

where it should be evident that it is assumed that Σ_{bb} is positive definite.

Correlated Measurement and Process Noise

This idea is at the heart of Kalman filtering and makes it straightforward to consider the following system (which corresponds to the case of correlated measurement and process noises[2]; this case is taken as the starting point since all the forms of the Kalman filter that will be considered are special cases of this formulation):

$$x_t = Ax_{t-1} + b_t + \Gamma\omega_t^x \tag{2.45}$$

$$y_t = Hx_t + \Psi\omega_t^y + \Lambda\omega_t^x \tag{2.46}$$

and

$$p(x_{t-1}|y_{1:t-1}) = \mathcal{N}(x_{t-1}; x_{t-1|t-1}, P_{t-1|t-1}) \tag{2.47}$$

$$\omega_t^x \sim \mathcal{N}(\omega_t^x; \mathbb{O}, \mathbb{I}) \tag{2.48}$$

$$\omega_t^y \sim \mathcal{N}(\omega_t^y; \mathbb{O}, \mathbb{I}) \tag{2.49}$$

where $\mathcal{N}(x; m, C)$ is the Gaussian distribution over the values of x with mean m and covariance C. $\mathbb{O}$ is an appropriately sized matrix of zeros and $\mathbb{I}$ is an appropriately sized identity matrix; the dimensions of ω_t^x and ω_t^y are respectively the same as that of the state and that of the measurement. Then

$$p(x_t|y_{1:t-1}) = \mathcal{N}(x_t; x_{t|t-1}, P_{t|t-1}) \tag{2.50}$$

$$p(x_t|y_{1:t}) = \mathcal{N}(x_t; x_{t|t}, P_{t|t}) \tag{2.51}$$

[2]While one can accommodate correlated measurement and process noises, by augmenting the state in such a way as to be able to use the *standard* Kalman filter recursions, the description given here is a more general and succinct form of the filter.

where

$$x_{t|t-1} = Ax_{t-1|t-1} + b_t \tag{2.52}$$
$$P_{t|t-1} = AP_{t-1|t-1}A^T + \Gamma\Gamma^T \tag{2.53}$$
$$y_{t|t-1} = Hx_{t|t-1} \tag{2.54}$$
$$x_{t|t} = x_{t|t-1} + P_{xy}P_{yy}^{-1}(y_t - y_{t|t-1}) \tag{2.55}$$
$$P_{t|t} = P_{t|t-1} - P_{xy}P_{yy}^{-1}P_{xy}^T \tag{2.56}$$

with

$$P_{xy} = \mathbb{E}\left[\left(x_t - x_{t|t}\right)\left(y_t - y_{t|t-1}\right)^T\right] \tag{2.57}$$
$$= \mathbb{E}\left[\left(x_t - x_{t|t}\right)\left(H\left(x_t - x_{t|t-1}\right) + \Psi w_t^y + \Lambda w_t^x\right)^T\right] \tag{2.58}$$
$$= P_{t|t-1}H^T + \mathbb{E}\left[\left(x_t - x_{t|t-1}\right)w_t^{yT}\right]\Psi^T + \mathbb{E}\left[\left(x_t - x_{t|t-1}\right)w_t^{xT}\right]\Lambda^T \tag{2.59}$$
$$= P_{t|t-1}H^T + \Gamma\Lambda^T \tag{2.60}$$
$$P_{yy} = \mathbb{E}\left[\left(y_t - y_{t|t-1}\right)\left(y_t - y_{t|t-1}\right)^T\right] \tag{2.61}$$
$$= \mathbb{E}\left[\left(H\left(x_t - x_{t|t-1}\right) + \Psi w_t^y + \Lambda w_t^x\right)\left(H\left(x_t - x_{t|t-1}\right) + \Psi w_t^y + \Lambda w_t^x\right)^T\right] \tag{2.62}$$
$$= HP_{t|t-1}H^T + \Psi\Psi^T + \Lambda\Lambda^T + H\mathbb{E}\left[\left(x_t - x_{t|t-1}\right)w_t^{xT}\right]\Lambda^T + \Lambda\mathbb{E}\left[w_t^x\left(x_t - x_{t|t-1}\right)^T\right]H^T \tag{2.63}$$
$$= HP_{t|t-1}H^T + \Psi\Psi^T + \Lambda\Lambda^T + H\Gamma\Lambda^T + \Lambda\Gamma^T H^T \tag{2.64}$$

Uncorrelated Noises

The more frequently encountered case of the Kalman filter occurs when $\Lambda = 0$, $b_t = 0$, $\Gamma\Gamma^T = Q$ and $\Psi\Psi^T = R$. In this case:

$$x_{t|t-1} = Ax_{t-1|t-1} \tag{2.65}$$
$$P_{t|t-1} = AP_{t-1|t-1}A^T + Q \tag{2.66}$$
$$x_{t|t} = x_{t|t-1} + P_{xy}P_{yy}^{-1}(y_t - y_{t|t-1}) \tag{2.67}$$
$$P_{t|t} = P_{t|t-1} - P_{xy}P_{yy}^{-1}P_{xy}^T \tag{2.68}$$

where

$$y_{t|t-1} = Hx_{t|t-1} \tag{2.69}$$
$$P_{yy} = HP_{t|t-1}H^T + R \tag{2.70}$$
$$P_{xy} = P_{t|t-1}H^T \tag{2.71}$$

where some of the equations are unchanged, but have been repeated for easy reference.

The product $P_{xy}P_{yy}^{-1}$ is then often referred to as the Kalman gain, K, since it is a weighting factor for the impact of the innovation, $y_t - y_{t|t-1}$, on the estimate of the state, $x_{t|t}$:

$$x_{t|t} = x_{t|t-1} + K(y_t - y_{t|t-1}) \tag{2.72}$$

By substituting for $y_{t|t-1}$, it is possible to show the update of the mean as a weighted sum of the prediction and (the projection of) the measurement:

$$\begin{aligned} x_{t|t} =& x_{t|t-1} + K\left(y_t - y_{t|t-1}\right) && (2.73)\\ =& x_{t|t-1} + K\left(y_t - KHx_{t|t-1}\right) && (2.74)\\ =& \left(I - KH\right)x_{t|t-1} + Ky_t && (2.75)\end{aligned}$$

P_{yy} is often called the covariance of the innovations, S, and its inverse is sometimes calculated by employing the matrix inversion lemma[3]:

$$\begin{aligned} S^{-1} =& \left(HP_{t|t-1}H^T + R\right)^{-1} && (2.81)\\ =& R^{-1} - \left(H^TR^{-1}\right)^T\left(P_{t|t-1}^{-1} + H^TR^{-1}H\right)^{-1}\left(H^TR^{-1}\right) && (2.82)\end{aligned}$$

The equation for $P_{t|t}$ is often described in alternative equivalent forms:

$$\begin{aligned} P_{t|t} =& P_{t|t-1} - P_{xy}P_{yy}^{-1}P_{xy}^T && (2.83)\\ =& P_{t|t-1} - P_{xy}P_{yy}^{-1}P_{yy}P_{yy}^{-1}P_{xy}^T && (2.84)\\ =& P_{t|t-1} - KSK^T && (2.85)\end{aligned}$$

or

$$\begin{aligned} P_{t|t} =& P_{t|t-1} - P_{xy}P_{yy}^{-1}\left(P_{t|t-1}H^T\right)^T && (2.86)\\ =& \left(I - KH\right)P_{t|t-1} && (2.87)\end{aligned}$$

or by the matrix inversion lemma

$$\begin{aligned} P_{t|t} =& P_{t|t-1} - \left(P_{t|t-1}H^T\right)\left(HP_{t|t-1}H^T + R\right)^{-1}\left(P_{t|t-1}H^T\right)^T && (2.88)\\ =& \left(P_{t|t-1}^{-1} - H^TR^{-1}H\right)^{-1} && (2.89)\end{aligned}$$

[3]For three matrices, A, B and C, application of the matrix inversion lemma implies that:

$$\left(A + BC^{-1}B^T\right)^{-1} = A^{-1} - \left(B^TA^{-1}\right)^T\left(C + B^TA^{-1}B\right)^{-1}\left(B^TA^{-1}\right) \quad (2.76)$$

since if one substitutes (2.76) for the inverse:

$$\begin{aligned} \left(A + BC^{-1}B^T\right)\left(A + BC^{-1}B^T\right)^{-1} =& I + BC^{-1}B^TA - 1 - \left(A + BC^{-1}B^T\right)\left(B^TA^{-1}\right)^T\left(C + B^TA^{-1}B\right)^{-1}\left(B^TA^{-1}\right) && (2.77)\\ =& I + BC^{-1}B^TA^{-1} - \left(B + BC^{-1}B^T\left(B^TA^{-1}\right)^T\right)\left(C + B^TA^{-1}B\right)^{-1}\left(B^TA^{-1}\right) && (2.78)\\ =& I + BC^{-1}B^TA^{-1} - BC^{-1}\left(C + B^TA^{-1}B\right)\left(C + B^TA^{-1}B\right)^{-1}\left(B^TA^{-1}\right) && (2.79)\\ =& I + BC^{-1}B^TA^{-1} - BC^{-1}B^TA^{-1} = I. && (2.80)\end{aligned}$$

where it has been used that $P_{yy} = S$ and $P_{t|t-1}$ are symmetric (since they are covariance matrices). By substituting, it also becomes evident that:

$$x_{t|t} = (I - KH)\, x_{t|t-1} + Ky_t = P_{t|t} P_{t|t-1}^{-1} x_{t|t-1} + Ky_t \tag{2.90}$$

It should be emphasised that while there are a large number of different forms for the filter, the formulations are equivalent.

Information Matrix Form

If, typically as a result of the use of *diffuse* initiation, the covariance matrices have infinite eigenvalues, then the above implementation can be modified[30] to only manipulate the inverse of the covariance matrices, $P_{t|t}^{-1}$, and the information state, $P_{t|t}^{-1} x_{t|t}$. The description is equivalent, since the derivation simply applies the matrix inversion lemma, but easier to implement since mechanisms for coping with infinite eigenvalues need not be considered:

$$\begin{aligned}
P_{t|t-1}^{-1} &= \left(AP_{t-1|t-1}A^T + Q\right)^{-1} && (2.91)\\
&= Q^{-1} - \left(A^T Q^{-1}\right)^T \left(P_{t-1|t-1}^{-1} + A^T Q^{-1} A\right)^{-1} \left(A^T Q^{-1}\right) && (2.92)\\
P_{t|t-1}^{-1} x_{t|t-1} &= P_{t|t-1}^{-1} A P_{t-1|t-1} \left(P_{t-1|t-1}^{-1} x_{t-1|t-1}\right) && (2.93)\\
&= \left(AP_{t-1|t-1}A^T + Q\right)^{-1} A P_{t-1|t-1} \left(P_{t-1|t-1}^{-1} x_{t-1|t-1}\right) && (2.94)\\
&= \left(\left(\left(AP_{t-1|t-1}A^T + Q\right)^{-1} A P_{t-1|t-1}\right)^{-1}\right)^{-1} \left(P_{t-1|t-1}^{-1} x_{t-1|t-1}\right) && (2.95)\\
&= \left(P_{t-1|t-1}^{-1} A^{-1} A P_{t-1|t-1} A^T + P_{t-1|t-1}^{-1} A^{-1} Q\right)^{-1} \left(P_{t-1|t-1}^{-1} x_{t-1|t-1}\right) && (2.96)\\
&= \left(A^T + P_{t-1|t-1}^{-1} A^{-1} Q\right)^{-1} \left(P_{t-1|t-1}^{-1} x_{t-1|t-1}\right) && (2.97)\\
P_{t|t}^{-1} &= P_{t|t-1}^{-1} + H^T R^{-1} H && (2.98)\\
P_{t|t}^{-1} x_{t|t} &= P_{t|t-1}^{-1} x_{t|t-1} + H^T R^{-1} y_t && (2.99)
\end{aligned}$$

Note that this form of the filter requires A^{-1} to exist. Indeed, it is possible to express the above solely in terms of A^{-1} without any reference to A itself. This was exploited in [30]. In [30], such an information matrix form of a Kalman filter was used to process data in reverse-time order. Since the filter only used the reverse of the reverse-time dynamics, there was then no necessity that the inverse of the forwards dynamics, A^{-1}, existed.

Model-specific Variants

If the system matrices are known and constant over time, then the covariances can be precalculated. It is also worth noting that in such cases, the covariance will tend to a constant value, which is defined by the system matrices. This means that, after some initial period, the uncertainty is constant. This steady state is the result of a balance between the increase in uncertainty as a result of the prediction step and the reduction in uncertainty due to the update step.

This observation leads to the $\alpha - \beta$ and $\alpha - \beta - \gamma$ filters, which can be viewed as a special case of the Kalman filter that assume that this convergence has occurred and use the resulting (fixed) value for the Kalman gain. There is then no need to store the covariances themselves. The two filters respectively assume that the system evolves according to constant velocity and constant acceleration models. These models are discussed in detail in chapter 3.

2.3.3 Smoothing

Application of either of the aforementioned fixed-interval smoothing algorithms when the models are linear and Gaussian is straightforward as a result of the following relationships, based on the fact that the product of two normal distributions is itself a normal distribution (which has already been used in the prediction step of the Kalman filter):

$$\mathcal{N}(x; m, C) = \mathcal{N}(x; m_1, C_1)\,\mathcal{N}(x; m_2, C_2) \tag{2.100}$$

where

$$m = C\left(C_1^{-1} m_1 + C_2^{-1} m_2\right) \tag{2.101}$$

$$C = \left(C_1^{-1} + C_2^{-1}\right)^{-1} \tag{2.102}$$

so

$$\mathcal{N}(x; m, C) = \mathcal{N}(x; m_1, C_1)\,\frac{1}{\mathcal{N}(x; m_2, C_2)} \tag{2.103}$$

where

$$m = C\left(C_1^{-1} m_1 - C_2^{-1} m_2\right) \tag{2.104}$$

$$C = \left(C_1^{-1} - C_2^{-1}\right)^{-1} \tag{2.105}$$

where it has been assumed that $C_1^{-1} > C_2^{-1}$ so that this division is valid (this is the case in the smoothing examples).

Just as with a Kalman filter, if the system matrices are constant, a steady state will exist in terms of the covariances. Both the forward-time and backward-time filters will tend to a steady state. Thus, the combined smoothed distribution will have a near-constant covariance between an initial and final period. This covariance will be tighter than either the forward-time or backward-time filter's covariances.

In the case of linear Gaussian dynamics, it is possible to deduce the parameters of the reverse dynamics $p(x_t|x_{t+1})$, which is needed by the two-filter form of the smoother, from the initial prior, $p(x_0)$, and the forward dynamics, $p(x_{t+1}|x_t)$. Note that:

$$p(x_t) = \int p(x_{0:t})\, dx_{0:t-1} \tag{2.106}$$

$$= \int p(x_0) \prod_{t'=1}^{t} p(x_{t'}|x_{t'-1})\, dx_{0:t-1} \tag{2.107}$$

$$= \int \mathcal{N}(x_0; m_0, C_0) \prod_{t'=1}^{t} \mathcal{N}(x_{t'}; A x_{t'-1}, Q)\, dx_{0:t-1} \tag{2.108}$$

$$= \mathcal{N}\left(x_t; m_{t|\emptyset}, P_{t|\emptyset}\right) \tag{2.109}$$

so the parameters of $p(x_t)$ can be obtained by running the prediction step of the Kalman filter (with no measurements) from the initial prior to t via all the intermediate times[4]. One can then calculate the parameters of $p(x_t|x_{t+1})$ by recalling that:

$$\begin{aligned} p(x_t|x_{t+1}) \propto & p(x_{t+1}|x_t)\, p(x_t) && (2.110)\\ = & \mathcal{N}(x_{t+1}; Ax_t, Q)\, \mathcal{N}\left(x_t; m_{t|\emptyset}, P_{t|\emptyset}\right) && (2.111)\\ = & \mathcal{N}\left(x_t; m_t^b, P_t^b\right) && (2.112) \end{aligned}$$

where by analogy with the update step in the Kalman filter, the conditioning on x_{t+1} gives rise to:

$$\begin{aligned} P_t^b = & \left(A^T {Q_t}^{-1} A + {P_{t|\emptyset}}^{-1}\right)^{-1} && (2.113)\\ m_t^b = & P_t^b {P_{t|\emptyset}}^{-1} m_{t|\emptyset} + P_{t|\emptyset} A^T \left(Q_t + A P_{t|\emptyset} A^T\right)^{-1} x_{t+1}. && (2.114) \end{aligned}$$

While it is true that as $P_{t|\emptyset}$ tends to infinity then:

$$\begin{aligned} \lim_{P_{t|\emptyset}\to\infty} P_t^b = & A^{-1} Q_t {A^T}^{-1} && (2.115)\\ \lim_{P_{t|\emptyset}\to\infty} m_t^b = & A^{-1} x_{t+1} && (2.116) \end{aligned}$$

this is an (often implicitly made) approximation. On an intuitive level this prior will remove any instabilities caused by inverting A; were A scalar and near zero, then A^{-1} would be very large in magnitude, making application of (2.116) cause the mean to rapidly diverge away from the origin. The prior enforces the constraint that the initial conditions make this very unlikely. Another intuitive explanation for the need for the prior is that the initial conditions need to be consistent; starting from the initial prior and then running the model forwards and then backwards should give the initial prior again. This is guaranteed by using (2.113) and (2.114).

2.4 Grid Based Filter

2.4.1 Assumptions

The Kalman filter is not the only exact algorithm for sequential inference. The grid based filter is another example of an optimal recursion resulting from a different set of assumptions:

- The state space is discrete and finite
- $p(x_0) = \sum_{i=1}^{N_{s(0)}} w_{0|0}^i \delta\left(x_0 - x_0^i\right)$
- $p\left(x_t^i | x_{t-1}^j\right)$ and $p\left(y_t | x_t^i\right)$ can be evaluated.

[4] In some cases, $p(x_t) \to p(x)$ as $t \to \infty$ and it is possible to calculate the parameters of $p(x)$. In other cases, considering an underlying continuous time system can make it possible to deduce the parameters of $p(x_t)$ directly from $p(x_0)$ without the need to calculate the intermediate distributions. However, in the general case, neither is possible and the integral has to be calculated.

where $\delta(x - x_0)$ is a delta function, a representation of a sample at x_0 that integrates to unity:

$$\delta(x - x_0) \triangleq \lim_{\Delta \to 0} \begin{cases} \frac{1}{\Delta} & x_0 - \frac{\Delta}{2} \leq x \leq x_0 + \frac{\Delta}{2} \\ 0 & \text{otherwise} \end{cases} . \tag{2.117}$$

2.4.2 Filtering

For each state x_t^i, let the conditional probability of that state, given measurements up to time t be denoted by $w_{t|t}^i$. Then, a recursion can be derived for the posterior pdf at time t.

$$p(x_{t-1}|y_{1:t-1}) = \sum_{i=1}^{N_{s(t-1)}} w_{t-1|t-1}^i \delta\left(x_{t-1} - x_{t-1}^i\right) \tag{2.118}$$

$$p(x_t|y_{1:t-1}) = \sum_{i=1}^{N_{s(t)}} w_{t|t-1}^i \delta(x_t - x_t^i) \tag{2.119}$$

$$p(x_t|y_{1:t}) = \sum_{i=1}^{N_{s(t)}} w_{t|t}^i \delta(x_t - x_t^i) \tag{2.120}$$

The weights can be calculated in terms of the transition and observation probabilities.

$$w_{t|t-1}^i = \sum_{j=1}^{N_{s(t-1)}} w_{t-1|t-1}^j p\left(x_t^i|x_{t-1}^j\right) \tag{2.121}$$

$$w_{t|t}^i = \frac{w_{t|t-1}^i p\left(y_t|x_t^i\right)}{\sum_{j=1}^{N_{s(t)}} w_{t|t-1}^j p\left(y_t|x_t^j\right)} \tag{2.122}$$

The above does assume that $p\left(x_t^i|x_{t-1}^j\right)$ and $p\left(y_t|x_t^i\right)$ are known, but does not constrain the particular form of these discrete distributions. Note that the algorithm's computational cost is quadratic in the number of discrete states, but that these operations can be straightforwardly parallelised.

2.4.3 Smoothing

Smoothing algorithms for this grid-based filter have been used extensively in speech processing[5], where they are known as Hidden Markov Models, HMMs [70, 100, 101, 114]. Two algorithms dominate. Both are exact algorithms if the assumptions of the grid based filter hold.

First, the Viterbi algorithm[37] calculates the MAP estimate of the path through the trellis of states, that sequence of discrete states that maximises the probability of the state sequence given the data. At each iteration in the filter, the maximum probabilities (and associated path) of all the paths through the trellis up to each state for the previous time step are stored. The maximum probability[6] path through the trellis up to each state for the current time must then be an extension of one of these paths. The probability is calculated for all possible combinations of states at the current and previous time given the

[5] In this environment, the assumptions of the filter only hold approximately.

[6] To avoid errors resulting from rounding (very) small probabilities to zero, the probabilities should be manipulated in terms of logarithms.

new measurement. The maximum probability (and associated path) associated with each state at the current time is then stored for each state at the current time and the algorithm then iterates:

$$\max_{x_{1:t-2}} p\left(x^i_{t-1}, x_{1:t-2}, y_{1:t-1}\right) = w^i_{1:t-1|t-1} \tag{2.123}$$

$$\arg\max_{x_{1:t-2}} p\left(x^i_{t-1}, x_{1:t-2}, y_{1:t-1}\right) = x^i_{1:t-1|t-1} \tag{2.124}$$

$$\max_{x_{1:t-1}} p\left(x^i_{t}, x_{1:t-1}, y_{1:t}\right) = w^i_{1:t|t} \tag{2.125}$$

$$\arg\max_{x_{1:t-1}} p\left(x^i_{t}, x_{1:t-1}, y_{1:t}\right) = x^i_{1:t-1|t-1} \tag{2.126}$$

where

$$w^i_{1:t|t} = \max_j w^{i,j}_{1:t|t} \tag{2.127}$$

$$w^{i,j}_{1:t|t} = p\left(y_t|x^i_t\right) p\left(x^i_t|x^j_{t-1}\right) w^j_{1:t-1|t-1} \tag{2.128}$$

$$x^i_{1:t|t} = \left\{x^i_t, x^{j(i)}_{1:t-1|t}\right\} \tag{2.129}$$

$$j(i) = \arg\max_j w^{i,j}_{1:t|t}. \tag{2.130}$$

This algorithm is unusual in that it is an efficient (partial, in the sense that it calculates only the maximum path and its probability) solution to a smoothing problem that only requires a single (forwards) pass of the data. No backwards recursion is required.

Application of the filter-smoother algorithm to the model gives rise to the Baum-Welch [100] algorithm, which is often configured to calculate the probability[7] of each discrete state at each time epoch given the entire data sequence. It is straightforward to modify the algorithm such that it also outputs the conditional probabilities $p\left(x^i_t|x^j_{t-1}, y_{1:L}\right)$ and so completely describes the joint distribution, $p\left(x_{1:L}|y_{1:L}\right)$.

2.5 Approximate Linear Gaussian Filtering

2.5.1 Introduction

While there do exist a few special cases for which other analytic solutions exist, the family of possible models that one would like to use is far bigger than those that permit analytic solutions.

One can approximate the models as linear and Gaussian in the locality of estimates and then use a Kalman Filter with these approximating models[10]. This results in the *Extended Kalman Filter*, or EKF.

[7] It is then necessary to be able to calculate the logarithm of the sum of two probabilities by manipulating the logarithms of the probabilities, which can be done as follows:

$$\log(a+b) = \log\left(a\left(1+\frac{b}{a}\right)\right) = \log(a) + \log\left(1+\exp\left(\log(b) - \log(a)\right)\right) \tag{2.131}$$

where it should be ensured that $a \geq b$. Should $a < b$ then a and b should be interchanged in (2.131).

Example

Throughout this and subsequent sections, a concrete example will be referred to. This example consists of using (noisy) measurements of the position of some system to calculate estimates regarding the position and velocity of the system. The constant velocity model itself will be discussed in more detail in chapter 3.

So, in the example, the system is taken to evolve according to:

$$x_t = Ax_{t-1} + \omega_t^x \tag{2.132}$$

where ω_t^x, the process noise, is a sample from a Gaussian distribution:

$$p\left(\omega_t^x\right) = \mathcal{N}\left(\omega_t^x; \mathbb{O}, Q\right). \tag{2.133}$$

where

$$A = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \tag{2.134}$$

$$Q = \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} q \tag{2.135}$$

and where q is a scalar quantity that defines the size of any deviations from straight line motion and Δ is the time between measurements. The probabilistic nature of the model dictates that while the predicted position at some point in the future takes account of the velocity at the present time, the uncertainty related to this estimate grows as the extrapolation extends further into the future. It is worth noting that this model is exactly equivalent to an integrated random walk evolving in continuous time and so Q has a determinant that is greater than zero; Q is positive definite. This means that even if one knew the position and velocity at time $t-1$ and the position at time t, there would still be uncertainty over the velocity at time t. As noted by a reviewer of [74], this is related to the spline smoothing approach described in [34].

The definition of the model in terms of an equation for x_t in terms of x_{t-1} is equivalent to the following definition in terms of a distribution for x_t conditional on x_{t-1}:

$$p\left(x_t|x_{t-1}\right) = \mathcal{N}\left(x_t; Ax_{t-1}, Q\right). \tag{2.136}$$

Note the implication of the fact that the process is Markov; the position and velocity at time t define the uncertainty over the position and velocity at all future times, $t' > t$. Hence, further knowledge of the position and velocity at any previous time, $t'' < t$, has no effect on this uncertainty at the future time, t'.

In the example, imagine as simple a nonlinear measurement model as possible:

$$y_t = \left(Fx_t\right)^2 + \omega_t^y \tag{2.137}$$

where

$$p\left(\omega_t^y\right) = \mathcal{N}\left(\omega_t^y; 0, R\right) \tag{2.138}$$

$$F = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{2.139}$$

where R is a (scalar) variance defining the accuracy of the measurements.

Again, this is equivalent to a probabilistic description:

$$p\left(y_t|x_t\right) = \mathcal{N}\left(y_t; \left(Fx_t\right)^2, R\right). \tag{2.140}$$

To be able to use an EKF for the example, one just needs to linearise (2.140) about the prediction $x_{t|t-1}$:

$$p\left(y_t|x_t\right) \approx \mathcal{N}\left(y_t; y_{t|t-1} + H\left(x_t - x_{t|t-1}\right), R\right) \tag{2.141}$$

where

$$y_{t|t-1} = \left(Fx_{t|t-1}\right)^2 \tag{2.142}$$

$$H = \begin{bmatrix} 2Fx_{t|t-1} & 0 \end{bmatrix}. \tag{2.143}$$

One can now use a Kalman filter with this approximating model; here (2.54) is simply replaced with (2.142) and the Kalman filter then operates as previously discussed[8]. $\Delta = 1$, $R = 1000$ and $q = 1$ and the filter is initialised with:

$$p\left(x_0\right) = \mathcal{N}\left(x_0; x_0^{\text{true}} + \delta, C\right) \tag{2.144}$$

where

$$p\left(\delta\right) = \mathcal{N}\left(\delta; \mathbb{O}, C\right) \tag{2.145}$$

$$C = \begin{bmatrix} 20 & 0 \\ 0 & 1 \end{bmatrix}. \tag{2.146}$$

Two cases are considered. In the first case, $x_0^{\text{true}} = \begin{bmatrix} 500 & 0 \end{bmatrix}$. In the second, $x_0^{\text{true}} = \begin{bmatrix} 5 & 0 \end{bmatrix}$. Nine simulations and runs of the EKF result in the estimated position (and true position) shown in figures 2.1 and 2.2 for the first and second cases respectively. The obvious question to ask is why the EKF, in a couple of examples seems to go drastically wrong.

The answer is that this approximation to the models is good when the nonlinearity and departure from Gaussian behaviour is small. To consider if this is the case, it is instructive to consider what nonlinearity means. In this context what matters is how good an approximation would be achieved by using a local linearisation of the models. So, by looking at the expected departure from linear behaviour, it is possible to get a handle on the amount of nonlinearity; if the variation in the linearity at different points in the posterior is large with respect to the linear coefficients then the nonlinearity is pronounced. So nonlinearity in this context is a function of both the nonlinear model and the diffuse nature of the pdf; an apparently less nonlinear model of two candidates can appear more nonlinear to the EKF if the distribution used with this less nonlinear model is more diffuse. In general, as the signal-to-noise drops, it will become increasingly likely that such scenarios are encountered since the the distribution will typically become more diffuse as the signal-to-noise falls; the underlying problem is the nonlinearity, not such things as the scale of the data.

[8] When the dynamics are nonlinear (2.52) will also need to be replaced with a linearisation based on $x_{t-1|t-1}$.

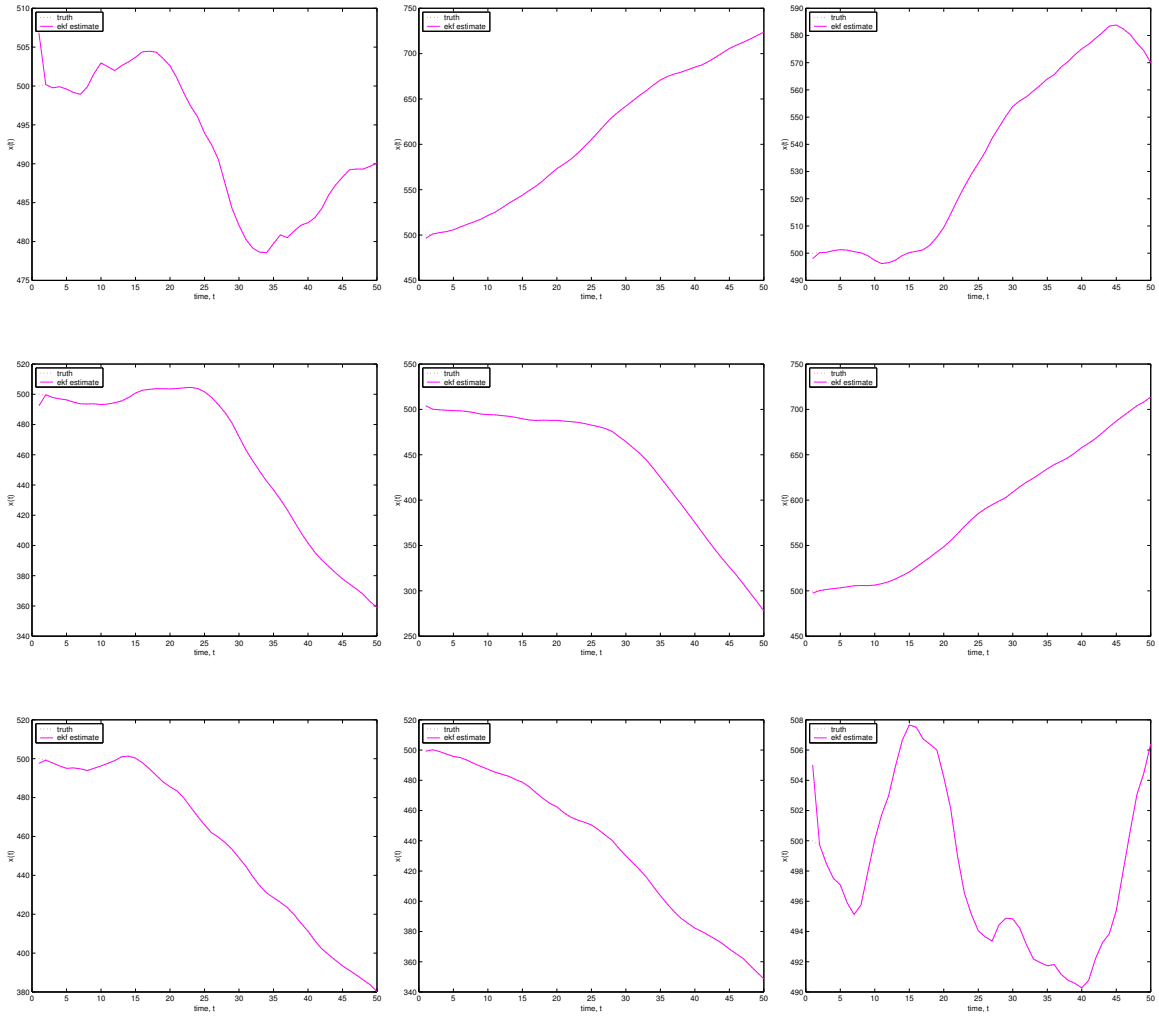


Figure 2.1: *Nine runs of the EKF for the model defined by (2.132) to (2.140) with* $x_0 = [500, 0]$. *Note that in all cases, the lines are sufficiently closely spaced that it is difficult to see that two lines are present.*

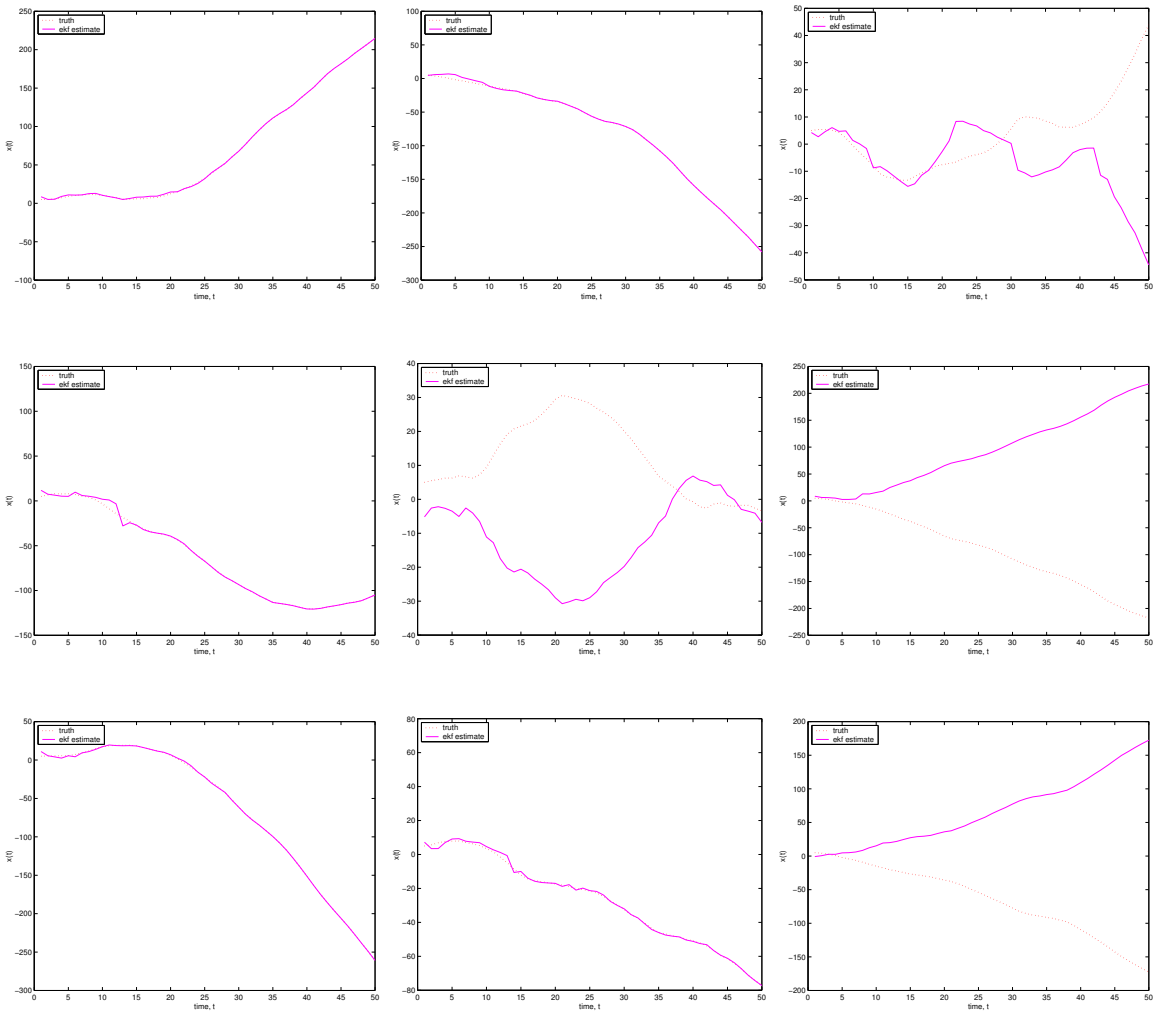


Figure 2.2: *Nine runs of the EKF for the model defined by (2.132) to (2.140) with* $x_0 = [5, 0]$.

In the example, the same change in the state will cause a much bigger (percentage) change in the linearisation when the position is small than when the position is large. So, the linear approximation is valid when the state is large and isn't valid when the state is small. This explains the observed behaviour.

Indeed, particle filtering was initially proposed as an alternative to the EKF for bearings only tracking[42]. When viewing a target from a platform such as an aircraft, certain sensors (eg. a camera) will give measurements of bearings with no associated estimates of range. The idea is to attempt to use the incoming bearings measurements to infer the two dimensional position, this is often made possible by the sensing platform out-manoeuvring the target and so moving relative to the target such that it is possible to triangulate over time.

This is an application for which the nonlinearity can be too severe to use an EKF, and for which alternatives to the EKF (such as the particle filter) are often advocated. Since only bearing (and so no range) is observed, the uncertainty over position is typically large and the distribution therefore diffuse. Small changes in position at high range will have much less of an effect on the linear approximation than the same change at small range. So, the nonlinearity is likely to become pronounced and an EKF can then be outperformed by a particle filter. An alternative approach is to use a bank of EKFs, each of which initially considers the target to be within some portion of the ranges. Since, for each such range-parameterised EKF, the nonlinearity is less pronounced, one can then achieve similar performance to that possible with a particle filter at a fraction of the computational cost[3].

2.5.2 Higher Order EKFs

Such a local approximation of the equations may be a sufficient description of the non-linearity. However, it is common that it is not. A better approximation can be made by considering the above approximation as using the first term in Taylor expansions of the non-linear functions. A higher order EKF that retains further terms in the Taylor expansions exists and results in a closer approximation to the true posterior. The additional complexity has prohibited its widespread use.

2.5.3 Unscented Kalman Filter

Another example of how to cater with mildly nonlinear non-Gaussian models is the Unscented Kalman filter[51, 54, 122], UKF. The UKF uses a deterministic sampling scheme to deduce the parameters of $p(x_t|y_{1:t})$ and matches the moments of the samples to those of the true distributions; this enables the UKF to capture the effects of higher order moments, not usually exploited by the EKF. The idea is that the posterior is well approximated by a Gaussian distribution, even though some of the models used and so intermediate distributions may be less well approximated as Gaussian. This filter is equivalent to a higher order EKF[118] and its algorithmic simplicity makes it more attractive than the higher order EKF.

The UKF algorithm begins by deterministically drawing samples:

$$\begin{bmatrix} x^i_{t-1} \\ \epsilon^i_{t-1} \\ v^i_t \end{bmatrix} \sim N\left(\begin{bmatrix} x_{t-1} \\ \epsilon_{t-1} \\ v_t \end{bmatrix}; \begin{bmatrix} m_{t-1|t-1} \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} P_{t-1|t-1} & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & R \end{bmatrix} \right), \tag{2.147}$$

with associated weights w^i. The sampling is chosen according to an appropriate deterministic procedure [86], which typically consists of having points at the corners of a simplex[58], the centres of the faces of a hypercube[52] or the corners of a hypercube[53]. The focus of work on selecting the sigma points has been on analysis of the moments of the original and sample-based distributions. Other approaches that consider the spherical symmetry of the samples are possible and do not appear to have been developed.

It's convenient to define that the dynamics falls into one of three classes:

$$x_t = \begin{cases} Fx_{t-1} + \omega^x_{t-1} & \text{Linear Gaussian} \\ f_1(x_{t-1}) + \omega^x_{t-1} & \text{Non-Linear Gaussian} \\ f_2(x_{t-1}, \omega^x_{t-1}) & \text{Non-Linear Non-Gaussian} \end{cases} \tag{2.148}$$

and also to consider three kinds of measurement model describing the relationship between y_t and x_t:

$$y_k = \begin{cases} Hx_k + v_k & \text{Linear Gaussian} \\ h_1(x_k) + v_k & \text{Non-Linear Gaussian} \\ h_2(x_k, v_k) & \text{Non-Linear Non-Gaussian} \end{cases} \tag{2.149}$$

The update stage of the UKF is the same as the Kalman filter equations, (2.55) and (2.56), where now:

$$x_{t|t-1} \approx \sum_{i=1}^{N_s} w^i f_2(x^i_{t-1}, \epsilon^i_{t-1}) \tag{2.150}$$

$$P_{t|t-1} \approx \sum_{i=1}^{N_s} w^i \left[f_2(x^i_{t-1}, \epsilon^i_{t-1}) - x_{t|t-1} \right] \left[f_2(x^i_{t-1}, \epsilon^i_{t-1}) - x_{t|t-1} \right]^T \tag{2.151}$$

$$y_{t|t-1} \approx \sum_{i=1}^{N_s} w^i h_2(f_2(x^i_{t-1}, \epsilon^i_{t-1}), v^i_t) \tag{2.152}$$

$$P_{yy} \approx \sum_{i=1}^{N_s} w^i [h_2(f_2(x^i_{t-1}, \epsilon^i_{t-1}), v^i_t) - y_{t|t-1}][h_2(f_2(x^i_{t-1}, \epsilon^i_{t-1}), v^i_t) - y_{t|t-1}]^T \tag{2.153}$$

$$P_{xy} \approx \sum_{i=1}^{N_s} w^i [f_2(x^i_{t-1}, \epsilon^i_{t-1}) - x_{t|t-1}][h_2(f_2(x^i_{t-1}, \epsilon^i_{t-1}), v^i_t) - y_{t|t-1}]^T, \tag{2.154}$$

While the UKF works well in mildly nonlinear environments, it is worth noting that it is catering for the case of nonlinear non-Gaussian models for both the dynamics and measurement models; it is sampling more than is probably needed. The process of mixing analytic integration and sampling is Rao-Blackwellisation and will be described in more detail in section 2.6.11. Here, it suffices to say that this Rao-Blackwellisation can be carried out and since sampling introduces errors, avoiding unnecessary sampling is advisable.

Indeed a previous method[123] has proposed a method for Rao-Blackwellisation in the presence of additive Gaussian noise. Although using the same concept, the approach presented here differs somewhat in that it only samples once, and so reduces the Monte-Carlo variance.

Table 2.1: *Sampled variables*

	$Fx_{k-1}+\epsilon_{k-1}$	$f(x_{k-1})+\epsilon_{k-1}$	$f(x_{k-1},\epsilon_{k-1})$
Hx_k+v_k	Kalman Filter (2.3)	x_{k-1}	x_{k-1},ϵ_{k-1}
$h(x_k)+v_k$	x_k	x_{k-1},ϵ_{k-1}	x_{k-1},ϵ_{k-1}
$h(x_k,v_k)$	x_k,v_k	$x_{k-1},\epsilon_{k-1},v_k$	UKF (2.5.3)/ Particle Filter (2.6)

Table 2.5.3 displays the different combinations of models that are possible. While the Kalman filter is appropriate if the models are linear and Gaussian and such approaches as particle filtering are advocated when the models are entirely non-linear and non-Gaussian (see section 2.6), other forms of filtering are possible for the other possible combinations.

The following subsections define the equations used in the prediction and update stages in the Kalman filter framework, thereby replacing equations (2.67) to (2.71). Each section is labelled with the definition of the system being considered for easy reference.

Algorithm 1: $x_k = Fx_{k-1}+\epsilon_{k-1}, \quad y_k = h_1(x_k)+v_k$

μ_x and P_{xx} can be calculated exactly using (2.67) and (2.68). Then:

$$x^i \sim N\left(x_k;\ \mu_x,\ P_{xx}\right) \tag{2.155}$$

$$\mu_y \approx \sum_{i=1}^{N_s} w^i h_1(x_k^i) \tag{2.156}$$

$$P_{yy} \approx \sum_{i=1}^{N_s} w^i [h_1(x_k^i)-\mu_y][h_1(x_k^i)-\mu_y]^T + R_k \tag{2.157}$$

$$P_{xy} \approx \sum_{i=1}^{N_s} w^i [x_k^i-\mu_x][h_1(x_k^i)-\mu_y]^T \tag{2.158}$$

Algorithm 2: $x_k = Fx_{k-1}+\epsilon_{k-1}, \quad y_k = h_2(x_k,v_k)$

Again, μ_x and P_{xx} can be calculated exactly using (2.67) and (2.68). One then uses the following scheme:

$$\begin{bmatrix} x_k^i \\ v_k^i \end{bmatrix} \sim N\left(\begin{bmatrix} x_k \\ v_k \end{bmatrix};\ \begin{bmatrix} \mu_x \\ 0 \end{bmatrix},\ \begin{bmatrix} P_{xx} & 0 \\ 0 & R_k \end{bmatrix}\right) \tag{2.159}$$

$$\mu_y \approx \sum_{i=1}^{N_s} w^i h_2(x_k^i, v_k^i) \tag{2.160}$$

$$P_{yy} \approx \sum_{i=1}^{N_s} w^i h_2(x_k^i, v_k^i) - \mu_y][h_2(x_k^i, v_k^i)-\mu_y]^T \tag{2.161}$$

$$P_{xy} \approx \sum_{i=1}^{N_s} w^i [x_k-\mu_x][h_2(x_k^i, v_k^i)-\mu_y]^T \tag{2.162}$$

Algorithm 3: $x_k = f_1(x_{k-1}) + \epsilon_{k-1}, \quad y_k = Hx_k + v_k$

In this case:

$$x_{k-1}^i \sim N\left(x_{k-1};\ m_{k-1|k-1},\ P_{k-1|k-1}\right) \tag{2.163}$$

$$\mu_x \approx \sum_{i=1}^{N_s} w^i f_1(x_{k-1}^i) \tag{2.164}$$

$$P_{xx} \approx \sum_{i=1}^{N_s} w^i [f_1(x_{k-1}^i) - \mu_x][f_1(x_{k-1}^i) - \mu_x]^T + Q_{k-1} \tag{2.165}$$

μ_y, P_{yy} and P_{xy} can then be calculated exactly using (2.69) to (2.71).

Algorithm 4: $x_k = f_1(x_{k-1}) + \epsilon_{k-1}, \quad y_k = h_1(x_k) + v_k$

The advantage in this case is that some of the integrals can be calculated exactly:

$$\begin{bmatrix} x_{k-1}^i \\ \epsilon_{k-1}^i \end{bmatrix} \sim N\left(\begin{bmatrix} x_{k-1} \\ \epsilon_{k-1} \end{bmatrix}; \begin{bmatrix} m_{k-1|k-1} \\ 0 \end{bmatrix}, \begin{bmatrix} P_{k-1|k-1} & 0 \\ 0 & Q_{k-1} \end{bmatrix}\right) \tag{2.166}$$

$$\mu_y \approx \sum_{i=1}^{N_s} w^i h_1 f_1(x_{k-1}^i) + \epsilon_{k-1}^i \tag{2.167}$$

$$P_{yy} \approx \sum_{i=1}^{N_s} w^i [h_1 f_1(x_{k-1}^i) + \epsilon_{k-1}^i - \mu_y][h_1 f_1(x_{k-1}^i) + \epsilon_{k-1}^i - \mu_y]^T + R_k \tag{2.168}$$

$$P_{xy} \approx \sum_{i=1}^{N_s} w^i [f_1(x_{k-1}^i) + \epsilon_{k-1}^i - \mu_x][h_1 f_1(x_{k-1}^i) + \epsilon_{k-1}^i - \mu_y]^T \tag{2.169}$$

μ_x and P_{xx} are calculated approximately as in (2.164) and (2.165).

Algorithm 5: $x_k = f_2(x_{k-1}, \epsilon_{k-1}), \quad y_k = Hx_k + v_k$

Here the samples are drawn as in (2.166) and then the (normal UKF) approximations in (2.150) and (2.151) are used to calculate μ_x and P_{xx}. μ_y, P_{yy} and P_{xy} can then be calculated exactly using (2.69) to (2.71).

Algorithm 6: $x_k = f_2(x_{k-1}, \epsilon_{k-1}), \quad y_k = h_1(x_k) + v_k$

Again the samples are drawn as in (2.166); μ_x and P_{xx} are calculated approximately using (2.150) and (2.151). The remaining parameters are calculated as follows:

$$\mu_y \approx \sum_{i=1}^{N_s} w^i h_1 f_2(x_{k-1}^i, \epsilon_{k-1}^i) \tag{2.170}$$

$$P_{yy} \approx \sum_{i=1}^{N_s} w^i [h_1 f_2(x_{k-1}^i, \epsilon_{k-1}^i) - \mu_y][h_1 f_2(x_{k-1}^i, \epsilon_{k-1}^i) - \mu_y]^T + R_k \tag{2.171}$$

$$P_{xy} \approx \sum_{i=1}^{N_s} w^i [f_2(x_{k-1}^i, \epsilon_{k-1}^i) - \mu_x][h_1 f_2(x_{k-1}^i, \epsilon_{k-1}^i) - \mu_y]^T \tag{2.172}$$

Algorithm 7: $x_k = f_1(x_{k-1}) + \epsilon_{k-1}, \quad y_k = h_2(x_k, v_k)$

Samples are drawn as in (2.147). μ_x and P_{xx} are calculated approximately as in (2.164) and (2.165). The remaining relations are then:

$$\mu_y \approx \sum_{i=1}^{N_s} w^i h_2(f_1(x_{k-1}^i) + \epsilon_{k-1}^i, v_k^i) \tag{2.173}$$

$$P_{yy} \approx \sum_{i=1}^{N_s} w^i [h_2(f_1(x_{k-1}^i) + \epsilon_{k-1}^i, v_k^i) - \mu_y][h_2(f_1(x_{k-1}^i) + \epsilon_{k-1}^i, v_k^i) - \mu_y]^T \tag{2.174}$$

$$P_{xy} \approx \sum_{i=1}^{N_s} w^i [f_1(x_{k-1}^i) + \epsilon_{k-1}^i - \mu_x][h_2(f_1(x_{k-1}^i) + \epsilon_{k-1}^i, v_k^i) - \mu_y]^T \tag{2.175}$$

Comment

It is worth considering where these Rao-Blackwellised Unscented Kalman filter algorithms fit into the wide range of filters readily available in the tracking literature. Clearly they are closely related to the Kalman filter and the traditional Unscented Kalman filter. The UKF has been likened to the Linear Regression Kalman filter [61] and the author believes that the non-linear Gaussian Rao-Blackwellised UKF algorithms are indeed special cases of the Linear Regression Kalman filter. Of course, any of the various methods of choosing sigma points can be applied to the RB-UKF in the same way as they would be applied to the UKF.

2.5.4 Iterated EKFs

The argument used to explain why the EKF seemed to diverge in some of the exemplar cases also helps explain why the Iterated Extended Kalman Filter[50], IEKF, often works well. The EKF (not IEKF) linearises once on the basis of the predicted estimate of the state, $x_{t|t-1}$. The uncertainty associated with this estimate is likely to be significantly larger than that associated with filtered estimate, $x_{t|t}$. So, by relinearising the measurement model using $x_{t|t}$, the parameters of the approximation to the models can be fine tuned. This can improve the accuracy of the parameters of the Gaussian approximation to $p(x_t|y_{1:t})$. Some further gains can be obtained by making this process iterate and using the estimates of $x_{t|t}$ from one iteration to relinearise the measurement model at the next iteration. One can also relinearise the dynamics using smoothed estimates of $x_{t-1|t}$ and then relinearise the measurement model using the prediction based on this revised linear approximation to the dynamics.

In the extreme case, one can relinearise at all points in a time series by using a smoothed estimate of $x_{0|t}$ to relinearise the initial dynamic model and then relinearising at all subsequent points using the estimates that result; if one changes the initial estimate (rather than only changing the approximation to the models), this is the same as using Expectation-Maximisation to learn the initial state of a system[111]. However, this is essentially changing the prior to reflect the observations, which seems intuitively unappealing. An alternative is to consider the (I)EKF to use an approximation to the models based on a Taylor series and to consider the iterative version of the filter to be altering the state about which the Taylor series is conducted. Under this scheme, in the first iteration of the IEKF, the models are ap-

proximated using on Taylor series about $x_{t-1|t-1}$ for the dynamic model and $x_{t|t-1}$ for the measurement model. In subsequent iterations, the models are then approximated using the values of $x_{t-1|L}$ and $x_{t|L}$ from the previous iteration. The prior remains constant.

It is worth stressing that in this nonlinear non-Gaussian environment, learning the initial state or equally re-approximating all the models in this way can affect the final state estimate; the final state estimate is a function of all the linearisations up to the final time so changing these linear approximations will change the final state estimate.

2.5.5 Approximate Gaussian Mixtures

It is possible to approximate a mixture of Gaussians with a single Gaussian by matching the mean and covariance of the mixture to that of the approximating Gaussian:

$$p(x) = \sum_{i=1}^{N} w^i \mathcal{N}(x; m_i, C_i) \tag{2.176}$$

$$\approx \mathcal{N}(x; m, C) \tag{2.177}$$

where

$$m = \sum_{i=1}^{N} w_i m_i \tag{2.178}$$

$$C = \sum_{i=1}^{N} w_i \left(C_i + (m_i - m)(m_i - m)^T \right). \tag{2.179}$$

Algorithms that use this approach are very popular indeed and will be discussed in detail in section 2.7. An alternative is to pick the best (or n-best) component(s) in terms of the components weights. This approach is also widespread and will also be discussed in section 2.7. While other schemes do exist[105] for approximating mixtures of Gaussians with mixtures with fewer components, their use is not as widespread. This is perhaps because of their computational expense though it is surprising that these algorithms are not used more often than they are.

2.6 Particle Filtering

As an alternative to approximating the models, it is possible to approximate $p(x_{1:t}|y_{1:t})$ directly. This is the approach adopted by the particle filter and is likely to be beneficial when the nonlinearity (as described above) and non-Gaussianity is pronounced.

The impact is that whereas one can improve Kalman Filter performance by using nonlinear processing of consecutive measurements[11, 12, 109], by using the particle filter, one automatically carries out this nonlinear path based processing. A number of articles and books do exist on the subject of particle filtering and the interested reader should refer to these texts[2, 28, 42, 56].

Like the Kalman Filter, the particle filter is a sequential algorithm. However, a better understanding of using particle filtering in a sequential setting is possible by considering the history of states; the

particle filter is actually an approach to inferring the path through the state-space over time which can be implemented by only storing quantities relating to the filtered time. So, to make it easy to see how the particle filter operates, the algorithm is described here from a path-based starting point. For completeness, an interpretation free from reference to the history of the states and based on Monte-Carlo integration is given in section 2.6.3.

The particle filter does not attempt to represent the distribution using an analytic form. Instead, the uncertainty (and so the distribution) is represented using the diversity of a set of N samples or particles. The ith of these particles consists of a hypothesised history of states of the target, $x^i_{1:t}$, and an associated weight, $\hat{w}^i_{1:t}$. Ideally one would want these samples to be samples from $p(x_{1:t}|y_{1:t})$. If this were the case, then:

$$x^i_{1:t} \sim p(x_{1:t}|y_{1:t}) \tag{2.180}$$

$$\hat{w}^i_{1:t} = \frac{1}{N}. \tag{2.181}$$

The particles do represent the pdf and it is possible to think of the particles as samples from the pdf which can be used to estimate expectations with respect to the pdf. However, a minor point to note is that the particles do not really approximate the true pdf at any point x since the particle distribution is zero everywhere except under (the support of) the samples, where the delta function makes the distribution infinite:

$$p(x_{1:t}|y_{1:t}) \not\approx \sum_{i=1}^{N} \hat{w}^i_{1:t} \delta\left(x^i_{1:t} - x_{1:t}\right). \tag{2.182}$$

What makes these samples useful is the fact that they can be used to statistically estimate the expected value of functions of the state:

$$\int f(x_{1:t})\, p(x_{1:t}|y_{1:t})\, dx_{1:t} \approx \int f(x_{1:t}) \sum_{i=1}^{N} \hat{w}^i_{1:t} \delta\left(x^i_{1:t} - x_{1:t}\right) dx_{1:t} \tag{2.183}$$

$$= \sum_{i=1}^{N} \hat{w}^i_{1:t} f\left(x^i_{1:t}\right). \tag{2.184}$$

So, if one had an algorithm which could (somehow) sample N times from $p(x_{1:t}|y_{1:t})$, one would obtain $x^i_{1:t}$ for $i = 1 \ldots N$. Then one could estimate quantities of interest such as $x_{t|t}$ and $P_{t|t}$, by just calculating the quantity from the weighted sample set using (2.184).

Unfortunately, it is unlikely that it will be possible to sample exactly from $p(x_{1:t}|y_{1:t})$; if one could sample exactly, it is probable that a particle filter would be unnecessary since $p(x_{1:t}|y_{1:t})$ would be likely to have an analytic form!

2.6.1 Importance Sampling

Instead of simulating from $p(x_{1:t}|y_{1:t})$ directly one can choose a convenient *proposal distribution* or *importance function*, $q(x_{1:t}|y_{1:t})$ which is easy to sample from and evaluate. Samples are drawn from this

proposal and the weights are modified using the principle of importance sampling. Where the proposal distribution proposes samples at which $q\left(x_{1:t}|y_{1:t}\right) > p\left(x_{1:t}|y_{1:t}\right)$ there are more samples than there should be. Hence these samples are given a reduced weight. Conversely, when samples are drawn in regions where $q\left(x_{1:t}|y_{1:t}\right) < p\left(x_{1:t}|y_{1:t}\right)$, there are too few samples and the samples are given an increased weight to counter this effect. This can be formulated by looking at how to calculate expectations with respect to one distribution given samples from another distribution:

$$\int p(x_{1:t}|y_{1:t})f(x_{1:t})dx_{1:t} = \int \frac{p(x_{1:t}|y_{1:t})}{q(x_{1:t}|y_{1:t})} f(x_{1:t})q(x_{1:t}|y_{1:t})dx_{1:t} \tag{2.185}$$

$$\approx \int \frac{p(x_{1:t}|y_{1:t})}{q(x_{1:t}|y_{1:t})} f(x_{1:t}) \frac{1}{N}\sum_{i=1}^{N} \delta\left(x_{1:t} - x_{1:t}^i\right) dx_{1:t} \tag{2.186}$$

$$= \frac{1}{N}\sum_{i=1}^{N} \hat{w}_{1:t}^i f\left(x_{1:t}^i\right) \tag{2.187}$$

where

$$x_{1:t}^i \sim q(x_{1:t}|y_{1:t}) \tag{2.188}$$

$$\hat{w}_{1:t}^i = \frac{p(x_{1:t}^i|y_{1:t})}{q(x_{1:t}^i|y_{1:t})}. \tag{2.189}$$

So, the idea at the heart of the particle filter is to pick a convenient approximating proposal to sample particles from:

$$x_{1:t}^i \sim q(x_{1:t}|y_{1:t}) \approx p(x_{1:t}|y_{1:t}) = \frac{p(x_{1:t}, y_{1:t})}{p(y_{1:t})}. \tag{2.190}$$

The particle filter then defines an *un-normalised weight*:

$$\tilde{w}_{1:t}^i = \frac{p(x_{1:t}^i, y_{1:t})}{q(x_{1:t}^i|y_{1:t})}. \tag{2.191}$$

It is then possible to calculate an approximation to $p(y_{1:t})$,

$$p(y_{1:t}) = \int p\left(x_{1:t}, y_{1:t}\right) dx_{1:t} \approx \sum_{i=1}^{N} \tilde{w}_{1:t}^i, \tag{2.192}$$

and so the *normalised weight* is

$$w_{1:t}^i = \frac{p(x_{1:t}^i|y_{1:t})}{q(x_{1:t}^i|y_{1:t})} = \frac{p(x_{1:t}^i, y_{1:t})}{q(x_{1:t}^i|y_{1:t})} \frac{1}{p(y_{1:t})} \tag{2.193}$$

$$= \frac{\tilde{w}_{1:t}^i}{\sum_{i=1}^{N} \tilde{w}_{1:t}^i}. \tag{2.194}$$

These normalised weights can then be used to estimate quantities of interest:

$$\int p(x_{1:t}|y_{1:t})f(x_{1:t})dx_{1:t} \approx \sum_{i=1}^{N} w_{1:t}^i f\left(x_{1:t}^i\right) \tag{2.195}$$

Such an algorithm would sample N times from $q\left(x_{1:t}|y_{1:t}\right)$ to get $x_{1:t}^i$ for $i = 1 \ldots N$. Each sample would have a weight $w_{1:t}^i$ as defined in (2.194). To calculate quantities of interest such as $x_{t|t}$ and $P_{t|t}$, one would then use (2.195). The author believes this is similar to the approach in [33, 108], which use an (Extended) Kalman smoother to obtain $q\left(x_{1:t}|y_{1:t}\right)$.

Note that while both the denominator and numerator of (2.194) are unbiased estimates, if the normalised weight is used, since it is a ratio of estimates, the filter is biased[9].

It is also worth noting that, for this to work, the importance function needs to have heavier tails than the true posterior. In an appeal to intuition that explains why this might be the case, consider the extreme (bad) case when the importance function is a delta function; in such cases, drawing more and more samples from the importance function won't improve any estimation based on the samples.

A more rigorous explanation is to consider a scaled version of the importance function. If the posterior is lighter tailed than the importance function then it will be possible to choose a non-infinite scale such that the scaled version of the importance function is larger in magnitude than the posterior for all values of the state. If this is the case, then populating the importance function's probability mass with samples will guarantee that the probability mass of the posterior is populated with samples. Evidently, the further the scale is from unity, the less efficient the importance function. The bottom-line is that one needs to be confident that the importance function is heavier tailed than the posterior.

2.6.2 Sequential Importance Sampling

As described to this point, the particle filter proposes particles (consisting of the entire history of states) at each point in time with no reference to the previously sampled particles. This is clearly undesirable and not sequential. Indeed, to perform *sequential importance sampling*, SIS, the proposal is defined to be of the following particular form[10]:

$$q(x_{1:t}|y_{1:t}) \triangleq \underbrace{q(x_{1:t-1}|y_{1:t-1})}_{\text{Keep existing path}} \underbrace{q(x_t|x_{t-1}, y_{1:t})}_{\text{Extend path}} \tag{2.196}$$

which means that the particles at one iteration are generated by extending the existing history of the particle with a sample for the state at the current time using the information from the history (summarised by x^i_{t-1}), the measurement, y_t and potentially, the history of previous measurements, $y_{1:t-1}$. Hence, if $x_{1:t-1}$ is not of interest, one only need to store x^i_t and $w^i_{1:t}$ for each particle at each iteration; the particle filter is indeed a sequential algorithm. The (incremental[11]) un-normalised weight then takes on an intuitively appealing form:

$$\tilde{w}^i_{1:t} = \frac{p(x^i_{1:t}, y_t|y_{1:t-1})}{q(x^i_{1:t}|y_{1:t})} = \frac{p(x^i_{1:t-1}|y_{1:t-1})}{q(x^i_{1:t-1}|y_{1:t-1})} \frac{p(x^i_t, y_t|x^i_{t-1})}{q(x^i_t|x^i_{t-1}, y_{1:t})} = w^i_{1:t-1} \underbrace{\frac{p(y_t|x^i_t)p(x^i_t|x^i_{t-1})}{q(x^i_t|x^i_{t-1}, y_{1:t})}}_{\text{Incremental Weight}}. \tag{2.197}$$

One can then approximate $p(y_t|y_{1:t-1})$ by:

$$p(y_t|y_{1:t-1}) = \int p(x_{1:t}, y_t|y_{1:t-1}) dx_{1:t} \approx \frac{1}{N} \sum_{i=1}^{N} \tilde{w}^i_{1:t}, \tag{2.198}$$

[9] The formulations in [33, 108] are claimed to give rise to unbiased estimates, but a sequential unbiased particle filter certainly appears to have eluded researchers up to the current time.

[10] This is slightly different to the usual formulation, for which $q(x_{1:t}|y_{1:t}) \triangleq q(x_{1:t-1}|y_{1:t-1})q(x_t|x_{t-1}, y_t)$, so as to accommodate all the proposals described in section 2.6.9.

[11] The un-normalised weight previously defined in 2.191 is defined as the ratio of $p(x^i_{1:t}, y_{1:t})$ to $q(x^i_{1:t}|y_{1:t})$, whereas here the incremental un-normalised weight is defined as the ratio of $p(x^i_{1:t}, y_t|y_{1:t-1})$ to $q(x^i_{1:t}|y_{1:t})$.

which is an approximation[12] that can be used to estimate the likelihood of the data (which will prove useful in chapter 4 in the context of model selection) by substituting into:

$$p\left(y_{1:t}\right) = \prod_{t'=1}^{t} p\left(y_{t'}|y_{1:t'-1}\right). \tag{2.199}$$

Note that a peculiar case arises when $t = 1$. In this case, $p\left(y_1|y_{1:0}\right) = p\left(y_1\right)$ can be obtained from (2.198) using the first set of samples for which $i = 1$. Note that the definition, $y_0 = \emptyset$, is used simply for notational convenience. The normalised weights can then be calculated:

$$w^i_{1:t} = \frac{p(x^i_{1:t}|y_{1:t})}{q(x^i_{1:t}|y_{1:t})} = \frac{p(x^i_{1:t}, y_t|y_{1:t-1})}{q(x^i_{1:t}|y_{1:t})} \frac{1}{p(y_t|y_{1:t-1})} \tag{2.200}$$

$$= \frac{\tilde{w}^i_{1:t}}{\sum_{i=1}^{N} \tilde{w}^i_{1:t}}. \tag{2.201}$$

So, to begin the algorithm samples N times from the prior, $p\left(x_0\right)$ and assigns each particle a weight of $\frac{1}{N}$. Then, at each iteration, an algorithm would now sample N times from $q\left(x_t|x_{t-1}, y_{1:t}\right)$ to get x^i_t for $i = 1 \ldots N$. Each sample would have a weight $w^i_{1:t}$ as defined in (2.201). To calculate quantities of interest such as $x_{t|t}$ and $P_{t|t}$, one would then use (2.195) as previously.

However, there is still a problem; whatever the choice of $q\left(x_t|x_{t-1}, y_{1:t}\right)$, eventually one of the particles will have a weight close to unity while all the others will have a weight close to zero. This essentially happens because, at every iteration, the weights can only diverge. To see why this might be problematic, consider what would happen to (2.195) in such a scenario; if the weights depart from uniform and so some of the weights becomes much larger than all the others, then these particles dominate any estimation. The other particles are then essentially wasted.

Another way to think about this is that the effective number of particles is reduced. This effective sample size can be approximated using the following statistic of the normalised weights:

$$N_{eff} \approx \frac{1}{\left(\sum_{i=1}^{N} {w^i_{1:t}}^2\right)} \tag{2.202}$$

To appeal to intuition, consider two cases. When all the $w^i_{1:t} = \frac{1}{N}$ for all i then $N_{eff} = N$ while when $w^i_{1:t} = 1$ for one value of i and $w^i_{1:t} = 0$ for all the others then $N_{eff} = 1$. So, the approximate effective sample size number is N when the weights are uniform and 1 in the extreme case of one of the particles having all the weight. So, by thresholding this effective sample size (perhaps with a threshold of $0.5N$), it is possible to spot when this problematic degeneracy is evident.

2.6.3 Monte-Carlo Integration Interpretation

An alternative interpretation of the particle filter is to consider the filter as using Monte-Carlo integration rather than importance sampling of the path; such a description is (of course) equivalent though it makes it more difficult to explain some of the more involved concepts; specifically the reasoning behind the use of different proposal distributions and for the normalisation of the particles weights. Hence, this section is simply present to provide a description of this alternative interpretation and should therefore

[12] Other approximations are proposed in [29] for example.

be considered in isolation from the subsequent sections that will assume an interpretation of the particle filter based on importance sampling of the path.

Under this Monte-Carlo interpretation, at iteration t of the filter, the particle filter stores N samples of the state, x_t^i, and an associated weight, w_t^i, as an approximation to $p\left(x_t|y_{1:t}\right)$. To obtain an approximation to $p\left(x_t|y_{1:t}\right)$ from the sample based approximation to $p\left(x_{t-1}|y_{1:t-1}\right)$, Monte-Carlo integration based on a proposal distribution $q\left(x_t|x_{t-1}, y_{1:t}\right)$, is used:

$$p\left(x_t|y_{1:t}\right) \propto p\left(y_t|x_t\right) \int p\left(x_t|x_{t-1}\right) p\left(x_{t-1}|y_{1:t-1}\right) dx_t \tag{2.203}$$

$$\approx p\left(y_t|x_t\right) \int p\left(x_t|x_{t-1}\right) \sum_i w_{t-1}^i \delta\left(x_{t-1} - x_{t-1}^i\right) dx_t \tag{2.204}$$

$$\approx \sum_i w_t^i \delta\left(x_t - x_t^i\right) \tag{2.205}$$

where:

$$w_t^i \propto w_{t-1}^i \frac{p\left(y_t|x_t^i\right) \sum_j p\left(x_t^i|x_{t-1}^j\right)}{q\left(x_t^i|x_{t-1}^i, y_{1:t}\right)} \tag{2.206}$$

where the weights are normalised to sum to unity (such that the samples represent a probability distribution). Note that the calculation of the weights for every particle involves a sum over all the particles, making the computational cost scale quadratically with the number of particles, unless an approximation scheme such as the following is used:

$$w_t^i \propto w_{t-1}^i \frac{p\left(y_t|x_t^i\right) p\left(x_t^i|x_{t-1}^i\right)}{q\left(x_t^i|x_{t-1}^i, y_{1:t}\right)}. \tag{2.207}$$

2.6.4 Resampling

The reason that this comes about is because the same number of samples is being used to conduct importance sampling on a pdf (of a history of states) that is increasing in dimensionality at every iteration. Eventually the samples will be insufficient in number. This is what is causing the aforementioned degeneracy phenomenon. In that it is the sequential setting that is of interest, a way to circumnavigate this problem is to intentionally move the degeneracy to somewhere that it doesn't matter; the path through the history of states.

So, to alleviate the effect of this degeneracy, a resampling stage is introduced. This key step was introduced in [42] and led to the first operational particle filter (approaches prior to this development had been unable to circumvent the explosion over time in the number of particles needed [45]). The resampling stage results in particles being replicated and discarded. The particles are redistributed so as to spread the particles (and so the computation) evenly across the posterior. This redistribution introduces errors but makes the weights more uniform. After the resampling stage, $w_{1:t}^i = \frac{1}{N}$ for all the particles. Particles that are replicated then share the same path and so the path pdf degenerates, but the filtered pdf is rejuvenated. To avoid any confusion, it should be stressed that the decision as to whether to resample is based on (2.202) being calculated at each iteration; hence since the statistic in (2.202) will typically fall below the aforementioned threshold several times, the resampling will typically take place a number of times.

There are a number of different resampling algorithms. The simplest to understand is *multinomial resampling*[42]. Each particle in the new particle set is a copy of the ith particle in the old particle set with probability w_t^i. So, the new generation is generated by sampling N times from the distribution formed by the weights on the old set of particles.

To understand the differences between this approach and other resampling algorithms, the new generation is defined to consist of N_i replicates of the ith particle in the old particle set. Different existing resampling algorithms are then the same with respect to $\mathbb{E}[N_i]$, the expected number of replicates, but differ with respect to $\mathbb{E}\left[(N_i - \mathbb{E}[N_i])^2\right]$, the variance in the number of replicates. This variance can be thought of as the size of an error; it is advantageous to introduce as few errors as possible and so sensible to use an algorithm that minimises this variance. While there are some candidate algorithms which don't guarantee constant N, these aren't discussed here.

Multinomial Resampling

As previously stated, multinomial resampling is the simplest resampling algorithm. The new particle set is generated by drawing N independent samples from the weighted old particle set. This approach can be implemented in a time that scales linearly with the number of samples[21, 103]. The deficiency of this approach is that there is then a finite probability of any particle being replicated any number of times. This results in a large variance in the number of replicates.

Residual Resampling

Residual resampling considers the expected number of replications of each particle and deterministically replicated some of the particles[65]. Specifically, the ith particle is replicated $(\mathbb{E}[N_i])^-$ times, where A^- denotes the integer less than or equal to A. The remaining $N - \sum_{j=1}^{N}(\mathbb{E}[N_j])^-$ particles are then sampled according to the aforementioned multinomial scheme. This reduces $\mathbb{E}\left[(N_i - \mathbb{E}[N_i])^2\right]$ since the particles cannot be replicated less than $(\mathbb{E}[N_i])^-$ times.

Systematic Resampling

Systematic resampling[56] has been the approach to be advocated up to this point. The new particle set is chosen so as to minimise $\mathbb{E}\left[(N_i - \mathbb{E}[N_i])^2\right]$; every particle is replicated either $(\mathbb{E}[N_i])^-$ or $(\mathbb{E}[N_i])^+$ times, where A^+ is the smallest integer greater than or equal to A. The algorithm draws a single sample and then has behaviour that is deterministic once this sample has been drawn.

If the state space is discrete, rather than continuous, then a similar approach can be used[36]. In this discrete environment, rather than resampling such that the particles are redistributed, the approach taken is to consider the possible evolutions and then select a subset; N samples are drawn from a set of $M > N$ samples. To reduce the variance, the scheme ensures that each sample is either selected once or not at all; no two of the N samples in the new population are then the same. It transpires that a minimum variance unbiased approach can be devised by calculating a threshold and then adding a sample to the new population for every sample in the old population with a weight above this threshold. The weight for these deterministically selected samples is then the corresponding sample in the old population's old

weight. For samples in the old population which are not deterministically selected, the probability of selecting the sample is taken to be proportional to the weight in the old generation and the weight that this sample then gets in the new population is the value of this aforementioned threshold. As a result, the expected value of the weight in the new population is that of the sample in the old generation.

Minimum Error Continuous State Resampling

The constraint that the resampling step is unbiased makes it impossible to deterministically reject any samples. However, by introducing a small bias, it is possible to reduce the variance sufficiently that the error is minimised.

If the new set of samples is selected deterministically, the resampling will be biased; the reason for having stochasticity in the resampling algorithm is so that, on average (over the potential new sets of samples), the resampling is unbiased. Without the constraint that the resampling step is unbiased, the task is simply to minimise a cost function.

The notation used here is chosen to aid comparison with [36]. The ith of M samples has a weight, q_i, associated with it. The sample set to be approximated consists of M samples with a sample set consisting of $N \leq M$ samples. A variable X_i is defined which is the weight in the new sample set associated with the ith sample in the old sample set. X_i takes a value, C_i, with a probability p_i and another value, $C_i^\star$, otherwise:

$$X_i = \begin{cases} C_i & \text{with probability, } p_i \\ C_i^\star & \text{otherwise} \end{cases}. \tag{2.208}$$

In [36], the minimisation of the variance of an unbiased estimator of a function of the state based on the resampled particles is used as motivation for the choice of resampling scheme. This was shown to motivate a resampling scheme that was unbiased and that minimised the variance of the original particles' weights summed over the particles. Here, we remove the unbiased condition and simply minimise the summed error between the original particles' weights and the total weight of the corresponding resampled particles. The idea is that this will reduce the error in the estimation of expectations based on the resampled particles. So, the cost function to minimise, in the design of the resampling operation, is the error, the mean squared distance between the random vector $X = \{X_1, \ldots, X_M\}$ and the vector of weights $q = \{q_1, \ldots, q_M\}$ and defined to be:

$$y\left(p_1, \ldots, p_M\right) \triangleq \mathbb{E}\left[\sum_{i=1}^{M} \left(X_i - q_i\right)^2\right]. \tag{2.209}$$

Considering the error as an expectation with respect to the distribution of the new set of samples, one can see that this expected cost is upper bounded by the cost resulting from all the probability mass being on the best new set of samples:

$$y\left(p_1, \ldots, p_M\right) = \sum_X P\left(X\right) y\left(X\right) \leq \max_X \left\{y\left(X\right)\right\} \tag{2.210}$$

where X is a vector comprising all the X_i and $y\left(X\right)$ is the associated cost function. $P\left(X\right)$ is the probability of the resampling algorithm resulting in X (and so all the X_i's) taking a particular value.

So, this implies that one should deterministically pick the new set of samples so that it minimises this cost; all the p_i's are either zero or unity. In the case of continuous states (for which $M = N$), this is achieved by assuming that all the samples have a weight of $\frac{1}{N}$, so $C_i \in \frac{1}{N}\left\{[Nq_i]^-, [Nq_i]^+\right\}$. The cost will be minimised by deterministically selecting $[Nq_i]^-$, replicants of the ith sample (which can be carried out by considering each sample in turn in $O(N)$ time) and selecting the $N - \sum_{i=1}^{N}[Nq_i]^-$ largest non-integer parts. This does makes it necessary to sort (at least partially) the list of the largest non-integer parts, but this can be done rapidly (in $O(N \log N)$ time).

It is worth noting that since the minimum variance unbiased resampling algorithms only have a very limited amount of stochastic behaviour, the performance of this minimum error approach is in fact very similar to that of the minimum variance unbiased algorithms.

Minimum Error Discrete State Resampling

When considering discrete states, resampling is conducting by selecting N of the $M > N$ available samples of the history of the discrete state[13]. By a similar argument to that used previously, the minimum error resampling scheme is simply that which deterministically selects the N hypotheses to *keep* and the corresponding C_i (and does this so as to minimise the error).

Again, one can choose to minimise the total error between the original vector of weights and the resampled weights as a means to minimising the error of an estimator based on the resampled hypotheses. So, consider first the minimum error achievable given the choice of N hypotheses of M hypotheses that are selected. The cost is then minimised through the choice of C_i for the N resampled samples ($C_i^\star = 0$ for all the other samples). The additional constraint that the C_i sum to unity across these N samples is then enforced by minimising a cost function with a lagrangian term penalising violation of this constraint:

$$C = \mathbb{E}\left[\sum_{i=1}^{M}(X_i - q_i)^2\right] + \underbrace{\lambda\left(\sum_{i=1}^{N}C_i - 1\right)}_{\text{Penalising term}} \tag{2.211}$$

$$= \sum_{i=1}^{N}(C_i - q_i)^2 + \sum_{i=N+1}^{M}{q_i}^2 + \lambda\left(\sum_{i=1}^{N}C_i - 1\right) \tag{2.212}$$

where the sign of λ is chosen such that the *Penalising term* is non-negative and where it is assumed that the samples are permuted such that the first N samples are those that are resampled.

Taking derivatives with respect to C_i:

$$\frac{dC}{dC_i} = 2(C_i - q_i) + \lambda \tag{2.213}$$

Then setting this derivative to zero implies:

$$C_i = q_i - \frac{\lambda}{2} \tag{2.214}$$

The implication is that each of the C_i takes the value of the corresponding value of q_i, plus some offset that is constant across the samples; the remaining probability mass is distributed equally across

[13]This assumes that, at each time epoch, it is feasible to enumerate the candidate extensions of the history of the discrete state; the state isn't practically continuous.

the resampled samples:

$$C_i = q_i + \frac{1 - \sum_{j=1}^{N} q_j}{N} \tag{2.215}$$

The corresponding cost is then:

$$C = \sum_{i=1}^{N} \left(\frac{1 - \sum_{j=1}^{N} q_j}{N} \right)^2 + \sum_{i=N+1}^{M} {q_i}^2 \tag{2.216}$$

$$= \sum_{i=1}^{N} \left(\frac{\sum_{j=N+1}^{M} q_j}{N} \right)^2 + \sum_{i=N+1}^{M} {q_i}^2 \tag{2.217}$$

$$= \frac{1}{N} \left(\sum_{i=N+1}^{M} q_i \right)^2 + \sum_{i=N+1}^{M} {q_i}^2 \tag{2.218}$$

Now, consider the change in cost that results from changing the set of samples that are resampled to include q_i in place of $q_{i'}$:

$$\Delta C = \frac{1}{N} \left(\bar{q} + q_{i'} \right)^2 + q_{i'}^2 - q_i - \frac{1}{N} \left(\bar{q} + q_i \right)^2 \tag{2.219}$$

where $\bar{q}$ is the total weight of all the other unused samples other than the sample that is either q_i or $q_{i'}$.

If $q_{i'} = q_i + \delta$ then:

$$\Delta C = \frac{1}{N} \left(\bar{q} + q_i + \delta \right)^2 + \left(q_i + \delta \right)^2 - q_i - \frac{1}{N} \left(\bar{q} + q_i \right)^2 \tag{2.220}$$

$$= \frac{1}{N} \left(2 \left(\bar{q} + q_i \right) \delta + \delta^2 \right) + 2 q_i \delta + \delta^2 \tag{2.221}$$

So, since $\bar{q} > 0$ and $q_i > 0$, if $\delta > 0$ (and so $q_{i'} > q_i$), then this change in cost, ΔC, is positive. Hence, the cost can always be reduced by removing from the set of resampled samples those samples with lower values of q_i and replacing them with samples with higher values of q_i. This process can be applied until the set of resampled samples is that with the highest weights; the minimum error discrete state resampling scheme is then to select the samples with the highest weights and redistribute the weights of the other samples evenly across the selected samples. Algorithms that select the best hypotheses in this way have existed since the 1970s under the name of Multiple Hypothesis Trackers[14, 102]! However, the even redistribution of the other samples weights makes this approach different[14].

Example

So, the above provides all the ingredients with which to apply a particle filter to the example. The dynamics are used as the proposal distribution:

$$q \left(x_t | x_{t-1}, y_{1:t} \right) \triangleq p \left(x_t | x_{t-1} \right). \tag{2.222}$$

We choose $N = 250$ particles and initialise the filter with N samples from the initial Gaussian distribution used for the EKF given in (2.144). We resample when N_{eff} falls below 125. We consider the second (difficult) case. We run the filter nine times as before. The particles' positions are shown with the true position in figure 2.3.

[14] Alternative forms of cost function could result in exactly these algorithms.

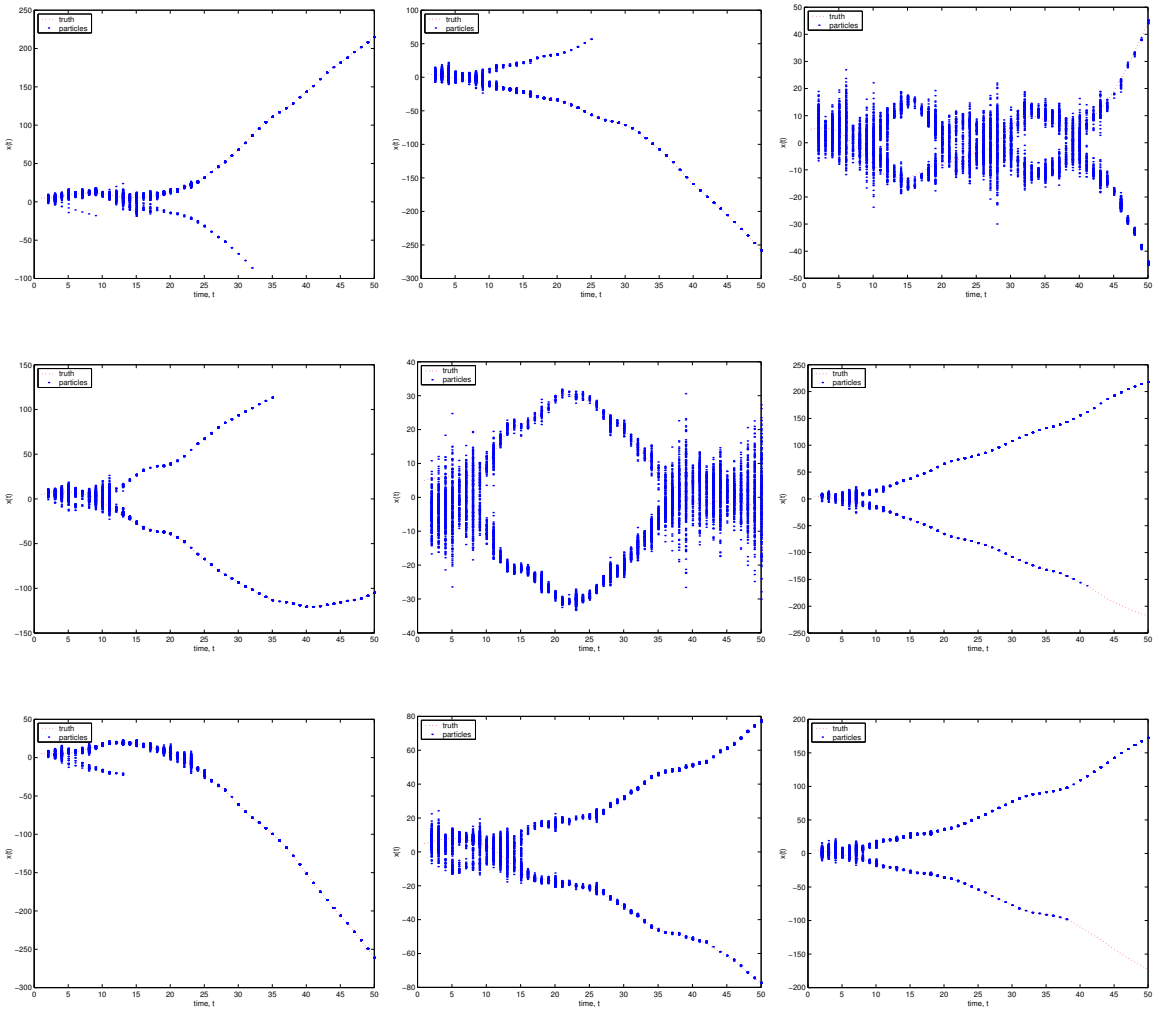


Figure 2.3: *Nine runs of the particle filter for the model defined by (2.132) to (2.140) with* $x_0 = [5, 0]$.

The point is that the particle filter is able to model the multi-modal distribution that results from the measurement model. Note that there could be occasions when the particle filter picks the wrong mode; persistently multimodal distributions do cause problems with the resampling step in particle filters and this is why so many particles are needed[15]. It is also worth noting that the weights are not being displayed. As a result, the diversity of the particles is slightly misleading at some points when the particles appear to fan out; the particles on the edge of the fan typically have low weights.

2.6.5 Smoothing

As with the Kalman filter, schemes do exist to perform smoothing[40, 56]; these consist of running the particle filter through the data and then performing a smoothing step that recurses backwards in time through the data processing the filtered distributions that were deduced. The particles' states are taken as fixed during this backwards recursion. The focus of [40] is on calculating the Maximum A-Posteriori path through the state space and on sampling trajectories, however it is straightforward to use the approach to deduce all the parameters of the (sample based approximation to the) joint distribution of the states within the path.

2.6.6 Frequently Encountered Pitfalls

One of the first papers on the subject proposed Sequential Importance Resampling, SIR[42], which is a special case of SIS. SIR defines that the proposal is the dynamic model, or dynamic prior, $q(x_t|x_{t-1}, y_{1:t}) \triangleq p(x_t|x_{t-1})$, and proposes to use multinomial resampling at every t. Particle filters have been criticised on the basis of the degeneracy that results when using this filter with systems that mix slowly in comparison to the evolution of the posterior; this happens when conducting parameter estimation or when using very informative measurements. This is because of two effects: firstly, the errors introduced by the (unnecessary) resampling dominate; secondly, and more importantly, the dynamic prior is often an appalling choice of proposal.

The solution often proposed is to use millions of particles. However, a more interesting approach is to consider what the particles are all doing. To do this, the computational cost of particle filters needs to be discussed.

2.6.7 Integration Methods

Before considering Monte carlo integration, it is instructive to briefly familiarise oneself with grid based methods for performing integrations. Numerical integration techniques such as the Rectangle rule, Trapezium rule and Simpson's rule can be used to carry out the integration. These techniques essentially

[15] Some recent work[119] has proposed a method for circumnavigating this problem by clustering the particles on the basis of their Euclidean distances from one another in the state space. For each cluster, on the basis of the total of the weights of the particles in the cluster, a decision is made as to whether to maintain or delete the cluster. The particles are then split between the clusters and resampling conducted on each cluster separately, so guaranteeing that the number of particles in a cluster remains constant and the representation of the multimodal distribution is maintained.

use a deterministically defined grid over the state space in order to then approximate the true integral, g, as the integral of a piecewise decomposition of the integrand. The different approaches then differ in what they consider each piecewise element to be. The differences are most easily understood in the scalar case.

$$g = \int f(x') p(x') dx' \approx \hat{g} \tag{2.223}$$

$$\hat{g} = \begin{cases} h \sum_{i=1}^{N} f\left(x^i\right) p\left(x^i\right) & \text{Rectangle} \\ \frac{h}{2} \sum_{i=1}^{N-1} f\left(x^i\right) p\left(x^i\right) + f\left(x^{i+1}\right) p\left(x^{i+1}\right) & \text{Trapezium} \\ \frac{h}{3} \sum_{i=2}^{N-1} f\left(x^{i-1}\right) p\left(x^{i-1}\right) + 4 f\left(x^i\right) p\left(x^i\right) + f\left(x^{i+1}\right) p\left(x^{i+1}\right) & \text{Simpson's} \end{cases} \tag{2.224}$$

The mean squared error, σ, in the integral is proportional to the grid spacing, h, to a power defined by the technique used, k ($k = 1$,$k = 2$ and $k = 4$ for the Rectangle, Trapezium and Simpson's rule respectively). The value of k is determined by looking at the Taylor series expansion used to derive the rules; $k + 1$ is the order of the lowest order terms ignored in the expansion. The error that this approximate integration method introduces is then:

$$\sigma = \sqrt{(g - \hat{g})^2} \propto h^k \tag{2.225}$$

Considering how a d dimensional space, where $x \in \mathbb{R}^d$, can be filled with points a distance h apart then gives rise to a relation between the error, the number of points and the dimensionality of the space.

$$h \propto N^{-d} \tag{2.226}$$

$$\sigma \propto N^{k-d} \tag{2.227}$$

As the dimensionality increases, the impact of the above is that a progressively large number of sample points are required to fill the space. This is the curse of dimensionality.

Monte Carlo integration facilitates the circumventing of this curse of dimensionality. This is highly counterintuitive and will now be explained. Monte Carlo integration does not use a deterministic scheme to choose the grid of samples, but samples stochastically N times from $p(x)$ to give samples $x^1 \ldots x^N$. The estimate of the integral is then the mean of the function $f(x)$ evaluated at these samples.

$$\hat{g} = \frac{1}{N} \sum_{i=1}^{N} f\left(x^i\right) \tag{2.228}$$

The strong law of large numbers then implies that as $N \to \infty$, $\hat{g} \to g$. So $\mathbb{E}[\hat{g}] = g$. The error can then be determined for a finite N.

$$\sigma^2 = \mathbb{E}\left[(\hat{g} - g)^2\right] = \mathbb{E}\left[\hat{g}^2\right] - 2\mathbb{E}[\hat{g}]\, g + g^2 \tag{2.229}$$

$$= \mathbb{E}\left[\hat{g}^2\right] - g^2 \tag{2.230}$$

$$= \frac{1}{N} \sum_{i=1}^{N} f\left(x^i\right)^2 - g^2 \tag{2.231}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left(f\left(x^i\right)^2 - Ng \right) \tag{2.232}$$

Thus, Monte Carlo integration gives errors of size inversely proportional to the root of N for any d. So, for any given d, changing N will have the same effect on the estimation error. This is true, but is also misleading (since it does not give any indication as to the affect of varying d with N fixed, but only the effect of varying N with d fixed). The apparent implication is that the dependence of the error on N is independent of d. This apparent confusion is the result of the effect of the constant of proportionality. Indeed, the constant of proportionality is not a function of dimensionality per se, since it is really a function of the peaked nature of the mismatch between the distribution from which the samples are drawn and the product of the function being integrated and the distribution. The flip-side is of this argument is that these mismatches are typically more peaked in higher dimensions since the product of a large number of flat scalar distributions will be very peaked in the higher dimensional space. So, the constant of proportionality, l, may be grossly affected by the dimensionality. A more intuitive way of expressing the error relation is perhaps:

$$\sigma = \frac{l\,(d)}{\sqrt{N}} \tag{2.233}$$

Monte Carlo integration relies on the use of a random number sequence. Such random number sequences are prone to statistical fluctuations; there is a finite probability that convergence will not have occurred within any given time scale. To combat this, Quasi-random number sequences have been used. Such Quasi-Monte Carlo [85] approaches offer slightly reduced performance in comparison to Monte Carlo integration, but are not prone to statistical effects in the same way.

So, to bring this argument to a close, it is essential, if one wants to make particle filtering a viable option in real-time environments, that the particles inhabit as low a dimensional space as possible.

2.6.8 Ergodicity

To be able to understand the dimensionality of the space which the particles inhabit, it is necessary to understand the concept of ergodicity. Ergodicity of a system is a measure of how quickly mixing it is. To understand this concept, it is helpful to consider two examples: In the first case, parameter estimation is considered. Parameters remain constant as time evolves, so if the value of the parameter is known at a point in time, the uncertainty over the value does not change with time; the system does not forget. In the second contrasting case, consider a dynamic system. For such a system, a known state at some point in the past still results in uncertainty over the state at some point in the future.

This differing rate with which the uncertainty grows with time is a measure of the history of the system. The faster the uncertainty grows, the shorter the memory. This memory is a measure of how quickly the system becomes unable to tell which of two states in the past are associated with another state at the present time. The less memory a system has, the more difficult it becomes to tell which of two previous states resulted in the current state. So, as a path extends with time, any error in the path has an effect that decreases as time progresses; the memory causes the filter to forget any errors.

Particle filters conduct importance sampling on the history of states. The samples need to populate this history sufficiently densely that any errors can be forgotten. So, the longer the memory of the system, the longer this history and hence the bigger the space that the particles must inhabit and the

more particles that will be required.

The impact of this is that it is advantageous to use systems with short memories.

2.6.9 Choice of Proposal Distribution

At first glance, the choice of system has been made at this point. However, if one considers the choice of proposal distribution as the choice of system which the particle filter simulates trajectories from, then different choices of proposal relate to different systems, different memories and so a potentially huge impact on the number of samples required. The closer that the proposal is to the posterior, the shorter the memory and so the fewer particles needed. The choice of proposal distribution is therefore a crucial step in the design of a particle filter.

The optimal choice of proposal (in terms of the conditional variance of the importance weights when the incremental proposal is of the form, $q(x_t|x_{t-1}, y_t)$) is to use $q(x_t|x_{t-1}, y_{1:t}) \triangleq p(x_t|x_{t-1}, y_t)$. If the only departure from linear Gaussian models is a nonlinearity in the dynamic model then this pdf can be sampled from. This means that the model has the following form:

$$x_t = a(x_{t-1}) + \omega_t^x \tag{2.234}$$

$$y_t = Hx_t + \omega_t^y \tag{2.235}$$

where $a(x)$ is a nonlinear function of x. As observed by a reviewer of [74], this form of model is popular[65, 97]; [97] refers to such models as cases when it is possible to do *exact adaption*.

This proposal is optimal within a certain class of proposals in terms of minimising the conditional variance of the importance weights[29]; so the sample used to extend the path of the particle turns out to have no effect on the weight update. This enables resampling to be carried out prior to the sampling, which can reduce the number of samples needed since the resampling ensures that the sampling is focused more efficiently. In some sense this is similar to the IEKF which readjusts its (linearisation) approximations in the light of the received measurements; here the samples are drawn from a distribution that has been readjusted to deter the samples from introducing degeneracy.

However, this optimal proposal cannot be used in the general case. Instead, such an approach can be used approximately and performs well when there is a need to interpolate between the measurement and the individual particles' state to form an approximation to the posterior, $p(x_t|x_{t-1}, y_t)$. The approximation can be carried out using an EKF[29] or UKF[117] based approximation for each particle.

A convenient alternative is to use the dynamics, $q(x_t|x_{t-1}, y_{1:t}) \triangleq p(x_t|x_{t-1})$. If one considers how close the posterior then is to the proposal, the difference will be small when the bulk of the information in the posterior comes from the history of the process rather than the measurement. Hence this proposal works well with diffuse likelihoods[16].

[16] Such likelihoods are found when considering low intensity targets moving in images. One might usually threshold the images and process the threshold crossings as *measurements*, but low intensity targets will cause this process to generate a large number of false alarms. While schemes have been proposed (by authors including the author of this thesis in collaboration with others) to enable efficient multi-target tracking to be conducted in such environments[77, 78, 89], comparisons indicate that the effects of the approximations introduced make it difficult to outperform a well implemented

By a similar argument, if the bulk of the information in the posterior comes from the measurement rather than the history of the process then using what can be thought of as a global approximation to the likelihood function can work well. In such cases, the proposal has some additional conditioning on the past measurements, but this can be implemented in terms of an approximation to the particle cloud:

$$q\left(x_t|x_{t-1}, y_{1:t}\right) \triangleq p\left(x_t|y_{1:t}\right) = \frac{p\left(y_t|x_t, y_{1:t-1}\right) p\left(x_t|y_{1:t-1}\right)}{p\left(y_t|y_{1:t-1}\right)} \tag{2.236}$$

$$\propto p\left(y_t|x_t, y_{1:t-1}\right) \int p\left(x_t, x_{t-1}|y_{1:t-1}\right) dx_{t-1} \tag{2.237}$$

$$\approx p\left(y_t|x_t, y_{1:t-1}\right) \int p\left(x_t|x_{t-1}\right) \hat{p}\left(x_{t-1}|y_{1:t-1}\right) dx_{t-1} \tag{2.238}$$

where $\hat{p}\left(x_{t-1}|y_{1:t-1}\right)$ is the Gaussian approximation to the particle cloud and the manipulations involved in the integral and multiplication use EKF or UKF based approximations. There is then one EKF or UKF for all the particles together. Other schemes exist for making use of the entire set of particles when proposing extensions to the particle's path. For example one can employ ideas analogous to cross-over in genetic algorithms to try to create each particle from the (good) constituents of many particles[89].

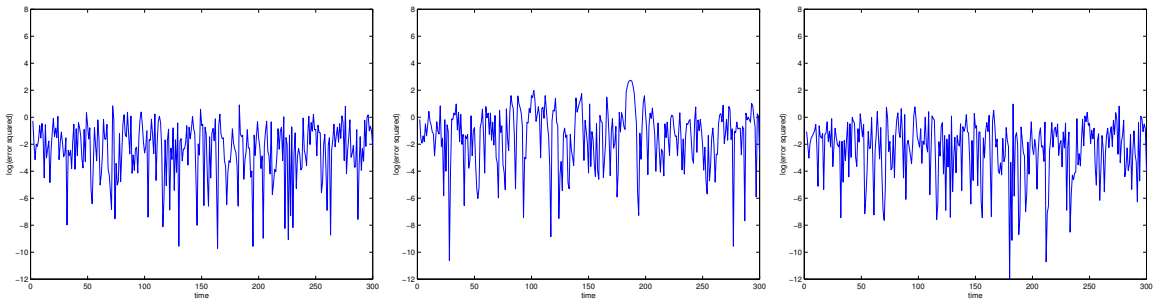
It is interesting to note that given an analytic form for the approximation, $\hat{p}\left(x_{t-1}|y_{1:t-1}\right)$, there could exist other choices of distribution for $p\left(x_t|x_{t-1}\right)$ and $p\left(y_t|x_t, y_{1:t-1}\right)$, for which, as a result of conjugacy relations, (2.238) is analytic (but for which $p\left(x_t|y_{1:t}\right)$ is not the same form as $\hat{p}\left(x_{t-1}|y_{1:t-1}\right)$).

A further approach is to encourage the particles to gradually move from the prior to the posterior. This has been done by using bridging densities [23] and progressive correction [91]. Both approaches introduce intermediate distributions between the prior and likelihood. The particles are then re-weighted according to these intermediate distributions and resampled. This "herds" the particles around the state space. A related technique is partitioned sampling [66]. This is useful if the likelihood is highly peaked, but can be factorised into a number of broader distributions. Typically, this occurs because each of the partitioned distributions are functions of some (not all) of the states and so are of lower dimensionality than the likelihood and so less peaked. By treating each of these partitioned distributions in turn and resampling on the basis of each such partitioned distribution, the particles are again herded around the state space. The use of intelligent sampling schemes is also inherent in the recent application of Quasi-Monte Carlo to particle filtering [88, 96].

Example

Such an approach is advantageous when using a particle filter to tackle the first case considered previously (when $x_0^{\text{true}} = \begin{bmatrix} 500 & 0 \end{bmatrix}$); while the (Extended) Kalman filter is arguably more appropriate to use than a particle filter in this case, the EKF does provide a benchmark and the case does highlight

particle filter that is based on [6]; one simply stacks the states of the targets ontop of one another and uses the dynamic prior as the (surprisingly good) proposal. It should be emphasised that while the approach of [89] is similar to that in [77, 78] in its approach, it introduces fewer approximations and always considers the joint distribution of all the targets. As a result, it is the approach advocated by the author for multi-target tracking with low-intensity targets in images.



(a) Extended Kalman Filter (b) Prior proposal (c) Linearised proposal

Figure 2.4: *A comparison of the errors for three filters for the first case.*

the possible benefit that results from using alternative proposal distributions in particle filters. We choose to use an EKF based approximation to the posterior, so the proposal takes on the following form:

$$q\left(x_t|x_{t-1}, y_{1:t}\right) \triangleq \mathcal{N}\left(x_t; m_t, C_t\right) \tag{2.239}$$

where

$$C_t = \left(Q^{-1} + H'R^{-1}H\right) \tag{2.240}$$

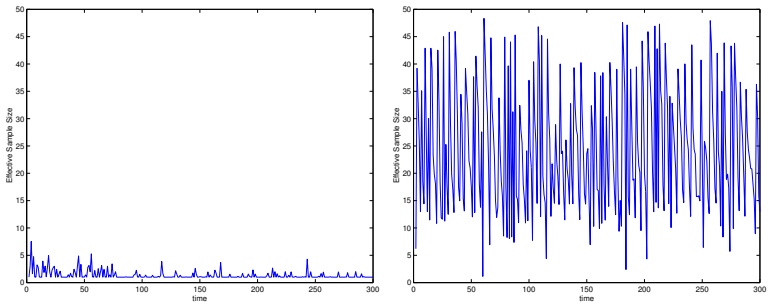
$$m_t = x_{t-1} + C_t H'R^{-1}\left(y_t - \left(FAx_{t-1}\right)^2\right). \tag{2.241}$$

We consider one long run with the same parameters as previously, but with a particle filter with just $N = 50$ particles and with resampling when N_{eff} falls below 15. The log squared estimation error for a Kalman filter, a particle filter with the dynamic prior as the proposal and a particle filter with this approximation to the posterior as the proposal are shown in figure 2.4. Evidently, at various iterations, using an EKF based approximation gives an improvement over the particle filter with the dynamic prior as a proposal in terms of this error.

As suggested by a reviewer of [74], it is particularly informative to look in more detail at the skewness of the particles' weights for both proposals. The effective sample size for the two proposals are given in figure 2.5. It is very evident that with the prior as the proposal, the effective sample size is consistently very near its lower bound of one. The particle filter that uses the approximation to the posterior as a proposal results in an effective sample size that is consistently higher. As a result, during the run, the particle filter that uses the prior as the proposal resampled at every time step whereas the particle filter with the more refined proposal resampled at 76 of the 300 iterations.

2.6.10 Jitter

An approach often advocated to rescue a particle filter from problems of degeneracy is to perturb the particles after resampling and so reintroduce diversity into the particle set. Since the resampling results



(a) Prior proposal (b) Linearised proposal

Figure 2.5: *A comparison of the effective sample size for a particle filter with a proposal that is either the prior and an approximation to the posterior based on linearised models.*

in a number of replicants of some of the members of the previous particle set, it might be sensible to attempt to push the particles apart to ensure that they explore different parts of the posterior. This jittering has been proposed since the outset of the recent research on particle filtering[42]. Some recent work has concentrated on formulating the approach as the sampling of the particles from a Kernel density approximation to the posterior; this results in regularised particle filters[84]. This enables, under some assumptions, the *optimal* kernel to be defined. However, a more rigorous approach is to accept any proposed jitter on the basis of a stochastic MCMC acceptance step[39]; this ensures that the samples are actually samples from the posterior of interest and so can be thought of as statistically rigorous jitter.

Another approach used to recover from degeneracy is the Auxiliary particle filter[97], often referred to as ASIR. This filter uses the same idea as mentioned previously in section 2.6.9 within the context of the optimal proposal. However, since in the general case, the weights are not known prior to the samples being drawn, ASIR calculates the weights using samples from the proposal and then conducts resampling. The difference to (normal) SIR is that the samples used to extend the particles paths are then removed and new extensions to the paths sampled (here, the samples that get removed are denoted $\hat{x}_t^i$ and the samples that result from the entire procedure are denoted x_t^i - as before). The impact is that those paths that give rise to highly weighted samples are resampled and extended. A minor complication arises; the new weight needs to remove the effect on the weight of the old sample, $\hat{x}_t^i$, and introduce the effect of the new sample, x_t^i:

$$x_t^i \sim q(x_t|x_{t-1}^i, y_{1:t}) \tag{2.242}$$

$$\bar{w}_{1:t}^i = \frac{p(y_t|x_t^i)p(x_t^i|x_{t-1}^i)}{q(x_t^i|x_{t-1}^i, y_{1:t})} \frac{q(\hat{x}_t^i|x_{t-1}^i, y_{1:t})}{p(y_t|\hat{x}_t^i)p(\hat{x}_t^i|x_{t-1}^i)}. \tag{2.243}$$

These weights then need to be normalised as previously.

2.6.11 Rao-Blackwellised Particle Filters

If the state-space that the particles populate is smaller, then one can imagine that fewer particles will be necessary. So, if it is possible to restrict the particles to live in a subset of the problem, it may be that the computational requirements can be reduced. Another way of looking at the same thing is to notice that analytic integration introduces no errors whereas any approximation must, by its nature, introduce errors. So, if it is necessary to approximate, one wants to do so as little as possible. Indeed, since

$$p(X,Y) = p(X)\,p(Y|X) = p(Y)\,p(X|Y)\,, \tag{2.244}$$

if one can calculate $p(X|Y)$ or $p(Y|X)$ analytically then there is no need to sample both X and Y.

This idea of mixing analytic integration with stochastic sampling is Rao-Blackwellisation. The benefit is sometimes difficult to quantify since, while one may need fewer particles, it may be that each Rao-Blackwellised particle is sufficiently more expensive than a vanilla particle that the overall computational cost is minimised by having more vanilla particles rather than fewer Rao-Blackwellised particles[17].

Rao-Blackwellisation can be thought of as expending the majority of the computational effort on the hard part of the problem. On an intuitive level, it can also be thought of in terms of a Kernel density approximation to the posterior. If the Kernels each only cover a small amount of the space, then one needs a large number of Kernels and it might well be more expedient to populate the posterior with samples. Conversely, one may get a very close match to the posterior with only a few Kernels.

Conditionally Discrete Distributions

The idea has been used implicitly since the outset of particle filtering when considering a likelihood that is a mixture distribution[4][18]. Conditioned on the path through the state space, it is common for some distribution over some discrete state to be analytically tractable. In [4, 69] this discrete state is the distribution over the association of the target with the measurements. However, it could equally be over the distribution over some discrete set of models or parameters and this idea is used in chapter 4.

Conditionally Linear Gaussian Distributions

Another example of Rao-Blackwellisation occurs when considering systems for which the measurement process is highly nonlinear while the dynamics are linear and Gaussian. Part of the state can then be Rao-Blackwellised. Each particle analytically integrates the distribution over part of the state using a Kalman filter (with correlated measurement and process noises) and the particles are only used to disambiguate the part of the state relating to the measurements[26, 77].

The system is assumed to evolve according to the linear-Gaussian model:

$$p(\theta_t|\theta_{t-1}) = \mathcal{N}(\theta_t; A\theta_{t-1}, Q) \tag{2.245}$$

[17] The same argument also holds when considering using different proposal distributions.

[18] Recently, this has been explicitly posed as Rao-Blackwellising the discrete distribution over the index of the mixture component[69].

where the state, θ_t, is partitioned into two parts, denoted, r_t and x_t:

$$\theta_t = \begin{bmatrix} r_t \\ x_t \end{bmatrix} \qquad A = \begin{bmatrix} A_{rr} & A_{rx} \\ A_{xr} & A_{xx} \end{bmatrix} \quad Q = \begin{bmatrix} Q_{rr} & Q_{rx} \\ Q_{xr} & Q_{xx} \end{bmatrix}. \tag{2.246}$$

One can re-write the state evolution (2.245) as follows:

$$\begin{bmatrix} r_t \\ x_t \end{bmatrix} = \begin{bmatrix} A_{rr} & A_{rx} \\ A_{xr} & A_{xx} \end{bmatrix} \begin{bmatrix} r_{t-1} \\ x_{t-1} \end{bmatrix} + \begin{bmatrix} Q_{rr} & Q_{rx} \\ Q_{xr} & Q_{xx} \end{bmatrix}^{\frac{1}{2}} \begin{bmatrix} \omega_k^z \\ \omega_k^x \end{bmatrix} \tag{2.247}$$

where $\omega_t^r \sim \mathcal{N}(\omega_t^r; \mathbb{O}, \mathbb{I})$ and $\omega_t^x \sim \mathcal{N}(\omega_t^x; \mathbb{O}, \mathbb{I})$ are mutually independent are the same size as r_t and x_t respectively. Since Q is a covariance matrix, $Q_{rx} = Q_{rx}^T$.

One member of the infinite set of matrix square roots of a matrix is given by[107]:

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix}^{\frac{1}{2}} = \begin{bmatrix} \left(A - BC^{-\frac{1}{2}^T} C^{-\frac{1}{2}} B^T\right)^{\frac{1}{2}} & BC^{-\frac{1}{2}^T} \\ \mathbb{O} & C^{\frac{1}{2}} \end{bmatrix} \tag{2.248}$$

Note that $C^{\frac{1}{2}} C^{\frac{1}{2}^T} = C \neq C^{\frac{1}{2}^T} C^{\frac{1}{2}}$, so if a non-symmetric matrix square root of C is used, care is necessary. One can then write (2.247) as:

$$r_t = A_{rr} r_{t-1} + A_{rx} x_{t-1} + \left(Q_{rr} - Q_{rx} Q_{xx}^{-\frac{1}{2}^T} Q_{xx}^{-\frac{1}{2}} Q_{xr}\right)^{\frac{1}{2}} \omega_k^r + Q_{xr} Q_{xx}^{-\frac{1}{2}^T} \omega_k^x \tag{2.249}$$

$$x_t = A_{xr} r_{t-1} + A_{xx} x_{t-1} + Q_{xx}^{\frac{1}{2}} \omega_k^x \tag{2.250}$$

Assuming that the inverse of A_{xx} exists, manipulation of (2.250) yields:

$$x_{t-1} = A_{xx}^{-1} x_t - A_{xx}^{-1} A_{xr} r_{t-1} - A_{xx}^{-1} Q_{xx}^{\frac{1}{2}} \omega_k^x \tag{2.251}$$

Substitution of (2.251) into (2.249) then yields:

$$\begin{aligned} r_t - \left(A_{rr} - A_{rx} A_{xx}^{-1} A_{xr}\right) r_{t-1} =& A_{rx} A_{xx}^{-1} x_t + \left(Q_{rr} - Q_{rx} Q_{xx}^{-\frac{1}{2}^T} Q_{xx}^{-\frac{1}{2}} Q_{xr}\right)^{\frac{1}{2}} \omega_k^r \\ &+ \left(Q_{xr} Q_{xx}^{-\frac{1}{2}^T} - A_{rx} A_{xx}^{-1} Q_{xx}^{\frac{1}{2}}\right) \omega_k^x \end{aligned} \tag{2.252}$$

So, knowing r_t and r_{t-1}, one can think of $r_t - \left(A_{rr} + A_{rx} A_{xx}^{-1} A_{xr}\right) r_{t-1}$ as being a linear Gaussian measurement of the state x_t, which evolves according to (biased) linear Gaussian dynamics. Specifically, the system is of the following form:

$$x_t = A x_{t-1} + b_t + \Gamma \omega_t^x \tag{2.253}$$

$$z_t = H x_t + \Psi \omega_t^z + \Lambda \omega_t^x \tag{2.254}$$

where

$$H = A_{rx} A_{xx}^{-1} \tag{2.255}$$

$$\Psi = \left(Q_{rr} - Q_{rx} Q_{xx}^{-\frac{1}{2}^T} Q_{xx}^{-\frac{1}{2}} Q_{xr} \right)^{1/2} \tag{2.256}$$

$$\Lambda = Q_{xr} Q_{xx}^{-1/2} - A_{rx} A_{xx}^{-1} Q_{xx}^{1/2} \tag{2.257}$$

$$A = A_{xx} \tag{2.258}$$

$$b_t = A_{xr} r_{t-1} \tag{2.259}$$

$$\Gamma = Q_{xx}^{1/2} \tag{2.260}$$

$$z_t = r_t - \left(A_{rr} - A_{rx} A_{xx}^{-1} A_{xr} \right) r_{t-1} \tag{2.261}$$

We can then use a Kalman filter to calculate $p\left(x_t | y_{1:t}\right) = p\left(x_t | r_{1:t}\right)$ and noting that the form of filter used needs to accommodate correlated measurement and process noise.

The measurement is then a nonlinear non-Gaussian function of r_t (which is a sufficient statistic of θ_t with respect to the measurement; eg. the measurements might relate to position with a state-space consisting of position, velocity and acceleration):

$$p\left(y_t | \theta_t\right) = p\left(y_t | r_t\right) \tag{2.262}$$

It is possible to then write:

$$p\left(\theta_{1:t} | y_{1:t}\right) = p\left(x_{1:t}, r_{1:t} | y_{1:t}\right) \tag{2.263}$$

$$= p\left(x_{1:t} | r_{1:t}\right) p\left(r_{1:t} | y_{1:t}\right) \tag{2.264}$$

and

$$p\left(x_t, r_{1:t} | y_{1:t}\right) = p\left(x_t | r_{1:t}\right) p\left(r_{1:t} | y_{1:t}\right) \tag{2.265}$$

Now $p\left(r_{1:t} | y_{1:t}\right)$ is the hard part of the problem, since if $r_{1:t}$ were known then (2.265) is (conditionally) Gaussian and its parameters can be optimally derived using a Kalman filter as described in section 2.3.2. The particles then represent $p\left(r_{1:t} | y_{1:t}\right)$, which evolves as follows:

$$p\left(r_{1:t} | y_{1:t}\right) \propto p\left(y_t | r_t\right) p\left(r_t | r_{1:t-1}\right) p\left(r_{1:t-1} | y_{1:t-1}\right) \tag{2.266}$$

As previously discussed in section 2.6.2, samples $r_{1:t}^i$ are formed by augmenting the samples $r_{1:t-1}^i$ with samples r_t^i as follows:

$$r_t^i \sim q\left(r_t | r_{t-1}^i, y_t\right) \tag{2.267}$$

The weights are then adjusted:

$$w_t^i \propto w_{t-1}^i \frac{p\left(y_t | r_t^i\right) p\left(r_t^i | r_{1:t-1}^i\right)}{q\left(r_t^i | r_{t-1}^i, y_{t-1}\right)} \tag{2.268}$$

It is instructive to make the following relation since $p\left(r_t^i | r_{1:t-1}^i\right)$ isn't deduced from the dynamics of r_t alone (ie. $p\left(r_t^i | r_{1:t-1}^i\right) \neq p\left(r_t^i | r_{t-1}^i\right)$):

$$p\left(r_t^i | r_{1:t-1}^i\right) = \int p\left(r_t^i, x_{t-1} | r_{1:t-1}^i\right) dx_{t-1} \tag{2.269}$$

$$= \int p\left(r_t^i | x_{t-1}, r_{1:t-1}^i\right) p\left(x_{t-1} | r_{1:t-1}^i\right) dx_{t-1} \tag{2.270}$$

So, $p\left(r_t|r^i_{1:t-1}, y_{1:t-1}\right)$ is the predicted density for r_t, based on the density for x_{t-1} deduced by the Kalman filter conditioned on $r^i_{1:t-1}$.

Such Rao-Blackwellised particle filters have been used as an elegant solution to the problem of Simultaneous Localisation and Mapping (SLAM) for which a map of a robot's environment is constructed at the same time as the robots location within the map[81][19].

In some circumstances, the Rao-Blackwellised distributions for all the particles are the same. In such cases, the parameters of the Gaussian distribution over part of the state need only be calculated once per iteration rather than once per particle[26, 44].

Example

In the example considered, tackling the problem using a filter that Rao-Blackwellises the velocity and only samples the position doesn't give much of an improvement since the Kernels turn out to be small; for constant velocity models, the space of velocities that could give rise to the same sequence of positions is sufficiently small that integrating it out doesn't give much benefit. This hasn't been discussed here in detail.

As observed by a reviewer of [74], were the parameters of the filter (eg. R and q) treated as unknown and were it possible to choose prior distributions such that the distributions of these parameters conditional on the state sequence was analytically tractable, one could Rao-Blackwellise these parameters. This could be done if one were to consider a discrete distribution over some candidate values for the parameters. However, this has many undesirable properties, such as having to fix these candidate values at the start of the run.

However, from a pragmatic perspective, when considering the first case, the nonlinearity is sufficiently unpronounced that approximately integrating the whole state is appealing. The approximately Rao-Blackwellised particle filter that results is an EKF!

2.7 Existing Tracking Algorithms

2.7.1 Introduction

The above discussion describes the constituents of algorithms facilitate sequential Bayesian inference in a generic context. The aim of this section is to discuss how these constituents have been applied to specific tracking problems.

[19] The robot observes the range and bearing of several landmarks which are assumed to be stationary. The particles then explore the uncertainty over the trajectory of the robot. For each particle, it transpires that the landmarks can then each be localised relative to the robot using a little Kalman filter for each landmark.

2.7.2 Maneuvering Targets

The motion of real targets is often decomposed into sections of quiescent motion interrupted by short turns. As a result, improved performance can be achieved by relaxing the assumption required to use a Kalman Filter that the dynamic model is linear and Gaussian.

Direct Modelling of Heavy Tailed Dynamics

One approach is to model the heavy-tailed distribution as non-Gaussian directly and to use particle filtering, however it is then non-obvious how to make the sampling efficient in the presence of the outliers. An alternative that has been proposed recently is to redefine the Kalman Filter in a heavy-tailed environment as the Kalman-Levy Filter[41, 113]. The heavy-tailed distributions result in multi-modal distributions since it is ambiguous if an observed outlier was caused by the dynamic process or by the measurement process and these two cases each result in a different mode in the posterior. The Kalman-Levy Filter does not therefore describe the posterior very well in this environment.

Generalised Pseudo Bayes-n

An alternative method for approximating a heavy tailed distribution in the locality of the mean is to use a (finite) discrete mixture of Gaussians[20]. This approach is widespread. The posterior then consists of a mixture of Gaussians since if the sequence of discrete indices (corresponding to the components in the mixture at each time index) were known then the Kalman filter could be used to deduce the parameters of the distribution over the state. So, the posterior is a mixture distribution with an exponentially increasing number of components. The algorithm that uses a Kalman filter for each such component is known as the Generalised Pseudo Bayes-n, GPBn, algorithm[10]. This is impractical for anything other than a very small number of time steps.

$$p\left(x_t|y_{1:t}\right) = \sum_{r_{1:t}} p\left(r_{1:t}|y_{1:t}\right) p\left(x_t|r_{1:t}, y_{1:t}\right) \tag{2.271}$$

$$= \sum_{r_1=1}^{M} \cdots \sum_{r_t=1}^{M} w_{t,r_1,\ldots,r_t|t} \mathcal{N}\left(x_t; m_{t|t,r_1,\ldots,r_t}, C_{t|t,r_1,\ldots,r_t}\right) \tag{2.272}$$

where r_t indexes the M models at time t and $m_{t|t,r_1,\ldots,r_t}$, $C_{t|t,r_1,\ldots,r_t}$ and $w_{t,r_1,\ldots,r_t|t}$ are respectively the mean, covariance and associated weight deduced from $y_{1:t}$ assuming that the sequence of models active was $r_1, \ldots, r_t$. $w_{t,r_1,\ldots,r_t|t}$ is then proportional to the joint likelihood of all the measurements given the model sequence, which can be calculated by multiplying $p\left(y_t|y_{1:t-1}, r_t\right) = \mathcal{N}\left(y_t; Hx_{t|t-1,r_t}, S_{t|t-1,r_t}\right)$ (where $S_{t|t-1,r_t}$ is the covariance of the innovations assuming that the model corresponding to r_t was active) at each iteration and multiplying by a transition probability, $p\left(r_t|r_{t-1}\right)$.

$$w_{t,r_1,\ldots,r_t|t} \propto \mathcal{N}\left(y_t; Hx_{t|t-1,r_t}, S_{t|t-1,r_t}\right) p\left(r_t|r_{t-1}\right) w_{t-1,r_1,\ldots,r_{t-1}|t-1}. \tag{2.273}$$

$w_{t,r_1,\ldots,r_t|t}$ sums to unity over all the paths, $r_1, \ldots r_t$.

[20] Note that heavy tailed distributions are strictly defined in terms of the rate of decay of the tails of the distribution in comparison to the rate of decay of a Gaussian distribution's tail. If one uses a mixture of Gaussians as an approximation to a heavy tailed distribution then the rate of decay of the tails is exactly that of a Gaussian. Hence, the approximation to a heavy tailed distribution is only valid in the locality of the mean - and not in the tails.

Generalised Pseudo Bayes 1 and 2

An alternative are the GPB1 and GPB2 algorithms, which approximate this distribution with fewer components. The GPB1[16], considers the possible active models at each time step and uses mixture reduction (as described in section 2.5.5) to collapse this distribution to a single Gaussian approximation to the posterior at the end of the iteration.

$$\sum_{r_t=1}^{M} w_{t,r_t|t}\mathcal{N}\left(x_t; m_{t|t,r_t}, C_{t|t,r_t}\right) \approx \mathcal{N}\left(x_t; m_{t|t}, C_{t|t}\right) \tag{2.274}$$

where $m_{t|t,r_t}$ and $C_{t|t,r_t}$ are deduced from $m_{t-1|t-1}$ and $C_{t-1|t-1}$ using the Kalman filter with the model corresponding to r_t and $w_{t,r_t|t}$ is just taken to be proportional to the model's likelihood at time t.

The GPB2[10] considers all pairs of models over two time steps and, at the end of the iteration, collapses down all components that correspond to the same model being active for the last iteration.

$$\sum_{r_{t-1}=1}^{M} w_{t,r_{t-1},r_t|t}\mathcal{N}\left(x_t; m_{t|t,r_{t-1},r_t}, C_{t|t,r_{t-1},r_t}\right) \approx w_{t,r_t|t}\mathcal{N}\left(x_t; m_{t|t,r_t}, C_{t|t,r_t}\right) \tag{2.275}$$

where $m_{t|t,r_{t-1},r_t}$ and $C_{t|t,r_{t-1},r_t}$ are deduced from $m_{t-1|t-1,r_{t-1}}$ and $C_{t-1|t-1,r_{t-1}}$ using the Kalman filter with the model corresponding to r_t and $w_{t,r_{t-1},r_t|t}$ is deduced from $w_{t-1,r_{t-1}|t-1}$ much in the same way as with the GPBn and is summed over r_{t-1} to calculate $w_{t,r_t|t}$:

$$w_{t,r_t,r_{t-1}|t} \propto \mathcal{N}\left(y_t; Hx_{t|t-1,r_t}, S_{t|t-1,r_t}\right) p\left(r_t|r_{t-1}\right) w_{t-1,r_{t-1}|t-1} \tag{2.276}$$

$$w_{t,r_t|t} = \sum_{r_{t-1}=1}^{M} w_{t,r_t,r_{t-1}|t}. \tag{2.277}$$

In both cases, the approximating components are an approximation to the true mixture - they are not components of it. This approximation assumes that combinations of models resulting in the same model being active at a given time produce the most similar components. This is not necessarily true, though in practical applications would seem to often be the case.

Interacting Multiple Model

The Interacting Multiple Model, IMM, algorithm [17] provides a popular comprise between the performance of GPB2 and computational expense of GPB1; it propagates a mixture with M components but only filters M not M^2 such components. Specifically, the mixture reduction is conducted before the filtering (rather than afterwards):

$$\sum_{r_{t-1}=1}^{M} w_{t-1,r_{t-1},r_t|t-1}\mathcal{N}\left(x_{t-1}; m_{t-1|t-1,r_{t-1},r_t}, C_{t-1|t-1,r_{t-1},r_t}\right) \approx w_{t-1,r_t|t-1}\mathcal{N}\left(x_{t-1}; m_{t-1|t-1,r_t}, C_{t-1|t-1,r_t}\right). \tag{2.278}$$

$w_{t,r_{t-1},r_t|t-1}$ are then calculated by using the last likelihood and the transition probabilities:

$$w_{t-1,r_{t-1},r_t|t-1} \propto \mathcal{N}\left(y_{t-1}; Hx_{t-1|t-1,r_{t-1}}, S_{t-1|t-1,r_{t-1}}\right) p\left(r_t|r_{t-1}\right) w_{t-2,r_{t-1}|t-2} \tag{2.279}$$

$$w_{t-1,r_t|t-1} = \sum_{r_t=1}^{M} w_{t-1,r_{t-1},r_t|t-1} \tag{2.280}$$

The Variable State-IMM, VS-IMM, is an extension of the IMM for the case when the number of models varies with time [63]. Typically the number of models varies because only some models can be applicable at any time. However, it can also be advantageous to vary the number of models since this may improve the approximation in the mixture reduction.

Multiple Hypothesis Filters

Rather than having a deterministic scheme for merging the hypotheses over some lag and so a computational expense that grows exponentially with the length of this lag, an alternative is to maintain a set of the most probable hypotheses. This has been conducted along with merging of similar hypotheses[32] and also in a stochastic particle filter context[31].

Comment

Often, the IMM and GPB algorithms are used in environments where given the sequence of models, the models are not linear and Gaussian, but are approximated as such. An obvious example is the tracking of aircraft using a radar. Due to the conversion between polar and Cartesian co-ordinates, the measurements are a nonlinear function of the state. However, the nonlinearity, as described earlier, is small at anything other than small ranges; across the support of the posterior that typically results, the variation in nonlinearity is small.

2.7.3 Data Association

The measurement to which each target relates is often ambiguous. This can be the result of missed detections or clutter (false alarms) or equally due to the presence of other targets. In this subsection, there is assumed to be a single target present. In the following subsection, methods for considering multiple targets will be discussed. In either case, the likelihood becomes a mixture with a component corresponding to each possible sequence of measurements. The growth of the mixture with time is again exponential making an exhaustive approach impossible for anything other than a very limited number of targets.

Direct Modelling of Heavy Tailed Likelihoods

One approach to coping with this problem is to model the likelihood using a heavy tailed distribution, such as a Student-T distribution. Such an approach is motivated on a statistical basis by considering the score function, the negative of the derivative of the log likelihood, $-\frac{d \log p(y|x)}{dx}$, evaluated at some measurement close to the predicted measurement, $y \approx y_{t|t-1}$. The score function is proportional to the change on the state estimate that results from receiving a measurement with a given innovation, $y - y_{t|t-1}$. In the case of a Gaussian model for the noise (so when using a Kalman Filter) the innovation is linearly related to the change in state estimate. As a result, an outlier results in a huge change in the estimate. This can be undesirable. A commonly adopted approach is then to use *gating*; a measurement is only used to update the state if the measurement is within some probabilistically defined region around the measurement predicted by the history of the process (one might define the region such that the probability

of a sample lying in the region was 95%, which can be calculated using the chi-squared distribution if the likelihood is Gaussian, as will be shown shortly). This results in a score function that is linearly related to the innovation in the gate and is zero outside the gate. The score function for this and other distributions described here are shown with their corresponding distributions in figure 2.6. The standard deviation of the Gaussian is taken to be 0.1. The gate is taken to be 2.5 standard deviations wide.

However, the definition of the (potentially probabilistic) threshold on the innovation is undesirable since it means that the behaviour is very different when measurements are received close to but on either side of the threshold. Using a Student-T distribution (with a particle filter to facilitate inference) gives the same asymptotic performance close to the predicted measurement and very far from it, but the score function transitions smoothly between these two extremes as can be seen in figure 2.6 shown with 5 degrees of freedom and the same standard deviation as the Gaussian. However, using a student-T is not easy to justify physically. An alternative approach is to use an *expected likelihood* for a Uniform distribution (for the clutter) over the observation region and a Gaussian centered on the expected measurement[69]. Given two probability distributions, $p_1(x)$ and $p_2(x)$ (which are here Uniform and Gaussian) the pdf for such a model is then defined as follows:

$$p(x) \propto \frac{p_1(x)}{p_1(x) + p_2(x)} p_1(x) + \frac{p_2(x)}{p_1(x) + p_2(x)} p_2(x) \tag{2.281}$$

where it is assumed that the right hand side can be integrated with respect to x such that $p(x)$ can exist.

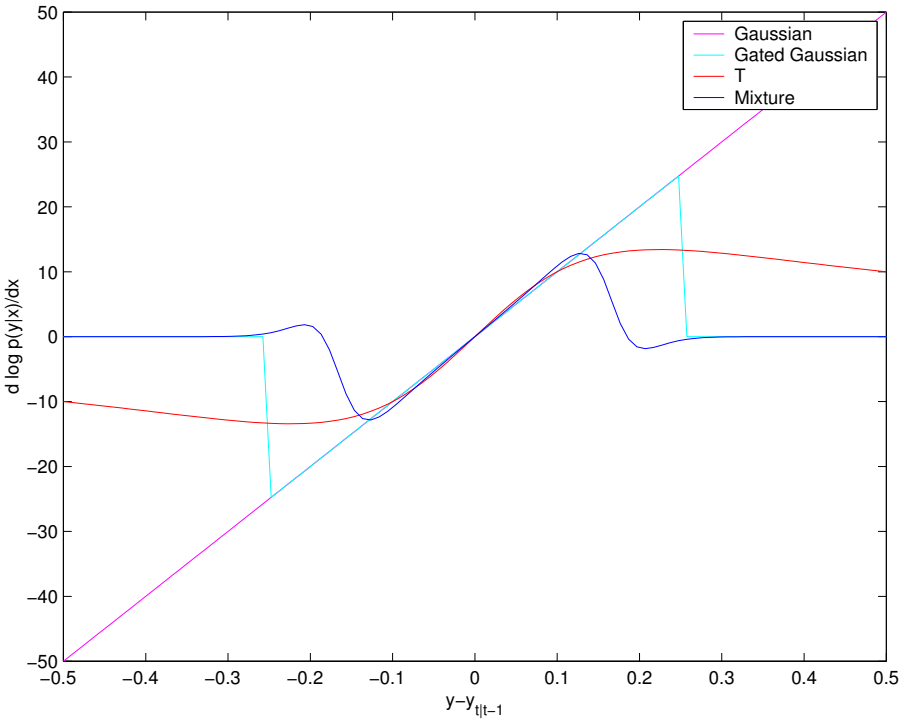
As can be seen in figure 2.6 for a uniform distribution over an area of 10 units, this results in a score function with a smooth transition, but this smooth transition looks very much like the score function when a gate is used[21]. However, the position for the transition is defined by the accuracy of the measurements and the extent of the uniform distribution and not by an arbitrary (albeit probabilistically defined) threshold. The use of a gate is therefore not as arbitrary as one might imagine at first sight.

If one does have such a gate[22] then the number of false alarms in the gate is often assumed to be Poisson distributed with the false alarms themselves distributed uniformly across the gate. The parameter of the Poisson model is then estimated (implicitly) from the number of measurements that are in the gate[23]. With M measurements in the gate the likelihood is then a mixture with each component consisting of a (joint) likelihood for all the measurements in the gate. The weights on the components in the mixture

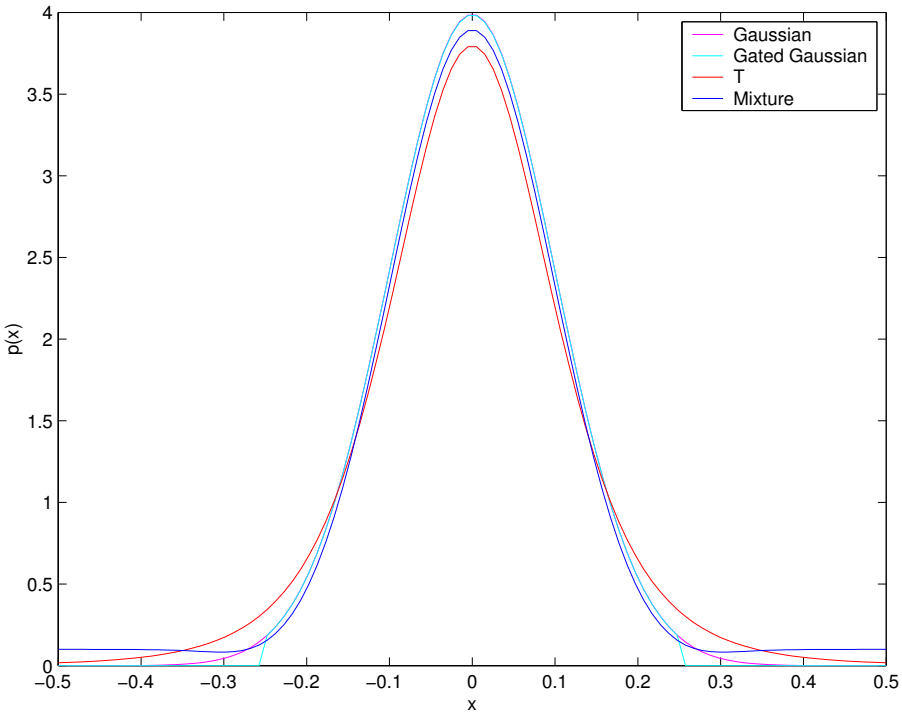
[21] The use of score functions is not totally appropriate here since the distribution is multi-modal; the pdf has local minima at $x \approx \pm 0.2$. This is why the score function crosses the x-axis at the corresponding values of x. However, the intuition gained is useful.

[22] If the predicted density is not a Gaussian, but a Gaussian mixture, then the gate is taken to either be the biggest of the gates for each of the components or some best fitting *overall* gate.

[23] One can also assume that the parameter of the Poisson distribution is linearly related to the volume of the gate and that the Poisson intensity is constant within defined spatial region for which this Poisson intensity is estimated, though such approaches will not be discussed here.



(a) Score Functions



(b) Probability Distributions

Figure 2.6: *Some probability distributions together with the corresponding score functions.*

corresponding to the measurements are then proportional to the likelihood for that hypothesis:

$$p\left(y_t^{1:M}|x_t\right) = w_t^0 + \sum_{i=1}^{M} w_t^i \mathcal{N}\left(y_t^i; Hx_t, S_t\right) \tag{2.282}$$

$$w_t^0 \propto p\left(y\left(x_t\right) \notin y_t^{1:M}\right) p\left(y_t^{1:M}|y\left(x_t\right) \notin y_t^{1:M}\right) \tag{2.283}$$

$$= \left(1 - P_D P_G\right) \left(\frac{1}{V}\right)^M \tag{2.284}$$

$$w_t^i \propto p\left(y\left(x_t\right) \in y_t^{1:M}\right) p\left(y_t^{1:M}|y\left(x_t\right) \in y_t^{1:M}\right) \tag{2.285}$$

$$= \frac{P_D}{M} \left(\frac{1}{V}\right)^{M-1} \mathcal{N}\left(y_t^i; Hx_t, S_t\right) \tag{2.286}$$

where $y\left(x_t\right) \in y_t^{1:M}$ denotes that the measurement corresponding to the target (which could be the empty set) is in the set of gated measurements, w_t^0 is the weight associated with the hypothesis that the target was not detected (or was outside the gate), P_D is the probability of being detected and P_G is the probability of the target being in the gate. The volume of a Gaussian gate can be calculated using the following relation:

$$V = \sqrt{|S|} c_{n_z} \lambda^{n_z/2} \tag{2.287}$$

where n_z is the dimensionality of the measurement space and c_n is the size of a unit hypersphere in n dimensional space (eg. if $n = 2$, $c_n = 2\pi$, while if $n = 3$, $c_n = \frac{4}{3}\pi$). λ is the size of the gate such that P_G is the integral of the χ^2 distribution (with n_z degrees of freedom) to λ, which is equivalent to the integral of the probability mass within a radius (from the origin) of λ (for an isotropic Gaussian with mean zero in n_z dimensions):

$$\int_{x=0}^{\lambda} p_{\chi^2, n_z}\left(x\right) dx = \int_{|x'| \leq \lambda, x' \in \mathcal{R}^{n_z}} \mathcal{N}(x; \mathbb{O}, \mathbb{I}) dx' = P_G. \tag{2.288}$$

It is worth noting that gating is sometimes advocated since the measurements outside the gate have negligible effect on the posterior; this is a result of the associated weights being small.

Nearest Neighbour Algorithms

A simpler approach to solving the data association problem is to simply consider the measurement nearest that predicted by the Kalman filter. Such Nearest Neighbour, NN, association is simple and effective if the scenario is simple.

An alternative is to chose the nearest neighbour in terms of some other distance; for example the Strongest-Nearest Neighbour algorithm, SNN, incorporates information relating to the strength of the return in the distance metric. Another alternative still is to use probability as the metric to find the nearest measurement (or hypothesis; the most likely hypothesis could be that none of the measurements relate to the target). This results in the Probabilistic-Nearest Neighbour algorithm[14].

All such approaches result in one association, which the filter then assumes to be correct.

Probabilistic Data Association Filter

The Probabilistic Data Association Filter, PDAF [9] carries out similar collapsing to the GPB1 in that it collapses the distribution down to a single Gaussian at each time step.

An alternative that is analogous to the GPB2 in the context of data association has also been proposed[93].

Track Splitting Filter

The Track Splitting Filter, TSF, maintains a Gaussian distribution for a set of the most probable hypotheses for the association history [10]. This is also known as the Track-Oriented Multiple Hypothesis Tracker[14]. As discussed in section 2.6.4, these approaches are equivalent to minimum error resampling with particle filters in this context. Particle filters with other resampling schemes have also been applied to this problem[31].

2.7.4 Multiple Targets

The primary problem with tracking multiple targets is twofold; there is ambiguity regarding the assignment of targets to measurements and it is uncertain as to how many targets are present.

Multi-Target Data Association

It is possible to use single target data association for each target in isolation. The posterior of the states for all the targets together is then approximated as a product of independent distributions for each target. It is typically the state of each target that is of interest and so the marginals of the posterior. However, there is no guarantee that the independently deduced distributions are indeed a good approximation to the marginals.

It is also worth noting that when considering a large number of targets, N^T, and a large number of measurements, N^M, the process of testing each combination of measurement and target to see if the measurement is in the target's gate can dominate the computational cost[124]. This process has a computational complexity that scales as $O\left(N^M N^T\right)$, so if either N^M is large this can dominate the computational expense of a tracker. An alternative scheme is to sort an informative dimension of the N^M measurements (in $O\left(N^M \log N^M\right)$ time) and then use a binary search (in $O\left(\log N^M\right)$ time) for each of the targets (so in $O\left(N^T \log N^M\right)$ time for all the targets together) to quickly find a subset of the measurements for which an exhaustive testing is necessary. This dramatically reduces the computational expense when the number of measurements is large.

The problem with considering the joint distribution of all the targets is that the number of possible assignments of targets to measurements grows exponentially with the number of targets. However, it is

possible to assume that the joint likelihood for each valid assignment[24] factorises across the targets:

$$p\left(y_t^{1:N^M}|x_t^{1:N^T},\theta,\theta\in V\right)=\prod_{i=1}^{N^T}\left[\frac{p\left(y_t^{j(i)}|x_t^i\right)}{p\left(y_t^{j(i)}\right)}\right]^{\mathbb{I}_{j(i)\neq 0}}\prod_{i=1}^{N^M}p\left(y_t^i\right) \tag{2.289}$$

where $j(i)$ is the measurement to which the ith target is assigned, which could be the non-detection hypothesis, θ is the assignment and V is the set of valid assignments. $\mathbb{I}_T$ is unity when T is true and zero otherwise.

If the joint likelihood is of this form then it is possible to find the maximum probability assignment of targets to measurements in substantially less than exponential time. The Global-Nearest Neighbour, GNN, algorithm (which is conceptually similar to the NN association algorithm) does exactly this. Algorithms that implement the algorithm include the Hungarian, auction, JVC and Munkres algorithms[10, 14]. The choice of algorithm most appropriate in a given environment is dictated by the denseness or difficulty of the assignment problem.

The Joint-PDAF, JPDAF, marginalises over the valid assignments so as to recalculate the weights on the mixture components constituting each targets' distribution[10]. A similar technique has been proposed in a particle filtering context, where it is known as Mutual Exclusion [66]. As is discussed in chapter 5, this marginalisation can also been carried out in substantially less than exponential time.

$$p\left(y_t^{1:N^M}|x_t^i,\theta\in V\right)=\sum_{\theta\in V}\int p\left(y_t^{1:N^M},\theta,x_t^{1:N^T}/x_t^i|x_t^i,\theta\in V\right)dx_t^{1:N^T}/x_t^i \tag{2.290}$$

$$=\sum_{\theta\in V}\int p\left(y_t^{1:N^M}|\theta,x_t^{1:N^T},\theta\in V\right)p\left(\theta|x_t^{1:N^T},\theta\in V\right)p\left(x_t^{1:N^T}/x_t^i|x_t^i,\theta\in V\right)dx_t^{1:N^T}/x_t^i \tag{2.291}$$

$$=\sum_{\theta\in V}\int p\left(y_t^{1:N^M}|x_t^{1:N^T},\theta,\theta\in V\right)p\left(\theta|\theta\in V\right)p\left(x_t^{1:N^T}/x_t^i\right)dx_t^{1:N^T}/x_t^i \tag{2.292}$$

$$\propto\int\frac{1}{p\left(x_t^i\right)}\prod_{i'=1}^{N^T}p\left(x_t^{i'}\right)\sum_{\theta\in V}\prod_{i'=1}^{N^T}\left[\frac{p\left(y_t^{j\left(i'\right)}|x_t^{i'}\right)}{p\left(y_t^{j(i')}\right)}\right]^{\mathbb{I}_{j(i')\neq 0}}\prod_{i'=1}^{N^M}p\left(y_t^{i'}\right)dx_t^{1:N^T}/x_t^i \tag{2.293}$$

where A/B is the set A excluding the element B and $p\left(\theta|\theta\in V\right)$ is taken to be a uniform prior over the valid associations (independent of the state of the targets since the measurement is not yet observed so that $p\left(\theta|\theta\in V\right)=p\left(\theta|x_t^{1:N^T},\theta\in V\right)$). $p\left(x_t^{1:N^T}\right)$ is assumed to factorise across the targets.

The Track-Oriented Multiple Hypothesis Tracker, MHT, itself similar to the TSF filter, picks the N-best associations for all the multiple targets jointly[102]. The performance of the filter has been shown to be improved if some delay is added in deciding which associations are the N-best[57]. This is because introducing a (fixed) lag makes the distribution over the possible assignments much closer to a delta function; picking the best association becomes a better approximation to marginalising over the association as a result of the lag. However, having a high fidelity representation of the fixed lag distribution is non-trivial for anything other than short lags. The probabilistic MHT algorithm, PMHT,

[24]For an assignment of targets to measurements to be valid, each target must relate to one measurement hypothesis (potentially the non-detection hypothesis) and each measurement must relate to at most one target.

uses a variational approximation to represent such a distribution and so is able to use EM over the associations[115]. However, this approximation makes it difficult to impose the true constraints.

The S-D assignment algorithm considers a sliding window of associations and uses Lagrangian relaxation to find the best (or N-best) sequence of associations over this lag[94, 95]. The assignment at the beginning of the window is then fixed. The assignment of targets to measurements is then posed as a constrained optimisation problem and Lagrangian relaxation used to solve this optimisation problem. Lagrangian relaxation pushes together an upper and lower bound to find a solution to an optimisation problem. In this context, the lower bound (on the probability of the association) is a potentially suboptimal solution that guarantees the constraints are satisfied deduced by assigning targets to measurements then measurements to the target-measurement pairs then target-measurement-measurement triples to measurements and so on. The upper bound solves a different unconstrained problem by modifying the costs in such a way that solutions that violate the constraints are penalised while solutions that satisfy the constraints remain unaffected.

Uncertain Numbers of Targets

It is often assumed that targets can appear and disappear. Tracks therefore need to be initiated and terminated. Simple approaches to dealing with uncertainty over the number of targets is to decouple the estimation of the number of targets from the estimation of their state. Tracks are then initiated using schemes that might find nearby measurement in consecutive scans of data and use least squares to estimate some state of the target given these measurements. Alternative schemes employ initiation on m-out-of-n scans of data. Tracks are typically terminated by thresholding such quantities as the determinant of the covariance associated with the track or the number of scans in which no measurements fell in the track's gate[10, 14].

These schemes are ad-hoc in that they do not explicitly consider the number of targets at time t, N_t^T, itself to be a random quantity. There are schemes that are receiving interest at the moment that do consider this uncertainty. If $x_t^{1:N_t^T}$ is the state of the targets at time t, then the true mutli-target posterior recursion is as follows[67]:

$$p\left(x_{t+1}^{1:N_{t+1}^T}|y_{1:t}\right) = \int p\left(x_{t+1}^{1:N_{t+1}^T}|x_t^{1:N_t^T}\right) p\left(x_t^{1:N_t^T}|y_{1:t}\right) dx_t^{1:N_t^T} \tag{2.294}$$

$$p\left(x_{t+1}^{1:N_{t+1}^T}|y_{1:t+1}\right) \propto p\left(y_{t+1}^{1:N^M}|x_{t+1}^{1:N_{t+1}^T}\right) p\left(x_{t+1}^{1:N_{t+1}^T}|y_{1:t}\right) \tag{2.295}$$

where $y_{1:t}$ is shorthand for all the received measurements and where $p\left(x_{t+1}^{1:N_{t+1}^T}|x_t^{1:N_t^T}\right)$ includes the birth of new targets, the death of some existing targets and the dynamics of the other existing targets:

$$p\left(x_{t+1}^{1:N_{t+1}^T}|x_t^{1:N_t^T}\right) = \prod_{i_t=1}^{N_t^T}\left(\left(1-P_D\left(x_t^{i_t}\right)\right)p\left(x_{t+1}^{i_{t+1}(i_t)}|x_t^{i_t}\right) + P_D\left(x_t^{i_t}\right)\right) P_B\left(N_B\right)\prod_{i=1}^{N_B} p_B\left(x_t^{i_t^b(i)}\right) \tag{2.296}$$

where

$$N_B = N_{t+1}^T - N_t^T - N_D \tag{2.297}$$

and where $P_D\left(x_t^{i_t}\right)$ is the probability of death, $P_B\left(n\right)$ is the probability of n targets being born, $i_{t+1}\left(i_t\right)$ is the index at time $t+1$ of the i_t target at time t and $i_t^b\left(i\right)$ is the index of the ith target born at time t. N_D is the number of targets that die and $p_B\left(x\right)$ is the distribution for the state of newborn targets. $p\left(y_{t+1}^{1:N^M}|x_{t+1}^{1:N_{t+1}^T},\theta\in V\right)$ is then an integral over the valid association hypotheses for an unknown number of targets:

$$p\left(y_{t+1}^{1:N^M}|x_{t+1}^{1:N_{t+1}^T},\theta\in V\right)\propto\sum_{N_{t+1}^T=0}^{\infty}p\left(N_{t+1}^T\right)\sum_{\theta\in V}\prod_{i=1}^{N_{t+1}^T}\left[\frac{p\left(y_{t+1}^{j(i)}|x_{t+1}^i\right)}{p\left(y_{t+1}^{j(i)}\right)}\right]^{\mathbb{I}_{j(i)\neq 0}}\prod_{i=1}^{N^M}p\left(y_{t+1}^i\right) \tag{2.298}$$

The problem is then how to model these distributions. One can sample the number of targets and their states jointly[1]. Another approach is to model the joint distribution of N^T and $x^{1:N^T}$ with a set of distributions for each of a number of N^T[18]:

$$p\left(x^{1:N^T}\right)\approx\sum_{N^T=0}^{N_{\max}^T}p\left(N^T\right)p\left(x^{1:N^T}|N^T\right) \tag{2.299}$$

where $N_{\max}^T$ is some upper limit on the number of targets that are thought to be present. Unfortunately, this approach does not scale well with the number of targets since the same target can appear in a (large) number of joint distributions if the number of targets is large. However, for a low number of targets, the algorithm has been shown to give good performance.

One can turn the conditioning the other way around:

$$p\left(x^{1:N^T}\right)\approx\int p\left(x\right)p\left(N^T|x\right)dx \tag{2.300}$$

where $p\left(N^T|x\right)$ is the limit as $dx\rightarrow 0$ of the distribution over the number of targets in the region of size dx centered on x. Such an approach has been adopted by the Probabilistic Hypothesis Density, PHD, filter where $p\left(N^T|x\right)$ is assumed to be a Poisson distribution[112, 121]. $p\left(x\right)$ is then represented as a weighted sample set consisting of a number of samples, x^i, and associated weights, w^i. $p\left(N^T|x^i\right)$ is then parameterised by a Poisson intensity, λ^i, and the filter stores $\left(w\lambda\right)^i=w^i\lambda^i$. The potential criticism of the approach is that the ordering of the targets is lost; there is no concept of how to tie the state of a given target at one point in time to the same target at another point in time. However, the performance shown has been impressive.

The prediction stage is reasonably straightforward to implement in both cases. The primary difficulty is how to implement the likelihood update. In [18], association ambiguity was not needed since there was no concept of measurements. In [112, 121], the assumption that the number of targets at a point is Poisson distributed is exploited to give rise to an analytic solution.

An alternative approach is to represent the uncertainty over the number of targets that are present by having a maximum number of targets that might be present and for each such target having a probability that the target is indeed present and a distribution as to where the target is given that it is present[82, 83]:

$$p\left(x^{1:N^T}\right)=\prod_{i=1}^{N_{\max}^T}\left(P_E^ip\left(x^i\right)+\left(1-P_E^i\right)\right) \tag{2.301}$$

where P_E^i is the probability that the ith target exists and $p\left(x^i\right)$ is the distribution for the ith target if it does exist. This makes it straightforward to implement the existing approaches to data association for a

given number of targets; a target not existing has the same impact on the other targets as it not being detected. Another advantage of this approach is that the labelling of the targets is preserved.

2.8 Conclusions

This chapter has described sequential algorithms including the particle filter and the many (Extended) Kalman filter based algorithms that are prevalent in the tracking literature. The fundamentals of representing uncertainty using a pdf have provided a starting point from which the discussion has progressed to considering methods for improving the computational cost of the particle filter and discussing the state-of-the-art in tracking, which forms the basis for the remainder of this thesis.

Discretisation 3

3.1 Introduction

There are a large number of different dynamic models used in the tracking literature. Each model has a number of forms[64]. The author believes that the reason that the same model has a number of forms is that the process of converting continuous time (stochastic) differential equations into (probabilistic) discrete time difference equations is not well understood by the tracking research community. Typically, the dynamics of the states are typically well defined and understood in a deterministic continuous context, whereas the models are to be used in a discrete context under the influence of continuous time stochastic effects. Though one can often use a sufficiently fine quantisation of continuous time to overcome any inadequacies in the discretisation process, the author believes that it is advisable to expend effort on the discretisation to enable a coarse quantisation to provide an adequate approximation to the continuous system. Indeed, as will be explained in the sequel, the discretisation process can lead to artifacts that are purely the result of when the discrete epochs occur; inserting an epoch can change the resulting discretisation. In most applications, this is undesirable from a pragmatic perspective but in some, such as when out-of-sequence measurements are received, this pragmatic desire can be replaced with a necessity to have high fidelity method for discretisation.

The introduction of randomness into the deterministic continuous differential equations can be analysed using Itô calculus[87]. Equally, it can be analysed by considering the noise as a 'Jerk' of input and then integrating the effect of the 'Jerk' (and the initial state) over time using a Taylor series. While the analysis using Itô calculus is exact, in that it deals with probability distributions at all points, it is complicated for anything other than very simple systems. However, the use of 'Jerk' models leads to dynamics which are of the form of equation 3.1 where $Q(x_\tau)$ is singular.

$$p(x_{\tau+\Delta}|x_\tau) = \mathcal{N}(x_{\tau+\Delta}, f(x_\tau), Q(x_\tau)) \tag{3.1}$$

As explained in section 2.6.11, the advantage of Rao-Blackwellising with conditional Gaussians is greatest when the Gaussian distributions have non-singular covariance matrices. This provides motivation to investigate alternative approaches to discretising processes that offer a compromise between the accuracy of the Itô method and the expediency of using 'Jerk' models.

The aim of this chapter is to provide an off-the-shelf approach to discretising processes that minimises

the need to understand and make use of the concepts in Itô calculus, which are unlikely to be familiar to the practitioner in the tracking research community, while making it possible to exploit Rao-Blackwellised particle filters with such models and so provide sequentially structured Bayesian solutions to a certain class of problems. The approach advocated is to make use of standard Laplace transform results that hold for linear differential equations and to approximate the (stochastic) differential equations with linear differential equations such that these results can be used. This offers a different compromise to a previously proposed approach based on linearising the Itô equations so as to produce linear stochastic differential equations[92, 110]; using such an approach is typically more complex and not easily applied for the larger systems considered here.

The different approaches including that proposed are outlined in sections 3.2, 3.3 and 3.4. Then the following sections consider a set of increasingly complex examples: the random walk in section 3.5; the constant velocity model in section 3.6; the constant velocity model with drag in section 3.7; the two dimensional constant turn rate model in section 3.8; the three dimensional coordinated turn model in section 3.8. The models are ordered with increasing complexity and increasing departure of the resulting discrete time models from those already prevalent in the literature[64]. The chapter concludes in section 3.10.

3.2 Itô Calculus

If τ denotes continuous time, then the Itô stochastic differentials of a stochastic process, x_τ are similar to classical differentials in that they represent infinitesimally small changes in a variable [87]. However there is one crucial difference: If dx is a stochastic differential then, on an intuitive level, its value is expressed probabilistically and not deterministically (eg. assuming dy to be deterministic, dx might take the value $+dy$ or $-dy$, each with equal probability)[1].

With that understood, a continuous time diffusion process can be described by an equation relating a small change in the state variable to a small change in continuous time and a small random walk.

$$dx_\tau = \mu(\tau, x_\tau)d\tau + \sigma(\tau, x_\tau)\, dw_\tau \tag{3.2}$$

where x_τ is a state at time τ, μ is known as the drift and σ is the volatility. $d\tau$ is a classical differential and dw_τ is a stochastic differential, a random walk of increment $\pm\sqrt{d\tau}$; dw_τ randomly takes the value $+\sqrt{d\tau}$ or $-\sqrt{d\tau}$, each with equal probability. The drift can be thought of as the deterministic element of the evolution of the state and the volatility dictates the size of the random element of x_τ's evolution[2].

[1] A more rigorous definition is to say that $d\tau$ is split into infinitely many periods of length $\delta\tau$. Over each such period, x is incremented with equal probability by either $\sqrt{\delta\tau}$ or $-\sqrt{\delta\tau}$. dx is then the result of the integration of this process over the time $d\tau$.

[2] Other schemes for manipulating stochastic differential equations exist (such as the use of Stratonovich integrals) and it is possible to devise equivalent descriptions of systems in such alternative schemes. The different methods result from different definitions of a stochastic differential ($\int \sigma(x)\, dw_t$), specifically where the volatility, $\sigma(x)$, is evaluated in each infinitesimal increment. Itô calculus assumes that the volatility is evaluated at the state at the start of the increment. This

The dimensionality of the random walk is defined in this way so as to ensure that the variance of the distance travelled by the random walk, w_τ, over time $d\tau$ is precisely $d\tau$.

Note that on any level, the time series, w_τ appears to be a random walk; its statistics are self-similar under an appropriate different scaling of each of time and space[3]. Thus w_τ is uncorrelated with $w_{\tau'}$ provided $\tau \neq \tau'$.

$$\mathbb{E}\left[w_{\tau'} w_\tau\right] = \begin{cases} \mathbb{E}\left[w_\tau^2\right] & \tau = \tau' \\ 0 & \tau \neq \tau' \end{cases} \tag{3.3}$$

The implication of the definition for the random walk, dw_τ, is that the square of the stochastic differential is equal to $d\tau$.

$$\left(dw_\tau\right)^2 = d\tau \tag{3.4}$$

So the expected value of the product, $dw_\tau dw_{\tau'}$ is $d\tau$ or 0 as follows:

$$\mathbb{E}\left[dw_\tau dw_{\tau'}\right] = \begin{cases} \mathbb{E}\left[\left(dw_\tau\right)^2\right] = \mathbb{E}\left[d\tau\right] = d\tau & \tau = \tau' \\ 0 & \tau \neq \tau' \end{cases} . \tag{3.5}$$

Interestingly, Einstein devised a similar argument to this in 1905[35], which was the same year that he formulated relativity. This is particularly interesting since the two ideas contradict each other; using the above and given a compact range of support for x_τ at time 0, for all positive time the support of x_τ is infinite since there is some finite probability that x_τ lies in any open set. This violates the ideas of relativity since the above imposes no limit to the support whereas relativity would impose such a constraint.

In any case, the knock on effect of (3.4) is that integrals with respect to dw_τ need special care since second order terms in dw_τ contribute under the stochastic integral. This proves important when considering more complex scenarios.

To discretise a process using Itô calculus one needs to integrate (3.2). However, while this is conceptually easy, for complex multivariate systems the calculation of the integrals can be difficult as will be illustrated in the sequel. The complication comes from the fact that the random process is closely coupled with the state's evolution.

An approach described for the scalar case in [92] and for the multi-dimensional case in [110] is to linearise the stochastic differential equations using the fact that they are of the following form:

$$dx_\tau = \phi\left(x_\tau\right) d\tau + \sigma dw_\tau \tag{3.6}$$

means that given a value for a purely diffusive process (with no drift) at time t, the expected value of the process at all points in the future is the value of the process at time t. Note that a *martingale* is often referred to in this context and is simply defined to be a process with this property; using Itô calculus, stochastic integrals give rise to martingale processes.

[3]While, under a strict definition, the process is therefore not fractal, it is intuitively appealing to draw a comparison with fractals, for which the process itself (rather than its statistics) would be self-similar under the same scaling of both time and space.

where x_τ and other such quantities are vectors. Itô's formula can then be used to relate a small change $d\phi(x_\tau)$ to small changes dx and $d\tau$ such that for vector quantities, x:

$$d\phi(x_\tau) = \Phi dx + M d\tau \tag{3.7}$$

where the elements of Φ and M are defined as follows:

$$\Phi_{ij} = \frac{d\phi_i}{dx_j} \tag{3.8}$$

$$M_i = \frac{1}{2}\text{trace}\left(\sigma\sigma^T H_i\right) \tag{3.9}$$

where H_k has elements

$$H_k = \left(\frac{d^2\phi_k(x)}{dx_i dx_j}\right) \tag{3.10}$$

and where $\phi_k(x)$ is the kth element of $\phi(x)$. Intuitively, the need for the $Md\tau$ term is the result of the fact that an integral is defined using a Taylor series as follows (for the scalar case):

$$df(x) = \lim_{\Delta x \to 0}\left(f(x+\Delta x) - f(x)\right) \tag{3.11}$$

$$= \lim_{\Delta x \to 0}\left(f(x) + \Delta x \frac{df(x')}{dx}\bigg|_{x'=x} + \frac{(\Delta x)^2}{2}\frac{d^2 f(x')}{dx^2}\bigg|_{x'=x} + ... - f(x)\right) \tag{3.12}$$

$$= \lim_{\Delta x \to 0}\left(\Delta x \frac{df(x')}{dx}\bigg|_{x'=x} + \frac{(\Delta x)^2}{2}\frac{d^2 f(x')}{dx^2}\bigg|_{x'=x} + ...\right) \tag{3.13}$$

In the case of classical integrals all terms after and including the $(\Delta x)^2$ term are second order (so don't need to be considered). However, if (3.4) holds then using (3.6):

$$(\Delta x)^2 = \left(\phi(x_\tau)\Delta\tau + \sigma\Delta w_\tau\right)^2 \tag{3.14}$$

$$= \phi(x_\tau)^2(\Delta\tau)^2 + 2\sigma\Delta w_\tau\phi(x_\tau)\Delta\tau + \sigma^2(\Delta w_\tau)^2 \tag{3.15}$$

$$= \phi(x_\tau)^2(\Delta\tau)^2 + 2\sigma\Delta w_\tau\phi(x_\tau)\Delta\tau + \sigma^2\Delta\tau \tag{3.16}$$

where the final term is first order (and all the others are not). Hence the terms relating to the second order derivatives need to be taken into account when conducting Itô integrals. Intuitively, the fact that the process is stochastic is resulting in deterministic effects on the process whatever the random realisation; the average random path is deterministic.

If Φ and M are taken to be constant, as in [110], it is possible to linearise the stochastic differential equations:

$$\phi(x_\tau) \approx \phi(x_{\tau_0}) + \Phi(x_\tau - x_{\tau_0}) + M(\tau - \tau_0) \tag{3.17}$$

and:

$$dx_\tau \approx \left(\phi(x_{\tau_0}) + \Phi(x_\tau - x_{\tau_0}) + M(\tau - \tau_0)\right)d\tau + \sigma dw_\tau \tag{3.18}$$

This approximating system of stochastic differential equations is linear and can be solved as follows:

$$x_{\tau_0+\Delta} - x_{\tau_0} = \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}d\tau\phi(x_{\tau_0}) + \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}(\tau-\tau_0)Md\tau + \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}\sigma dw_\tau \tag{3.19}$$

If Φ^{-1} exists, as assumed in [110], then:

$$x_{\tau_0+\Delta} - x_{\tau_0} = \Phi^{-1}\left(e^{\Phi\Delta} - I\right) + \left(\Phi^{-2}\left(e^{\Phi\Delta} - I\right) - \Phi^{-1}\Delta\right) M + \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}\sigma dw_\tau \tag{3.20}$$

where the term post-multiplied by M is derived using the standard result (shown for the scalar case for simplicity):

$$\int xe^{\alpha x} = \frac{xe^{\alpha x}}{\alpha} - \frac{e^{\alpha x}}{\alpha^2} + c \tag{3.21}$$

3.3 'Jerk' Model

'Jerk' models are much simpler and considerably less rigorous. The differential equations are integrated using a Taylor series expansion about the time, τ:

$$x_{\tau+\Delta} = x_\tau + \dot{x}_\tau\Delta + \ddot{x}_\tau\frac{\Delta^2}{2} + \ldots \tag{3.22}$$

where any truncation of an incomplete series will result in the relation being an approximation.

The uncertainty in the trajectory is then modelled by adding one more derivative than those that define the deterministic system and defining this rate of change to be a (Gaussian) probability distribution with zero mean and an appropriately chosen covariance. The covariance of $x_{\tau+\Delta}$ is then calculated by considering the effect of this input on $x_{\tau+\Delta}$.

3.4 Laplace Transforms

Laplace transforms can be used to discretise continuous time equations by using a Taylor series to approximate the nonlinear process with a simpler (local) approximation. The differential is approximated with a linearisation about $\mathbf{x}(\tau_0)$.

$$\dot{\mathbf{x}}_\tau^0 = \phi\left[\mathbf{x}_\tau^0\right] + \mathbf{w}_\tau^0 \tag{3.23}$$

$$\approx \phi\left[\mathbf{x}_{\tau_0}^0\right] + \Phi\left(\mathbf{x}_\tau^0 - \mathbf{x}_{\tau_0}^0\right) + \mathbf{w}_\tau^0 \tag{3.24}$$

where $\phi[\mathbf{x}_\tau]$ is the true nonlinear continuous time differential and Φ is now the matrix parameterising the approximating (locally linear) system. $\mathbf{w}_\tau$ is a continuous time input process that models the continuous time stochastic process.

Define

$$\mathbf{x}_\tau \triangleq \mathbf{x}_\tau^0 - \mathbf{x}_{\tau_0}^0 \tag{3.25}$$

$$\mathbf{w}_\tau \triangleq \phi\left[\mathbf{x}_{\tau_0}^0\right] + \mathbf{w}_\tau^0. \tag{3.26}$$

It is then clear that with this change of origin, (3.24) is of the following form:

$$\dot{\mathbf{x}}_\tau = \Phi \mathbf{x}_\tau + \mathbf{w}_\tau \tag{3.27}$$

The Laplace transform of a vector is defined as:

$$\bar{\mathbf{x}}(s) = \mathcal{L}(\mathbf{x}_\tau) = \int_0^\infty \mathbf{x}_\tau e^{-s\tau} d\tau \quad s > 0 \tag{3.28}$$

where $\mathcal{L}(\mathbf{x}_\tau)$ is the Laplace transform of $\mathbf{x}_\tau$.

Taking Laplace transforms of (3.27), gives:

$$\bar{\mathbf{x}}(s) = (s\mathbf{I} - \Phi)^{-1} (\bar{\mathbf{w}}(s) + \mathbf{x}_0) \tag{3.29}$$

The first term in (3.29) can be expressed as a power series and then inverse Laplace transforms taken. By definition (and analogy with the scalar case), the resulting expression is the matrix exponential, known as the state transition matrix.

$$\begin{aligned}
\mathcal{L}^{-1}\left((s\mathbf{I} - \Phi)^{-1}\right) &= \mathcal{L}^{-1}\left(\frac{1}{s}\left(\mathbf{I} - \Phi s^{-1}\right)^{-1}\right) \\
&= \mathcal{L}^{-1}\left(\sum_{k\geq 0} \Phi^k s^{-(k+1)}\right) \quad \text{if } k \to \infty, \Phi^k \to 0 \\
&= I + \Phi\tau + \Phi^2 \frac{\tau^2}{2!} + \ldots + \Phi^k \frac{\tau^k}{k!} + \ldots \\
&= e^{\Phi\tau}
\end{aligned} \tag{3.30}$$

Using this definition simplifies the notation when taking inverse Laplace transforms of (3.29).

$$\mathbf{x}_\tau = e^{\Phi\tau}\mathbf{x}_0 + \int_{\mu=0}^{\tau} e^{\Phi(\tau-\mu)}\mathbf{w}_\mu d\mu \tag{3.31}$$

A change of time origin then yields the general equation for discretising continuous processes using Laplace transforms, given by:

$$\mathbf{x}_{\tau_0+\Delta} = e^{\Phi\Delta}\mathbf{x}_{\tau_0} + \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}\mathbf{w}_\tau d\tau \tag{3.32}$$

This is of the form of (3.33) where clearly $A = e^{\Phi\Delta}$. It is assumed that, over the time from τ_0 to $\tau_0 + \Delta$, the input, $\mathbf{w}_\tau$, has a constant expected value and covariance. Thus, $\mathbf{b}$ can be easily defined and ω_{τ_0} is the process noise which has zero mean and a covariance defined in (3.35).

$$\mathbf{x}_{\tau_0+\Delta} = A\mathbf{x}_{\tau_0} + \mathbf{b} + \omega_{\tau_0} \tag{3.33}$$

where

$$\begin{aligned}
\mathbf{b} &= \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}\mathbb{E}[\mathbf{w}_\tau]\, d\tau \\
&= \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} d\tau \; \mathbb{E}[\mathbf{w}_\tau]
\end{aligned} \tag{3.34}$$

and

$$Q = \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} \mathbb{E}\left[\mathbf{v}_\tau \mathbf{v}_\tau^T\right] \left(e^{\Phi(\tau_0+\Delta-\tau)}\right)^T d\tau \tag{3.35}$$

where

$$\mathbf{v}_\tau = \mathbf{w}_\tau - \mathbb{E}\left[\mathbf{w}_\tau\right] \tag{3.36}$$

The use of Laplace Transforms for the discretisation of the original continuous time processes then reduces to the calculation of $\mathbf{b}$ and Q; $A = e^{\Phi\Delta}$ is not needed since according to the definition in (3.25), $\mathbf{x}_{\tau_0} = 0$:

$$\mathbf{x}^0_{\tau_0+\Delta} = \mathbf{x}^0_{\tau_0} + \mathbf{b} + \omega_{\tau_0} \tag{3.37}$$

This approach linearises the (deterministic) differential equations and is very similar to the approach of [110] that leads to (3.20) and which linearises the stochastic differential equations. The result is derived here using Laplace transforms rather than Itô calculus. This interpretation does provide a method for deriving $e^{\Phi\Delta}$ (using Laplace transforms), which, as will be illustrated later, makes it easier to see how to approximate the derivation of $e^{\Phi\Delta}$ so as to make the calculations tractable. This method can be thought of as an approximation to (3.20), which approximates $M \approx 0$. For high dimensional systems, the calculation of M can include a large number of terms making it unattractive. In short, while a linearisation of the Itô calculus results in the high fidelity solution of (3.20), a simpler approach more applicable for large systems, that does not require an understanding of Itô calculus (the complexity of which is arguably why the use of the approach is not widespread in the tracking literature) and offers the fidelity required to make use of Rao-Blackwellisation is to linearise the (deterministic) differential equations.

3.5 Random Walk

3.5.1 Model

Perhaps the simplest conceivable model used in tracking for the evolution of a state is the random walk. The differential of the state is simply the input, which is assumed to be a zero mean process with known covariance. This then implies an implicit Gaussian assumption on the statistics of the noise process.

$$\dot{x}_\tau = w_\tau \tag{3.38}$$

3.5.2 Application of Itô Calculus

A continuous random walk is then a special case of (3.2) with $\mu = 0$ and $\sigma\left(\tau, x_\tau\right) = \sigma$, a constant; it is a purely diffusive process.

$$dx_\tau = \sigma dw_\tau \tag{3.39}$$

It is possible to integrate the two sides of the equation. τ_t is the continuous time equivalent to the discrete time epoch t.

$$\begin{aligned}\int_{\tau=\tau_0}^{\tau_0+\Delta} dx_\tau &= \int_{\tau=\tau_0}^{\tau_0+\Delta} \sigma dw_\tau \\ x_{\tau_0+\Delta} - x_{\tau_0} &= \sigma\left(w_{\tau_0+\Delta} - w_{\tau_0}\right)\end{aligned} \tag{3.40}$$

By the central limit theorem, since the dw_τ are independently and identically distributed, any $w_{\tau_0} - w_{\tau_0+\Delta}$ will be distributed according to a zero-mean Gaussian. The variance of the Gaussian can be calculated as follows:

$$\mathbb{E}\left[\left(w_{\tau_0} - w_{\tau_0+\Delta}\right)\left(w_{\tau_0} - w_{\tau_0+\Delta}\right)^T\right] = \mathbb{E}\left[\int_{\tau=\tau_0}^{\tau_0+\Delta} dw_\tau \int_{\tau'=\tau_0}^{\tau_0+\Delta} dw_{\tau'}\right] \tag{3.41}$$

$$= \mathbb{E}\left[\int_{\tau=\tau_0}^{\tau_0+\Delta} \int_{\tau'=\tau_0}^{\tau_0+\Delta} dw_{\tau'} dw_\tau\right] \tag{3.42}$$

$$= \mathbb{E}\left[\int_{\tau=\tau_0}^{\tau_0+\Delta} \left(dw_\tau\right)^2\right] + \mathbb{E}\left[\int_{\tau=\tau_0}^{\tau_0+\Delta} \int_{\tau'=\tau_0,\tau'\neq\tau}^{\tau_0+\Delta} dw_{\tau'} dw_\tau\right] \tag{3.43}$$

$$= \int_{\tau=\tau_0}^{\tau_0+\Delta} \mathbb{E}\left[\left(dw_\tau\right)^2\right] + \int_{\tau=\tau_0}^{\tau_0+\Delta} \int_{\tau'=\tau_0,\tau'\neq\tau}^{\tau_0+\Delta} \mathbb{E}\left[dw_{\tau'} dw_\tau\right] \tag{3.44}$$

$$= \int_{\tau=\tau_0}^{\tau_0+\Delta} d\tau + 0 \tag{3.45}$$

$$= \Delta \tag{3.46}$$

where (3.5) is substituted in (3.44); for all pairs of $dw_{\tau'}$ and dw_τ in the second term, $\tau' \neq \tau$.

Since the second factor in the right hand side of (3.40) is of the form $w_{\tau_0} - w_{\tau_0+\Delta}$, a solution to the stochastic differential equation is forthcoming as follows:

$$x_{\tau+\Delta} - x_\tau \sim \mathcal{N}\left(0, \sigma^2\Delta\right) \tag{3.47}$$

so

$$p\left(x_{\tau+\Delta}|x_\tau\right) = \mathcal{N}\left(x_{\tau+\Delta}; x_\tau, \sigma^2\Delta\right). \tag{3.48}$$

3.5.3 Application of 'Jerk' Model

The use of a 'Jerk' model considers the noise to be an unknown velocity, $\dot{x}_\tau$. So, applying (3.22) to this system yields:

$$x_{\tau_0+\Delta} = x_{\tau_0} + \Delta\dot{x}_{\tau_0} \tag{3.49}$$

So:

$$\mathbb{E}\left[x_{\tau_0+\Delta}\right] = x_{\tau_0} \tag{3.50}$$

since

$$\mathbb{E}\left[\dot{x}_{\tau_0}\right] = 0 \tag{3.51}$$

Choosing

$$\mathbb{E}\left[\left(\dot{x}_{\tau_0}\right)^2\right] \triangleq \frac{\sigma^2}{\Delta} \tag{3.52}$$

then makes the discrete random walk have the same covariance as previously:

$$Q = \mathbb{E}\left[\left(x_{\tau_0+\Delta} - x_{\tau_0}\right)^2\right] = \mathbb{E}\left[\left(\Delta\dot{x}_{\tau_0}\right)^2\right] \tag{3.53}$$

$$= \Delta^2\mathbb{E}\left[\left(\dot{x}_{\tau_0}\right)^2\right] \tag{3.54}$$

$$= \Delta^2\left(\frac{\sigma^2}{\Delta}\right) = \Delta\sigma^2 \tag{3.55}$$

This variance, Q, describes the distribution but does not make it Gaussianly distributed. One could define it as such, but to show that the distribution is indeed exactly Gaussian, one could solve the stochastic differential equations as previously. Alternatively, one could define $\dot{x}_\tau$ to be Gaussianly distributed. Since x_τ is a linear function of $\dot{x}_\tau$, it must be Gaussianly distributed as previously:

$$p\left(x_{\tau_0+\Delta}|x_{\tau_0}\right) = \mathcal{N}\left(x_{\tau_0+\Delta}; x_{\tau_0}, \sigma^2\Delta\right). \tag{3.56}$$

3.5.4 Application of Laplace Transforms

In the case of this model, $\Phi = 0$ and so:

$$e^{\Phi\Delta} = \mathcal{L}^{-1}\left((s-0)^{-1}\right) = \mathcal{L}^{-1}\frac{1}{s} = 1 \tag{3.57}$$

Also, since $\phi\left(x\right) = \phi\left(x_{\tau_0}\right) = 0$, $\mathbb{E}\left[w\left(\tau\right)\right] = 0$ and so $\mathbf{b} = 0$.

Q can then be calculated as follows where the value of the input, at time, τ, $\mathbf{v}_\tau$, is assumed to have a variance of σ^2 (since this gives rise to the same value of Q):

$$Q = \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\tau_0+\Delta-\tau}\left(\sigma^2\right)\left(e^{\tau_0+\Delta-\tau}\right)^T d\tau \tag{3.58}$$

$$= \int_{\tau=\tau_0}^{\tau_0+\Delta}\left(\sigma^2\right)d\tau \tag{3.59}$$

$$= \Delta\sigma^2 \tag{3.60}$$

So, it is possible to find the moments as previously and then define that the distribution is Gaussian and matches these moments:

$$p\left(x_{\tau_0+\Delta}|x_{\tau_0}\right) = \mathcal{N}\left(x_{\tau_0+\Delta}; x_{\tau_0}, \sigma^2\Delta\right). \tag{3.61}$$

3.5.5 Discussion

All three approaches deduce the same (simple) result but with very differing degrees of complexity and rigour.

3.6 Constant Velocity Model

3.6.1 Model

The next most complex model is the (nearly) constant velocity model, which assumes the velocity evolves according to a random walk. The position, x_t, is then the integral of the velocity, $\dot{x}_t$. The noise is input as a change in the velocity:

$$\begin{bmatrix} \dot{x}_\tau \\ \ddot{x}_\tau \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_\tau \\ \dot{x}_\tau \end{bmatrix} + \begin{bmatrix} 0 \\ w_\tau \end{bmatrix} \tag{3.62}$$

3.6.2 Application of Itô Calculus

One can write down the two coupled stochastic differential equations that represent this system. To keep the notation simple, $\dot{x}_u$ is still used to denote the second state, though it technically isn't a differential any more.

$$dx_\tau = \dot{x}_\tau d\tau \tag{3.63}$$

$$d\dot{x}_\tau = \sigma dw_\tau \tag{3.64}$$

Instead of solving for the joint distribution directly, one can solve each equation separately, show that the joint distribution is Gaussian and then calculate the cross-covariance. Solving the second equation is very similar to the approach used for the random walk; the velocity evolves according to a random walk, so:

$$\dot{x}_{\tau_0+\Delta} - \dot{x}_{\tau_0} = \sigma \left(w_{\tau_0+\Delta} - w_{\tau_0} \right) \tag{3.65}$$

$$p\left(\dot{x}_{\tau_0+\Delta} | \dot{x}_{\tau_0}\right) = \mathcal{N}\left(\dot{x}_{\tau_0+\Delta}; \dot{x}_{\tau_0}, \sigma^2 \Delta\right). \tag{3.66}$$

The first equation is more complex but can be approached in a similar way.

$$\begin{aligned} \int_{\tau=\tau_0}^{\tau_0+\Delta} dx_\tau &= \int_{\tau=\tau_0}^{\tau_0+\Delta} \dot{x}_\tau d\tau \\ x_{\tau_0+\Delta} - x_{\tau_0} &= \int_{\tau=\tau_0}^{\tau_0+\Delta} \dot{x}_{\tau_0} d\tau + \int_{\tau=\tau_0}^{\tau_0+\Delta} \int_{\tau'=\tau_0}^{\tau} \sigma dw_{\tau'} d\tau \\ &= \dot{x}_{\tau_0}\Delta + \int_{\tau=\tau_0}^{\tau_0+\Delta} \sigma w_\tau d\tau \\ &= \dot{x}_{\tau_0}\Delta + \sigma \left(w'_{\tau_0+\Delta} - w'_{\tau_0} \right) \\ p\left(x_{\tau_0+\Delta} | x_{\tau_0}, \dot{x}_{\tau_0}\right) &= \mathcal{N}\left(x_{\tau_0+\Delta}; x_{\tau_0} + \dot{x}_{\tau_0}\Delta, \frac{\sigma^2 \Delta^3}{3} \right) \end{aligned} \tag{3.67}$$

Proving that the classical integral of a random walk is indeed Gaussian with the given covariance requires a consideration of the terms in the summation constituting the integral. To this end, the time

$[0, \Delta]$ is split into N epochs and the limit of the sum as $N \to \infty$ calculated. $\hat{w}_i$ is then $w_{\frac{\Delta i}{N}}$.

$$\begin{aligned}
\int_{\tau=0}^{\Delta} w_\tau d\tau &= \lim_{N\to\infty} \sum_{i=1}^{N} \hat{w}_i \frac{\Delta}{N} \\
&= \lim_{N\to\infty} \left(\hat{w}_1 + \hat{w}_2 + \ldots + \hat{w}_N\right) \frac{\Delta}{N} \\
&= \lim_{N\to\infty} \left(N\hat{w}_1 + (N-1)\left(\hat{w}_2 - \hat{w}_1\right) + \ldots + \left(\hat{w}_N - \hat{w}_{N-1}\right)\right) \frac{\Delta}{N} \\
&\sim \lim_{N\to\infty} \left(\mathcal{N}\left(0, \frac{N^2}{N}\Delta\right) + \mathcal{N}\left(0, \frac{(N-1)^2}{N}\Delta\right) + \ldots + \mathcal{N}\left(0, \frac{1}{N}\Delta\right)\right) \frac{\Delta}{N} \\
&\sim \lim_{N\to\infty} \frac{\Delta}{N} \sum_{i=1}^{N} \mathcal{N}\left(0, \frac{i^2}{N}\Delta\right) \\
&\sim \mathcal{N}\left(0, \Delta^3 \lim_{N\to\infty} \frac{1}{N^3} \sum_{i=1}^{N} i^2\right) \\
&\sim \mathcal{N}\left(0, \Delta^3 \lim_{N\to\infty} \frac{N\left(N+1\right)\left(2N+1\right)}{6N^3}\right) \\
&\sim \mathcal{N}\left(0, \frac{\Delta^3}{3}\right)
\end{aligned} \tag{3.68}$$

where $\mathcal{N}\left(m, C\right)$ is a sample from a Gaussian distribution with mean m and covariance C.

To prove that the joint distribution of w_τ and $v_\tau = \int_{\tau'=0}^{\Delta} w_{\tau'} d\tau'$ is Gaussian, a similar approach can be used. If $\hat{v}_i = v_{\frac{\Delta i}{N}}$, then $p\left(\hat{v}_i | \hat{v}_{i-1}\right)$ is Gaussian while $p\left(\hat{v}_1\right)$ and $p\left(\hat{w}_N | \hat{v}_1 \ldots \hat{v}_N\right)$ are delta functions. The joint distribution of $\hat{w}_N$ and $\hat{v}_1 \ldots \hat{v}_N$ is then a product of normal distributions and delta functions. The integral of this distribution with respect to $\{\hat{v}_i\}$, for all $\hat{v}_i$ for $i < N$, is then itself a normal distribution:

$$\begin{aligned}
p\left(\hat{w}_N, \hat{v}_N\right) &= \lim_{N\to\infty} \int_{\mathbb{R}^{N-1}} p\left(\hat{w}_N, \hat{v}_1 \ldots \hat{v}_N\right) d\{\hat{v}_i\}_{i<N} \\
&= \lim_{N\to\infty} \int_{\mathbb{R}^{N-1}} p\left(\hat{w}_N | \hat{v}_1 \ldots \hat{v}_N\right) p\left(\hat{v}_1 \ldots \hat{v}_N\right) d\{\hat{v}_i\}_{i<N} \\
&= \lim_{N\to\infty} \int_{\mathbb{R}^{N-1}} p\left(\hat{w}_N | \hat{v}_1 \ldots \hat{v}_N\right) \prod_{i=2}^{N} p\left(\hat{v}_i | \hat{v}_{i-1}\right) p\left(\hat{v}_1\right) d\{\hat{v}_i\}_{i<N}
\end{aligned} \tag{3.69}$$

Thus the noise vector is indeed jointly Gaussian. One can calculate the cross-covariance of $w'_{\tau_0+\Delta} - w'_{\tau_0}$ with $w_{\tau_0+\Delta} - w_{\tau_0}$.

$$\begin{aligned}
\mathbb{E}\left[\left(w_{\tau_0+\Delta} - w_{\tau_0}\right)\left(w'_{\tau_0+\Delta} - w'_{\tau_0}\right)\right] &= \mathbb{E}\left[\int_{\tau=\tau_0}^{\tau_0+\Delta} dw_\tau \int_{\tau'=\tau_0}^{\tau_0+\Delta} \left(\tau_0 + \Delta - \tau'\right) dw_{\tau'}\right] \\
&= \int_{\tau=\tau_0}^{\tau_0+\Delta} \left(\tau_0 + \Delta - \tau\right) d\tau \\
&= \frac{\Delta^2}{2}
\end{aligned} \tag{3.70}$$

where use has been made of (3.5) as previously.

This then gives a covariance matrix, Q:

$$Q = \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \tag{3.71}$$

and a transition matrix, A:

$$A = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \tag{3.72}$$

where:

$$p\left(\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix} | \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix}; A\begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix}, Q\right). \tag{3.73}$$

A desirable property of any model of the continuous process is that, Q_{13}, the uncertainty in the state resulting from progression from time τ_1 to $\tau_3 = \tau_1 + \Delta$ should be the same as, Q_{123}, that resulting from τ_1 to $\tau_2 = \tau_1 + \alpha\Delta$ to τ_3. This model satisfies this condition:

$$\begin{aligned}
Q_{13} &= \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\alpha\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \\
Q_{12} &= \begin{bmatrix} \frac{(\alpha\Delta)^3}{3} & \frac{(\alpha\Delta)^2}{2} \\ \frac{(\alpha\Delta)^2}{2} & (\alpha\Delta) \end{bmatrix} \sigma^2 \\
Q_{23} &= \begin{bmatrix} \frac{((1-\alpha)\Delta)^3}{3} & \frac{((1-\alpha)\Delta)^2}{2} \\ \frac{((1-\alpha)\Delta)^2}{2} & (1-\alpha)\Delta \end{bmatrix} \sigma^2 \\
A_{23} &= \begin{bmatrix} 1 & (1-\alpha)\Delta \\ 0 & 1 \end{bmatrix} \\
Q_{123} &= A_{23}Q_{12}A_{23}^T + Q_{23} \\
&= \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \\
&= Q_{13}
\end{aligned} \tag{3.74}$$

This feature is particularly desirable when considering out-of-sequence measurements[4]. If $Q_{123} \neq Q_{13}$ then the insertion the out-of-sequence time (irrespective of the measurement) affects the probability of the transition from the state before to the state after. This means that the effect of the dynamics change and can swamp the impact of the likelihood of the out-of-sequence measurement itself. This is most pronounced when processing an out-of-sequence measurement with a particle filter[90] using dynamics with singular covariance matrices. For each particle, the states, x_1 and x_3, respectively relate to times just before and after the time of the out-of-sequence measurement, are fixed (as a result of the sampling of the state). If Q is singular, then $p(x_2|x_1)$ and $p(x_2|x_3)$ both have support restricted to subspaces of the state space. As a result of the choice of dynamic model, the states at the time of the out-of-sequence measurement are then restricted to the intersection of the supports of these distributions. This region of common support could be a delta function or could be non-existent. This extreme case of the problem is not encountered when using Kalman filter based methods for processing out-of-sequence measurements

[4]Out-of-sequence measurements are received in a different order to the order of times to which they relate and often result from communications delays.

since the support of the state at the times after and before the time of the out-of-sequence measurements is more than a single state.

3.6.3 Application of 'Jerk' Model

The 'Jerk' model now assumes the noise to be an unknown acceleration, $\ddot{x}_\tau$. So, applying (3.22) to this system yields:

$$x_{\tau_0+\Delta} = x_{\tau_0} + \Delta\dot{x}_{\tau_0} + \frac{\Delta^2}{2}\ddot{x}_{\tau_0} \tag{3.75}$$

$$\dot{x}_{\tau_0+\Delta} = \dot{x}_{\tau_0} + \Delta\ddot{x}_{\tau_0} \tag{3.76}$$

which can then be rewritten in terms of this unknown acceleration:

$$\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} + \begin{bmatrix} \frac{\Delta^2}{2} \\ \Delta \end{bmatrix} \ddot{x}_{\tau_0} \tag{3.77}$$

So:

$$\mathbb{E}\left\{ \begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix} \right\} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} \tag{3.78}$$

By assuming that the unknown acceleration has variance $\frac{\sigma^2}{\Delta}$, one then arrives at a covariance matrix, Q:

$$Q = \begin{bmatrix} \frac{\Delta^2}{2} \\ \Delta \end{bmatrix} \begin{bmatrix} \frac{\Delta^2}{2} & \Delta \end{bmatrix} \frac{\sigma^2}{\Delta} = \begin{bmatrix} \frac{\Delta^3}{4} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \tag{3.79}$$

This has some unappealing quantities. The most obvious is that there is only one degree of freedom, the unknown acceleration. As a result, the covariance matrix is singular.

The approach results in an unappealing artifact in that the uncertainty resulting from going from time τ_1 to τ_3 is no longer the same as that resulting from τ_1 to τ_2 to τ_3:

$$\begin{aligned}
Q_{13} &= \begin{bmatrix} \frac{\Delta^3}{4} & \frac{\alpha\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \\
Q_{12} &= \begin{bmatrix} \frac{(\alpha\Delta)^3}{4} & \frac{(\alpha\Delta)^2}{2} \\ \frac{(\alpha\Delta)^2}{2} & (\alpha\Delta) \end{bmatrix} \sigma^2 \\
Q_{23} &= \begin{bmatrix} \frac{((1-\alpha)\Delta)^3}{4} & \frac{((1-\alpha)\Delta)^2}{2} \\ \frac{((1-\alpha)\Delta)^2}{2} & (1-\alpha)\Delta \end{bmatrix} \sigma^2 \\
A_{23} &= \begin{bmatrix} 1 & (1-\alpha)\Delta \\ 0 & 1 \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
Q_{123} &= A_{23} Q_{12} A_{23}^T + Q_{23} \\
&= \begin{bmatrix} \frac{\Delta^3(1+\alpha(1-\alpha))}{4} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \\
&\neq Q_{13}
\end{aligned} \tag{3.80}$$

This means that the uncertainty differs as a result of the existence of the epoch. This is highly undesirable.

3.6.4 Application of Laplace Transforms

In the case of this model:

$$\Phi = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{3.81}$$

$$\phi\left(\begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix}\right) = \Phi \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} = \begin{bmatrix} \dot{x}_{\tau_0} \\ 0 \end{bmatrix} \tag{3.82}$$

$$\mathbb{E}\left[\mathbf{v}_\tau {\mathbf{v}_\tau}^T\right] = \begin{bmatrix} 0 & 0 \\ 0 & \sigma^2 \end{bmatrix} \tag{3.83}$$

so:

$$e^{\Phi\Delta} = \mathcal{L}^{-1}\left(\begin{bmatrix} s & -1 \\ 0 & s \end{bmatrix}^{-1}\right) = \mathcal{L}^{-1}\left(\begin{bmatrix} \frac{1}{s} & \frac{1}{s^2} \\ 0 & \frac{1}{s} \end{bmatrix}^{-1}\right) = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \tag{3.84}$$

and then:

$$\mathbf{b} = \int_{\tau=\tau_0}^{\tau_0+\Delta} \begin{bmatrix} 1 & \tau_0+\Delta-\tau \\ 0 & 1 \end{bmatrix} d\tau \begin{bmatrix} \dot{x}_{\tau_0} \\ 0 \end{bmatrix} \tag{3.85}$$

$$= \begin{bmatrix} \Delta & (\tau_0+\Delta)\Delta - \frac{1}{2}(\tau_0+\Delta)^2 + \frac{1}{2}\tau_0^2 \\ 0 & \Delta \end{bmatrix} \begin{bmatrix} \dot{x}_{\tau_0} \\ 0 \end{bmatrix} = \begin{bmatrix} \Delta & -\frac{1}{2}\Delta^2 \\ 0 & \Delta \end{bmatrix} \begin{bmatrix} \dot{x}_{\tau_0} \\ 0 \end{bmatrix} \tag{3.86}$$

$$= \begin{bmatrix} \Delta\dot{x}_{\tau_0} \\ 0 \end{bmatrix} \tag{3.87}$$

so

$$\mathbb{E}\left\{ \begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix} \right\} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} \tag{3.88}$$

and

$$\begin{aligned} Q &= \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} \begin{bmatrix} 0 & 0 \\ 0 & \sigma^2 \end{bmatrix} \left(e^{\Phi(\tau_0+\Delta-\tau)}\right)^T d\tau \\ &= \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \end{aligned} \tag{3.89}$$

3.6.5 Discussion

The use of Itô calculus and Laplace transforms both result in the same model. The use of Laplace transforms is substantially simpler though less rigorous. The 'Jerk' model gives rise to a slightly simpler model, but this model has some undesirable properties; the resulting covariance matrix is singular and this results in the uncertainty relating to the process being affected by the presence of epochs.

3.7 Constant Velocity Model with Drag

3.7.1 Model

The next most complex model is the (nearly) constant velocity model with drag. This is similar to the previous model, but also includes a term that would cause the velocity to exponentially decay to zero were there no input:

$$\begin{bmatrix} \dot{x}_\tau \\ \ddot{x}_\tau \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -K \end{bmatrix} \begin{bmatrix} x_\tau \\ \dot{x}_\tau \end{bmatrix} + \begin{bmatrix} 0 \\ w_\tau \end{bmatrix} \tag{3.90}$$

where K is the drag coefficient.

3.7.2 Application of Itô Calculus

It is straightforward to write the stochastic differential equations for this system:

$$dx_\tau = \dot{x}_\tau d\tau \tag{3.91}$$

$$d\dot{x}_\tau = -K\dot{x}_\tau d\tau + \sigma dw_\tau \tag{3.92}$$

To solve this pair of coupled stochastic differential equations, note that it is an Ornstein-Uhlenbeck process for which the stochastic differential equations are linear and for which the solution is known to be (see [68] for example):

$$\mathbb{E}\left\{\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix}\right\} = \begin{bmatrix} \frac{1}{K}\left(1-e^{-K\Delta}\right)\dot{x}_{\tau_0} + x_{\tau_0} \\ e^{-K\Delta}\dot{x}_{\tau_0} \end{bmatrix} \tag{3.93}$$

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12} & Q_{22} \end{bmatrix} \tag{3.94}$$

where

$$Q_{11} = \mathbb{E}\left\{\int_{\tau_0}^{\tau_0+\Delta}\left(\frac{\sigma}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right)dw_\tau\right)^2\right\} \tag{3.95}$$

$$= \mathbb{E}\left\{\int_{\tau_0}^{\tau_0+\Delta}\frac{\sigma^2}{K^2}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right)^2 d\tau\right\} \tag{3.96}$$

$$= \frac{\sigma^2}{K^2}\int_{\tau_0}^{\tau_0+\Delta}\left(1-2e^{-K(\tau_0+\Delta-\tau)}+e^{-2K(\tau_0+\Delta-\tau)}\right)d\tau \tag{3.97}$$

$$= \frac{\sigma^2}{K^2}\left(\Delta - \frac{2}{K}\left(1-e^{-K\Delta}\right)+\frac{1}{2K}\left(1-e^{-2K\Delta}\right)\right) \tag{3.98}$$

$$Q_{12} = \mathbb{E}\left\{\int_{\tau'=\tau_0}^{\tau_0+\Delta}\int_{\tau=\tau_0}^{\tau_0+\Delta}\frac{\sigma}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right)\sigma e^{-K\left(\tau_0+\Delta-\tau'\right)}dw_\tau dw_{\tau'}\right\} \tag{3.99}$$

$$= \int_{\tau=\tau_0}^{\tau_0+\Delta}\frac{\sigma^2}{K}\left(e^{-K(\tau_0+\Delta-\tau)}-e^{-2K(\tau_0+\Delta-\tau)}\right)d\tau \tag{3.100}$$

$$= \frac{\sigma^2}{K}\left(\frac{1}{K}\left(1-e^{-K\Delta}\right)-\frac{1}{2K}\left(1-e^{-2K\Delta}\right)\right) \tag{3.101}$$

$$Q_{22} = \mathbb{E}\left\{\left(\int_{\tau_0}^{\tau_0+\Delta}\sigma e^{-K(\tau_0+\Delta-\tau)}dw_\tau\right)^2\right\} \tag{3.102}$$

$$= \int_{\tau_0}^{\tau_0+\Delta}\sigma^2 e^{-2K(\tau_0+\Delta-\tau)}d\tau \tag{3.103}$$

$$= \frac{\sigma^2}{2K}\left(1-e^{-2K\Delta}\right) \tag{3.104}$$

It is useful to note that for small $K\Delta$:

$$e^{-K\Delta} \approx 1 - K\Delta \approx 1 \tag{3.105}$$

$$\frac{1}{K}\left(1 - e^{-K\Delta}\right) \approx \frac{1}{K}\left(1 - \left(1 - K\Delta + \frac{1}{2}K^2\Delta^2 - \frac{1}{6}K^3\Delta^3\right)\right) \tag{3.106}$$

$$= \Delta - \frac{1}{2}K\Delta^2 + \frac{1}{6}K^2\Delta^3 \approx \Delta \tag{3.107}$$

$$\frac{1}{2K}\left(1 - e^{-2K\Delta}\right) \approx \frac{1}{2K}\left(1 - \left(1 - 2K\Delta + \frac{1}{2}4K^2\Delta^2 - \frac{1}{6}8K^3\Delta^3\right)\right) \tag{3.108}$$

$$= \Delta - K\Delta^2 + \frac{2}{3}K^2\Delta^3 \approx \Delta \tag{3.109}$$

so:

$$\frac{1}{K}\left(\frac{1}{K}\left(1 - e^{-K\Delta}\right) - \frac{1}{2K}\left(1 - e^{-2K\Delta}\right)\right) \approx \frac{1}{K}\left(-\frac{1}{2}K\Delta^2 + K\Delta^2\right) = \frac{\Delta^2}{2} \tag{3.110}$$

$$\frac{1}{K^2}\left(\Delta - \frac{2}{K}\left(1 - e^{-K\Delta}\right) + \frac{1}{2K}\left(1 - e^{-2K\Delta}\right)\right) \approx \frac{1}{K^2}\left(-\frac{1}{3}K^2\Delta^3 + \frac{2}{3}K^2\Delta^3\right) = \frac{\Delta^3}{3} \tag{3.111}$$

$$\tag{3.112}$$

So, as one might expect, in the case of small $K\Delta$ (so small time increments, Δ, or small drag coefficient, K, or both) the constant velocity model with drag is well approximated by the previously described constant velocity model.

3.7.3 Application of 'Jerk' Model

Again, the noise is assumed to be an unknown acceleration, $\ddot{x}_\tau$.

$$x_{\tau_0+\Delta} = x_{\tau_0} + \Delta\dot{x}_{\tau_0} + \frac{\Delta^2}{2}\ddot{x}_{\tau_0} \tag{3.113}$$

$$\dot{x}_{\tau_0+\Delta} = \dot{x}_{\tau_0} + \Delta\left(-K\dot{x}_{\tau_0} + \ddot{x}_{\tau_0}\right) \tag{3.114}$$

which can then be rewritten in terms of this unknown acceleration:

$$\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 - K\Delta \end{bmatrix} \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} + \begin{bmatrix} \frac{\Delta^2}{2} \\ \Delta \end{bmatrix} \ddot{x}_{\tau_0} \tag{3.115}$$

So:

$$\mathbb{E}\left\{\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix}\right\} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 - K\Delta \end{bmatrix} \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} \tag{3.116}$$

By again assuming that the unknown acceleration has variance $\frac{\sigma^2}{\Delta}$, one then arrives at a covariance matrix, Q:

$$Q = \begin{bmatrix} \frac{\Delta^2}{2} \\ \Delta \end{bmatrix} \begin{bmatrix} \frac{\Delta^2}{2} & \Delta \end{bmatrix} \frac{\sigma^2}{\Delta} = \begin{bmatrix} \frac{\Delta^3}{4} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma^2 \tag{3.117}$$

Note that when using the 'Jerk' model, the covariance is unchanged from the case when drag is not present and only the mean is modified. The modified mean is equivalent to the first terms in the Taylor series for the result of using Itô calculus.

3.7.4 Application of Laplace Transforms

In the case of this model:

$$\Phi = \begin{bmatrix} 0 & 1 \\ 0 & -K \end{bmatrix} \tag{3.118}$$

$$\phi\left(\begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix}\right) = \Phi \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} = \begin{bmatrix} \dot{x}_{\tau_0} \\ -K\dot{x}_{\tau_0} \end{bmatrix} \tag{3.119}$$

$$\mathbb{E}\left[\mathbf{v}_\tau {\mathbf{v}_\tau}^T\right] = \begin{bmatrix} 0 & 0 \\ 0 & \sigma^2 \end{bmatrix} \tag{3.120}$$

so:

$$e^{\Phi\Delta} = \mathcal{L}^{-1}\left(\begin{bmatrix} s & -1 \\ 0 & s+K \end{bmatrix}^{-1}\right) = \mathcal{L}^{-1}\left(\begin{bmatrix} \frac{1}{s} & \frac{1}{s(s+K)} \\ 0 & \frac{1}{(s+K)} \end{bmatrix}^{-1}\right) = \begin{bmatrix} 1 & \frac{1}{K}\left(1-e^{-K\Delta}\right) \\ 0 & e^{-K\Delta} \end{bmatrix} \tag{3.121}$$

Then:

$$\mathbf{b} = \int_{\tau=\tau_0}^{\tau_0+\Delta} \begin{bmatrix} 1 & \frac{1}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right) \\ 0 & e^{-K(\tau_0+\Delta-\tau)} \end{bmatrix} d\tau \begin{bmatrix} \dot{x}_{\tau_0} \\ -K\dot{x}_{\tau_0} \end{bmatrix} \tag{3.122}$$

$$= \begin{bmatrix} \Delta & \frac{1}{K}\Delta + \frac{1}{K^2}\left(e^{-K\Delta}-1\right) \\ 0 & \frac{1}{K}\left(1-e^{-K\Delta}\right) \end{bmatrix} \begin{bmatrix} \dot{x}_{\tau_0} \\ -K\dot{x}_{\tau_0} \end{bmatrix} \tag{3.123}$$

$$= \begin{bmatrix} \frac{1}{K}\left(1-e^{-K\Delta}\right)\dot{x}_{\tau_0} \\ \left(e^{-K\Delta}-1\right)\dot{x}_{\tau_0} \end{bmatrix} \tag{3.124}$$

So:

$$\mathbb{E}\left\{\begin{bmatrix} x_{\tau_0+\Delta} \\ \dot{x}_{\tau_0+\Delta} \end{bmatrix}\right\} = \begin{bmatrix} x_{\tau_0} \\ \dot{x}_{\tau_0} \end{bmatrix} + \begin{bmatrix} \frac{1}{K}\left(1-e^{-K\Delta}\right)\dot{x}_{\tau_0} \\ \left(e^{-K\Delta}-1\right)\dot{x}_{\tau_0} \end{bmatrix} \tag{3.125}$$

$$= \begin{bmatrix} x_{\tau_0} + \frac{1}{K}\left(1-e^{-K\Delta}\right)\dot{x}_{\tau_0} \\ e^{-K\Delta}\dot{x}_{\tau_0} \end{bmatrix} \tag{3.126}$$

$$\tag{3.127}$$

which is the same form as derived using Itô calculus. Q can then be derived:

$$Q = \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} \begin{bmatrix} 0 & 0 \\ 0 & \sigma^2 \end{bmatrix} \left(e^{\Phi(\tau_0+\Delta-\tau)}\right)^T d\tau \tag{3.128}$$

$$= \int_{\tau=\tau_0}^{\tau_0+\Delta} \begin{bmatrix} 1 & \frac{1}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right) \\ 0 & e^{-K(\tau_0+\Delta-\tau)} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \sigma^2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \frac{1}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right) & e^{-K(\tau_0+\Delta-\tau)} \end{bmatrix} d\tau \tag{3.129}$$

$$= \sigma^2 \int_{\tau=\tau_0}^{\tau_0+\Delta} \begin{bmatrix} \frac{1}{K^2}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right)^2 & \frac{1}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right)e^{-K(\tau_0+\Delta-\tau)} \\ \frac{1}{K}\left(1-e^{-K(\tau_0+\Delta-\tau)}\right)e^{-K(\tau_0+\Delta-\tau)} & \left(e^{-K(\tau_0+\Delta-\tau)}\right)^2 \end{bmatrix} d\tau \tag{3.130}$$

$$\tag{3.131}$$

which is the same set of integrals as were solved when considering Itô calculus.

3.7.5 Discussion

Again, the use of Itô calculus and Laplace transforms both result in the same model and again the 'Jerk' model gives rise to a slightly simpler model.

3.8 Two Dimensional Constant Turn Rate Model

3.8.1 Model

At time, τ, the a target moves in a two dimensional space with position defined by x_τ and y_τ. The target moves according to a velocity, V_τ, and a heading, θ_τ. The heading changes at a rate $\dot{\theta}_\tau$. The dynamics are near deterministic but with a continuous time input ω_τ^V and $\omega_\tau^{\dot{\theta}}$.

$$\begin{bmatrix} \dot{x}_\tau \\ \dot{y}_\tau \\ \dot{V}_\tau \\ \dot{\theta}_\tau \\ \ddot{\theta}_\tau \end{bmatrix} = \begin{bmatrix} V_\tau \cos\theta_\tau \\ V_\tau \sin\theta_\tau \\ 0 \\ \dot{\theta}_\tau \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega_\tau^V \\ 0 \\ \omega_\tau^{\dot{\theta}} \end{bmatrix} \tag{3.132}$$

3.8.2 Use of Itô Calculus

The system can be described by the following set of coupled differential equations:

$$\begin{bmatrix} dx_\tau \\ dy_\tau \\ dV_\tau \\ d\theta_\tau \\ d\dot{\theta}_\tau \end{bmatrix} = \begin{bmatrix} V_\tau \cos\theta_\tau \\ V_\tau \sin\theta_\tau \\ 0 \\ \dot{\theta}_\tau \\ 0 \end{bmatrix} d\tau + \begin{bmatrix} 0 \\ 0 \\ \sigma_V dw_\tau^V \\ 0 \\ \sigma_{\dot{\theta}} dw_\tau^{\dot{\theta}} \end{bmatrix} \tag{3.133}$$

where $dw_\tau^{\dot{\theta}}$ and dw_τ^V are mutually uncorrelated. These stochastic differential equations are nonlinear, but can be linearised, in which case:

$$\Phi = \begin{bmatrix} 0 & 0 & \cos\theta_\tau & -V_\tau \sin\theta_\tau & 0 \\ 0 & 0 & \sin\theta_\tau & V_\tau \cos\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.134}$$

and $\frac{d^2\phi_k(x)}{dx_{\tau_0} dx_{\tau_0}{}^T}$ is a matrix of zeros for $k = 3, 4, 5$ and:

$$\frac{d^2\phi_1(\mathbf{x}_\tau)}{dx_\tau dx_\tau{}^T} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\sin\theta_\tau & 0 \\ 0 & 0 & -\sin\theta_\tau & -V_\tau \cos\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.135}$$

$$\frac{d^2\phi_2(\mathbf{x}_\tau)}{dx_\tau dx_\tau{}^T} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos\theta_\tau & 0 \\ 0 & 0 & \cos\theta_\tau & -V_\tau \sin\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.136}$$

such that M (which is defined in (3.7 and is the vector that defines the deterministic effect resulting from the presence of random perturbations) is a vector of zeros when evaluated at any x. This is an artifact resulting from the fact that the stochasticity is being input to the rate of change of heading and the speed, but the deterministic behaviour resulting from the presence of this random input only manifests itself when these processes are integrated through time. However, if we consider a fixed value of M to be used over some time-interval in a discretisation context, then it is possible to consider a small change dM and so, rather than consider the ith column of H_k, $H^{i,k} = \frac{d^2\phi_k(x_\tau)}{dx_{i,\tau} dx_\tau}$ to be constant, Itô's formula can be used:

$$dH^{i,k} = \Psi dx + N d\tau \tag{3.137}$$

where

$$\Psi_{i'j'} = \frac{dH^{i,k}_{i'}}{dx_{j'}} \tag{3.138}$$

and

$$N_l = \frac{1}{2}\text{trace}\left(\sigma\sigma^T J_l\right) \tag{3.139}$$

where J_l has elements

$$J_l = \left(\frac{d^2 H^{i,k}}{dx_{i'} dx_{j'}}\right) \tag{3.140}$$

and where $H_l^{i,k}$ is the lth element of $H^{i,k}$. This then results in a need to consider up to the fourth order differentials of $\phi(x_\tau)$, but most of these are zero except:

$$\frac{d^3\phi_1(x_\tau)}{d\theta_\tau^3} = V_\tau \sin\theta_\tau \tag{3.141}$$
$$\frac{d^3\phi_1(x_\tau)}{d\theta_\tau^2 dV_\tau} = -\cos\theta_\tau \tag{3.142}$$
$$\frac{d^3\phi_2(x_\tau)}{d\theta_\tau^3} = -V_\tau \cos\theta_\tau \tag{3.143}$$
$$\frac{d^3\phi_2(x_\tau)}{d\theta_\tau^2 dV_\tau} = -\sin\theta_\tau \tag{3.144}$$
$$\frac{d^4\phi_1(x_\tau)}{d\theta_\tau^4} = V_\tau \cos\theta_\tau \tag{3.145}$$
$$\frac{d^4\phi_1(x_\tau)}{d\theta_\tau^3 dV_\tau} = \sin\theta_\tau \tag{3.146}$$
$$\frac{d^4\phi_2(x_\tau)}{d\theta_\tau^4} = V_\tau \sin\theta_\tau \tag{3.147}$$
$$\frac{d^4\phi_2(x_\tau)}{d\theta_\tau^3 dV_\tau} = -\cos\theta_\tau \tag{3.148}$$

where it has been used that $\frac{d^2\phi(x_\tau)(x)}{dV_\tau d\theta_\tau} = \frac{d^2\phi(x_\tau)}{d\theta_\tau dV_\tau}$.

It transpires that then N is a matrix of zeros but that:

$$dH_1^{i,k} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\cos\theta_\tau & 0 \\ 0 & 0 & -\cos\theta_\tau & V_\tau \sin\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} dx_\tau = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\cos\theta_\tau & 0 \\ 0 & 0 & -\cos\theta_\tau & V_\tau \sin\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_\tau \cos\theta_\tau d\tau \\ V_\tau \sin\theta_\tau d\tau \\ \sigma_V dw_\tau^V \\ \dot{\theta}\tau d\tau \\ \sigma_{\dot{\theta}} dw_\tau^{\dot{\theta}} \end{bmatrix} \tag{3.149}$$

$$= \begin{bmatrix} 0 \\ 0 \\ -\cos\theta_\tau \dot{\theta}_\tau d\tau \\ V_\tau \sin\theta_\tau \dot{\theta}\tau d\tau - \cos\theta_\tau \sigma_V dw_\tau^V \\ 0 \end{bmatrix} \tag{3.150}$$

$$dH_2^{i,k} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\sin\theta_\tau & 0 \\ 0 & 0 & -\sin\theta_\tau & -V_\tau \cos\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} dx_\tau = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\sin\theta_\tau & 0 \\ 0 & 0 & -\sin\theta_\tau & -V_\tau \cos\theta_\tau & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_\tau \cos\theta_\tau d\tau \\ V_\tau \sin\theta_\tau d\tau \\ \sigma_V dw_\tau^V \\ \dot{\theta}\tau d\tau \\ \sigma_{\dot{\theta}} dw_\tau^{\dot{\theta}} \end{bmatrix} \tag{3.151}$$

$$= \begin{bmatrix} 0 \\ 0 \\ -\sin\theta_\tau \dot{\theta}_\tau d\tau \\ -V_\tau \cos\theta_\tau \dot{\theta}\tau d\tau - \sin\theta_\tau \sigma_V dw_\tau^V \\ 0 \end{bmatrix} \tag{3.152}$$

It is then possible to calculate M_τ:

$$M_\tau - M_{\tau_0} = M_\tau = \tilde{M}(\tau - \tau_0) \tag{3.153}$$

where

$$\tilde{M} = \begin{bmatrix} -\sigma_{\dot{\theta}}^2 \cos\theta_{\tau_0} \dot{\theta}_{\tau_0} \\ -\sigma_{\dot{\theta}}^2 \sin\theta_{\tau_0} \dot{\theta}_{\tau_0} \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.154}$$

So:

$$d\phi(x_\tau) = \Phi dx + \tilde{M}(\tau - \tau_0) d\tau \tag{3.155}$$

and:

$$dx_\tau \approx \left(\phi(x_{\tau_0}) + \Phi(x_\tau - x_{\tau_0}) + \tilde{M} \frac{1}{2} (\tau - \tau_0)^2 \right) d\tau + \sigma dw_\tau \tag{3.156}$$

such that:

$$x_{\tau_0+\Delta} - x_{\tau_0} = \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} d\tau \phi\left(x_{\tau_0}\right) + \frac{1}{2}\int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}\left(\tau-\tau_0\right)^2 d\tau \tilde{M} + \int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} \sigma dw_\tau \tag{3.157}$$

It then remains to calculate $e^{\Phi\Delta}$, which can be done using Laplace transforms as follows:

$$e^{\Phi\tau} = \mathcal{L}^{-1}\left(\left(sI-\Phi\right)^{-1}\right) \tag{3.158}$$

$$= \mathcal{L}^{-1}\left(\left[\begin{array}{ccccc} s & 0 & -\cos\theta_{\tau_0} & V_{\tau_0}\sin\theta_{\tau_0} & 0 \\ 0 & s & -\sin\theta_{\tau_0} & -V_{\tau_0}\cos\theta_{\tau_0} & 0 \\ 0 & 0 & s & 0 & 0 \\ 0 & 0 & 0 & s & -1 \\ 0 & 0 & 0 & 0 & s \end{array}\right]^{-1}\right) \tag{3.159}$$

$$= \mathcal{L}^{-1}\left(\left[\begin{array}{ccccc} \frac{1}{s} & 0 & \frac{1}{s^2}\cos\theta_{\tau_0} & -\frac{1}{s^2}V_{\tau_0}\sin\theta_{\tau_0} & -\frac{1}{s^3}V_{\tau_0}\sin\theta_{\tau_0} \\ 0 & \frac{1}{s} & \frac{1}{s^2}\sin\theta_{\tau_0} & \frac{1}{s^2}V_{\tau_0}\cos\theta_{\tau_0} & \frac{1}{s^3}V_{\tau_0}\cos\theta_{\tau_0} \\ 0 & 0 & \frac{1}{s} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{s} & \frac{1}{s^2} \\ 0 & 0 & 0 & 0 & \frac{1}{s} \end{array}\right]\right) \tag{3.160}$$

$$= \left[\begin{array}{ccccc} 1 & 0 & \cos\theta_{\tau_0}\tau & -V_{\tau_0}\sin\theta_{\tau_0}\tau & -V_{\tau_0}\sin\theta_{\tau_0}\frac{\tau^2}{2} \\ 0 & 1 & \sin\theta_{\tau_0}\tau & V_{\tau_0}\cos\theta_{\tau_0}\tau & V_{\tau_0}\cos\theta_{\tau_0}\frac{\tau^2}{2} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & \tau \\ 0 & 0 & 0 & 0 & 1 \end{array}\right] \tag{3.161}$$

so:

$$\int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)} d\tau \phi\left(x_{\tau_0}\right) = \left[\begin{array}{c} \Delta V_{\tau_0}\cos\theta_{\tau_0} - \frac{\Delta^2}{2}V_{\tau_0}\sin\theta_{\tau_0}\phi_{\tau_0} \\ \Delta V_{\tau_0}\sin\theta_{\tau_0} + \frac{\Delta^2}{2}V_{\tau_0}\cos\theta_{\tau_0}\phi_{\tau_0} \\ 0 \\ \Delta\phi_{\tau_0} \\ 0 \end{array}\right] \tag{3.162}$$

$$\frac{1}{2}\int_{\tau=\tau_0}^{\tau_0+\Delta} e^{\Phi(\tau_0+\Delta-\tau)}\left(\tau-\tau_0\right)^2 d\tau \tilde{M} = \frac{1}{6}\left[\begin{array}{c} -\Delta^3\sigma_{\dot{\theta}}^2\cos\theta_{\tau_0}\dot{\theta}_{\tau_0} \\ -\Delta^3\sigma_{\dot{\theta}}^2\sin\theta_{\tau_0}\dot{\theta}_{\tau_0} \\ 0 \\ 0 \\ 0 \end{array}\right] \tag{3.163}$$

and:

$$Q = \begin{bmatrix} Q_{11} & Q_{12} & \sigma_V^2 c\frac{\Delta^2}{2} & -\sigma_\phi^2 V_{\tau_0} s\frac{\Delta^4}{8} & -\sigma_\phi^2 V_{\tau_0} s\frac{\Delta^3}{6} \\ Q_{12} & Q_{22} & \sigma_V^2 s\frac{\Delta^2}{2} & \sigma_\phi^2 V_{\tau_0} c\frac{\Delta^4}{8} & \sigma_\phi^2 V_{\tau_0} c\frac{\Delta^3}{6} \\ \sigma_V^2 c\frac{\Delta^2}{2} & \sigma_V^2 s\frac{\Delta^2}{2} & \sigma_V^2\Delta & 0 & 0 \\ -\sigma_\phi^2 V_{\tau_0} s\frac{\Delta^4}{8} & \sigma_\phi^2 V_{\tau_0} c\frac{\Delta^4}{8} & 0 & \sigma_\phi^2\frac{\Delta^3}{3} & \sigma_\phi^2\frac{\Delta^2}{2} \\ -\sigma_\phi^2 V_{\tau_0} s\frac{\Delta^3}{6} & \sigma_\phi^2 V_{\tau_0} c\frac{\Delta^3}{6} & 0 & \sigma_\phi^2\frac{\Delta^2}{2} & \sigma_\phi^2\Delta \end{bmatrix} \tag{3.164}$$

where

$$Q_{11} = \sigma_V^2 c^2\frac{\Delta^3}{3} + \sigma_\phi^2 V_{\tau_0}^2 s^2\frac{\Delta^5}{20} \tag{3.165}$$

$$Q_{12} = \left(\sigma_V^2\frac{\Delta^3}{3} - \sigma_\phi^2 V_{\tau_0}^2\frac{\Delta^5}{20}\right) cs \tag{3.166}$$

$$Q_{22} = \sigma_V^2 s^2\frac{\Delta^3}{3} + \sigma_\phi^2 V_{\tau_0}^2 c^2\frac{\Delta^5}{20} \tag{3.167}$$

and where

$$c = \cos\theta_{\tau_0} \tag{3.168}$$

$$s = \sin\theta_{\tau_0} \tag{3.169}$$

Pulling all this together:

$$x_{\tau_0+\Delta} - x_{\tau_0} = \begin{bmatrix} \Delta V_{\tau_0}\cos\theta_{\tau_0} - \frac{\Delta^2}{2}V_{\tau_0}\sin\theta_{\tau_0}\dot\theta_{\tau_0} \\ \Delta V_{\tau_0}\sin\theta_{\tau_0} + \frac{\Delta^2}{2}V_{\tau_0}\cos\theta_{\tau_0}\dot\theta_{\tau_0} \\ 0 \\ \Delta\dot\theta_{\tau_0} \\ 0 \end{bmatrix} - \frac{1}{6}\begin{bmatrix} \Delta^3\sigma_{\dot\theta}^2\cos\theta_{\tau_0}\dot\theta_{\tau_0} \\ \Delta^3\sigma_{\dot\theta}^2\sin\theta_{\tau_0}\dot\theta_{\tau_0} \\ 0 \\ 0 \\ 0 \end{bmatrix} + \omega_{\tau_0} \tag{3.170}$$

The first term on the right hand side in (3.170) is approximately the same as that used in the literature[64]:

$$\begin{bmatrix} V_{\tau_0}\cos\theta_{\tau_0}\Delta - V_{\tau_0}\sin\theta_{\tau_0}\dot\theta_{\tau_0}\frac{\Delta^2}{2} \\ V_{\tau_0}\sin\theta_{\tau_0}\Delta + V_{\tau_0}\cos\theta_{\tau_0}\dot\theta_{\tau_0}\frac{\Delta^2}{2} \\ 0 \\ \dot\theta_{\tau_0}\Delta \\ 0 \end{bmatrix} \approx \begin{bmatrix} V_{\tau_0}\cos\left(\theta_{\tau_0} + \frac{\dot\theta_{\tau_0}\Delta}{2}\right) \\ V_{\tau_0}\sin\left(\theta_{\tau_0} + \frac{\dot\theta_{\tau_0}\Delta}{2}\right) \\ 0 \\ \dot\theta_{\tau_0} \\ 0 \end{bmatrix}\Delta \tag{3.171}$$

where the approximation is valid for small $\frac{\dot\theta_{\tau_0}\Delta}{2}$. The second term on the right hand side of (3.170) is analogous to that used to remove the bias associated with Gaussian approximations to the likelihood function for radar measurements[7, 62]. In the same way as with these likelihood functions, the bias is dependent on the angular accuracy; this is intuitively appealing since uncertainty over the rate of change of heading will result in a skewed distribution with a mean biased away from the mode. The covariance is similar, if slightly more complex, in structure to that previously described[64].

3.8.3 Application of 'Jerk' Model

One can use the first term in a Taylor series expansion to linearise the continuous time equation.

$$\begin{bmatrix} x_{\tau_0+\Delta} \\ y_{\tau_0+\Delta} \\ V_{\tau_0+\Delta} \\ \theta_{\tau_0+\Delta} \\ \dot{\theta}_{\tau_0+\Delta} \end{bmatrix} \approx \begin{bmatrix} x_{\tau_0} \\ y_{\tau_0} \\ V_{\tau_0} \\ \theta_{\tau_0} \\ \dot{\theta}_{\tau_0} \end{bmatrix} + \begin{bmatrix} V_{\tau_0}\cos\theta_{\tau_0} \\ V_{\tau_0}\sin\theta_{\tau_0} \\ 0 \\ \dot{\theta}_{\tau_0} \\ 0 \end{bmatrix} \Delta + \begin{bmatrix} 0 \\ 0 \\ \omega^V_{\tau_0} \\ 0 \\ \omega^{\dot{\theta}}_{\tau_0} \end{bmatrix} \Delta \tag{3.172}$$

This gives rise to:

$$\mathbb{E}\left\{ \begin{bmatrix} x_{\tau_0+\Delta} \\ y_{\tau_0+\Delta} \\ V_{\tau_0+\Delta} \\ \theta_{\tau_0+\Delta} \\ \dot{\theta}_{\tau_0+\Delta} \end{bmatrix} \right\} = \begin{bmatrix} x_{\tau_0} + \Delta V_{\tau_0}\cos\theta_{\tau_0} \\ y_{\tau_0} + \Delta V_{\tau_0}\sin\theta_{\tau_0} \\ V_{\tau_0} \\ \theta_{\tau_0} + \dot{\theta}_{\tau_0}\Delta \\ \dot{\theta}_{\tau_0} \end{bmatrix} \tag{3.173}$$

and with suitable definitions for the input process:

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_V^2\Delta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\dot{\theta}}^2\Delta \end{bmatrix} \tag{3.174}$$

This form of the covariance is that used in other discretisations of this system[64], though interestingly not with this mean but with that described earlier.

3.8.4 Application of Laplace Transforms

As should be obvious, the result deduced based on the use of Laplace transforms is that given by (3.170) but without the second term on the right hand side.

3.8.5 Discussion

The system is made up of a number of states that evolve according to a nonlinear set of stochastic differential equations. As a result, approximations are necessary. Using a linearisation of the stochastic differential equations is possible in this example and gives a result that includes an intuitively appealing bias term. Use is made of Laplace transforms to derive the matrix quantities used. A linearisation of the differential equations gives rise to the same approximation but without the bias term. The model previously described in the literature appears to be a combination of the mean derived from the linearisation of the differential equations and a (singular) covariance derived by the use of a simpler 'Jerk' model.

3.9 Three Dimensional Co-ordinated Turn Model

3.9.1 Model

At time, τ, a target has a three dimensional position, $\mathbf{x}_\tau$, a velocity, $\mathbf{v}_\tau$, an acceleration, $\mathbf{a}_\tau$, and an inertial rate of change of acceleration, $\mathbf{u}_\tau$. Again, the dynamics are near deterministic, but have a continuous time input, $\mathbf{w}_t$.

$$\begin{bmatrix} \dot{\mathbf{x}}_\tau \\ \dot{\mathbf{v}}_\tau \\ \dot{\mathbf{a}}_\tau \\ \dot{\mathbf{u}}_\tau \end{bmatrix} = \begin{bmatrix} \mathbf{v}_\tau \\ \mathbf{a}_\tau \\ \dot{\mathbf{a}}_\tau \\ -\alpha\mathbf{u}_\tau \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{w}_\tau \end{bmatrix} \tag{3.175}$$

where

$$\dot{\mathbf{a}}_\tau = \frac{\mathbf{v}_\tau \times \mathbf{a}_\tau}{|\mathbf{v}_\tau|^2} \times (\mathbf{a}_\tau - \mathbf{g}) + \mathbf{T}(\mathbf{v}_\tau, \mathbf{a}_\tau)\,\mathbf{u}_\tau \tag{3.176}$$

$$\mathbf{T}(\mathbf{v}_\tau, \mathbf{a}_\tau) = \begin{bmatrix} \mathbf{t} & \mathbf{b} & \mathbf{n} \end{bmatrix} \tag{3.177}$$

$$\mathbf{t} = \frac{\mathbf{v}}{|\mathbf{v}|} \tag{3.178}$$

$$\mathbf{b} = \frac{(\mathbf{a}-\mathbf{g}) \times \mathbf{v}}{|(\mathbf{a}-\mathbf{g}) \times \mathbf{v}|} \tag{3.179}$$

$$\mathbf{n} = \mathbf{b} \times \mathbf{t} \tag{3.180}$$

To the best knowledge of the author, this is the extent of the description of this model in the literature[13, 64].

3.9.2 Application of Itô Calculus

The system can be described by the following set of coupled differential equations:

$$\begin{bmatrix} d\mathbf{x}_\tau \\ d\mathbf{v}_\tau \\ d\mathbf{a}_\tau \\ d\mathbf{u}_\tau \end{bmatrix} = \begin{bmatrix} \mathbf{v}_\tau \\ \mathbf{a}_\tau \\ \dot{\mathbf{a}}_\tau \\ -\alpha\mathbf{u}_\tau \end{bmatrix} d\tau + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ d\mathbf{w}_\tau \end{bmatrix} \tag{3.181}$$

where the elements of $d\mathbf{w}_\tau$ are mutually uncorrelated as before with diagonal covariance:

$$\mathbb{E}\left\{ d\mathbf{w}_\tau d\mathbf{w}_\tau{}^T \right\} = \boldsymbol{\sigma} d\tau \tag{3.182}$$

These stochastic differential equations are nonlinear, but can be linearised, in which case:

$$\Phi = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} & \mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\alpha\mathbf{I} \end{bmatrix} \tag{3.183}$$

where $\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}$, $\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}}$ and $\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}$ are matrices consisting of the derivatives of the elements of $\dot{\mathbf{a}}_\tau$ with respect to the elements of $\mathbf{v}_\tau$, $\mathbf{a}_\tau$ and $\mathbf{u}_\tau$ respectively.

In much the same way as in the previous example, for M to impact the discretisation, the second order derivatives of $\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}$ with respect to the elements of $\mathbf{a}_\tau$ will be required. This is sufficiently complex that it is not considered here.

3.9.3 Application of 'Jerk' Model

The 'Jerk' model will not be considered here since it will result in a singular covariance matrix and this does not expedite any comparison with the literature in the case of this model.

3.9.4 Application of Laplace Transforms

With Φ defined as previously, calculating $e^{\Phi\tau}$ is then somewhat more complicated than previously, though conceptually simple.

$$e^{\Phi\tau} = \mathcal{L}^{-1}\left((s\mathbf{I}-\Phi)^{-1}\right) = \mathcal{L}^{-1}\begin{bmatrix} s\mathbf{I} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & s\mathbf{I} & -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & s\mathbf{I}-\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} & -\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & (s+\alpha)\,\mathbf{I} \end{bmatrix}^{-1} \tag{3.184}$$

It is possible to decompose the matrix into the product of a block upper triangular and block lower triangular pair of matrices, respectively U and L.

$$s\mathbf{I}-\Phi = LU \tag{3.185}$$

where

$$L = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3.186}$$

$$U = \begin{bmatrix} s\mathbf{I} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & s\mathbf{I} & -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & s\mathbf{I}-\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}}-s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & -\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & (s+\alpha)\,\mathbf{I} \end{bmatrix} \tag{3.187}$$

The inverse of block triangular matrices are easily derived.

$$L^{-1} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3.188}$$

$$U^{-1} = \begin{bmatrix} s^{-1}\mathbf{I} & s^{-2}\mathbf{I} & s^{-2}\mathbf{A}^{-1} & s^{-2}\left(s+\alpha\right)^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & s^{-1}\mathbf{I} & s^{-1}\mathbf{A}^{-1} & s^{-1}\left(s+\alpha\right)^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}^{-1} & \left(s+\alpha\right)^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \left(s+\alpha\right)^{-1}\mathbf{I} \end{bmatrix} \tag{3.189}$$

where

$$\mathbf{A} = s\mathbf{I} - \mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} - s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.190}$$

The inverse of a product of two matrices is the product of the inverse of the two matrices with the order swapped.

$$(s\mathbf{I} - \Phi)^{-1} = U^{-1}L^{-1} \tag{3.191}$$

$$= \begin{bmatrix} s^{-1}\mathbf{I} & s^{-2}\mathbf{I} + s^{-3}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & s^{-2}\mathbf{A}^{-1} & s^{-2}\left(s+\alpha\right)^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & s^{-1}\mathbf{I} + s^{-2}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & s^{-1}\mathbf{A}^{-1} & s^{-1}\left(s+\alpha\right)^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & s^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{A}^{-1} & \left(s+\alpha\right)^{-1}\mathbf{A}^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \left(s+\alpha\right)^{-1}\mathbf{I} \end{bmatrix} \tag{3.192}$$

Since the inverse Laplace transform of the inverse of $\mathbf{A}$ cannot easily be derived, the inverse of $\mathbf{A}$ is approximated using the first term in a Binomial Expansion, for which it is easy to calculate the inverse Laplace transform. This assumes that the primary contribution to the state at time τ is the result of low order differentials of the state at time τ. The author asserts that this is a valid approximation to make since it makes the mathematics tractable and seems intuitively reasonable in realistic scenarios.

$$\mathbf{A}^{-1} = \left(s\mathbf{I} - \mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} - s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right)^{-1} \tag{3.193}$$

$$= s^{-1}\left(\mathbf{I} - s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} - s^{-2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right)^{-1} \tag{3.194}$$

$$\approx s^{-1}\left(\mathbf{I} + s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + s^{-2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right) \tag{3.195}$$

So, $e^{\Phi\Delta}$ can be calculated.

$$e^{\Phi\Delta} = \begin{bmatrix} \mathbf{I} & \Delta\mathbf{I} + \mathbf{C}_3\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{C}_2 & \left(\frac{\Delta_3}{6}\mathbf{I} + \frac{\Delta_4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta_5}{120}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right)\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{I} + \mathbf{C}_2\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{C}_1 & \left(\frac{\Delta_2}{2}\mathbf{I} + \frac{\Delta_3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta_4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right)\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{C}_1\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{C}_0 & \left(\Delta_1\mathbf{I} + \frac{\Delta_2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta_3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right)\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & e^{-\alpha\Delta}\mathbf{I} \end{bmatrix} \tag{3.196}$$

where

$$C_0 = \mathbf{I} + \Delta\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.197}$$

$$C_1 = \Delta\mathbf{I} + \frac{\Delta^2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.198}$$

$$C_2 = \frac{\Delta^2}{2}\mathbf{I} + \frac{\Delta^3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.199}$$

$$C_3 = \frac{\Delta^3}{6}\mathbf{I} + \frac{\Delta^4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^5}{120}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.200}$$

and where the following relations are useful; the approximations are instructive and valid when $\alpha\Delta$ is small.

$$\Delta_1 = \frac{1}{\alpha}\left(1 - e^{-\alpha\Delta}\right) \approx \Delta \tag{3.201}$$

$$\frac{\Delta_2}{2} = \frac{1}{\alpha^2}\left(e^{-\alpha\Delta} - 1 + \alpha\Delta\right) \approx \frac{\Delta^2}{2} \tag{3.202}$$

$$\frac{\Delta_3}{6} = \frac{1}{\alpha^3}\left(1 - e^{-\alpha_\Delta} - \alpha\Delta + \frac{(\alpha\Delta)^2}{2}\right) \approx \frac{\Delta^3}{6} \tag{3.203}$$

$$\frac{\Delta_4}{24} = \frac{1}{\alpha^4}\left(e^{-\alpha_\Delta} - 1 + \alpha\Delta - \frac{(\alpha\Delta)^2}{2} + \frac{(\alpha\Delta)^3}{6}\right) \approx \frac{\Delta^4}{24} \tag{3.204}$$

$$\frac{\Delta_5}{120} = \frac{1}{\alpha^5}\left(1 - e^{-\alpha_\Delta} - \alpha\Delta + \frac{(\alpha\Delta)^2}{2} - \frac{(\alpha\Delta)^3}{6} + \frac{(\alpha\Delta)^4}{24}\right) \approx \frac{\Delta^5}{120} \tag{3.205}$$

So, assuming that the approximations are valid, $\mathbf{b}$ follows.

$$\mathbf{b} \approx \begin{bmatrix} \Delta\mathbf{I} & \frac{\Delta^2}{2}\mathbf{I} + \mathbf{B}_4\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{B}_3 & \mathbf{B}_4\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \Delta\mathbf{I} + \mathbf{B}_3\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{B}_2 & \mathbf{B}_3\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{B}_2\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{B}_1 & \mathbf{B}_2\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Delta\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{v}_t \\ \mathbf{a}_t \\ \dot{\mathbf{a}}_t \\ -\alpha\mathbf{u}_t \end{bmatrix} \tag{3.206}$$

where

$$\mathbf{B}_1 = \Delta\mathbf{I} + \frac{\Delta^2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.207}$$

$$\mathbf{B}_2 = \frac{\Delta^2}{2}\mathbf{I} + \frac{\Delta^3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.208}$$

$$\mathbf{B}_3 = \frac{\Delta^3}{6}\mathbf{I} + \frac{\Delta^4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^5}{120}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.209}$$

$$\mathbf{B}_4 = \frac{\Delta^4}{24}\mathbf{I} + \frac{\Delta^5}{120}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} + \frac{\Delta^6}{720}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} \tag{3.210}$$

To calculate Q with the approximation in (3.195) and so the formulation of $e^{\Phi\Delta}$ in (3.196) would be complicated. Instead, a simpler approximation is used. This is a harsher form of the approximation in (3.195) and implies that the uncertainty input to the system between times τ_0 and $\tau_0 + \Delta$ is the result of low order differentials.

$$\mathbf{A}^{-1} = \left(s\mathbf{I} - \mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}} - s^{-1}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}\right)^{-1} \tag{3.211}$$

$$\approx s^{-1}\mathbf{I} \tag{3.212}$$

So, a different approximation to $e^{\Phi\Delta}$ follows and so an equation for Q.

$$e^{\Phi\Delta} = \begin{bmatrix} \mathbf{I} & \Delta\mathbf{I} + \frac{\Delta^3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \frac{\Delta^2}{2}\mathbf{I} & \frac{\Delta_3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{I} + \frac{\Delta^2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \Delta\mathbf{I} & \frac{\Delta_2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \Delta\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}} & \mathbf{I} & \Delta\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & e^{-\alpha\Delta}\mathbf{I} \end{bmatrix} \tag{3.213}$$

$$Q = \begin{bmatrix} \frac{\Delta^7}{252}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\Delta^6}{72}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\Delta^5}{30}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\bar{\Delta}_4}{24}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma} \\ \frac{\Delta^6}{72}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\Delta^5}{20}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\Delta^4}{8}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\bar{\Delta}_3}{6}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma} \\ \frac{\Delta^5}{30}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\Delta^4}{8}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\Delta^3}{3}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\bar{\Delta}_2}{2}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}\boldsymbol{\sigma} \\ \frac{\bar{\Delta}_4}{24}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\bar{\Delta}_3}{6}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \frac{\bar{\Delta}_2}{2}\boldsymbol{\sigma}\mathbf{F}_{\dot{\mathbf{a}}\mathbf{u}}{}^T & \bar{\Delta}_1\boldsymbol{\sigma} \end{bmatrix} \tag{3.214}$$

where here a similar set of approximations to those used previously are instructive.

$$\bar{\Delta}_1 = \frac{1}{2\alpha}\left(e^{-2\alpha\Delta} - 1\right) \approx \Delta \tag{3.215}$$

$$\frac{\bar{\Delta}_2}{2} = \left(1 - e^{-\alpha\Delta}\left(1 + \alpha\Delta\right)\right) \approx \frac{\Delta^2}{2} \tag{3.216}$$

$$\frac{\bar{\Delta}_3}{6} = \left(1 - e^{-\alpha\Delta}\left(1 + \alpha\Delta + \frac{(\alpha\Delta)^2}{2}\right)\right) \approx \frac{\Delta^3}{6} \tag{3.217}$$

$$\frac{\bar{\Delta}_4}{24} = \left(1 - e^{-\alpha\Delta}\left(1 + \alpha\Delta + \frac{(\alpha\Delta)^2}{2} + \frac{(\alpha\Delta)^3}{6}\right)\right) \approx \frac{\Delta^4}{24} \tag{3.218}$$

The structure of the covariance matrix is as one would expect for a jerk[5] model. The covariance matrix is non-singular. However, the lack of dependence of the terms in the covariance matrix on the differentials $\mathbf{F}_{\dot{\mathbf{a}}\mathbf{v}}$ and $\mathbf{F}_{\dot{\mathbf{a}}\mathbf{a}}$ indicates that the approximation of (3.212) may be overly harsh.

3.9.5 Discussion

This system is sufficiently complex that it is only practical to use the approach based on Laplace transforms. Even when this is done, further approximations are necessary to keep the calculations tractable. However, the result is a system with a non-singular covariance matrix and so amenable to the application of Rao-Blackwellised particle filters[6].

3.10 Conclusions

To use Rao-Blackwellised particle filters or process out-of-sequence measurements, it is essential that the derivation of the discrete time model is high fidelity. To make any approach developed as accessible as possible by the tracking research community, it must also be as simple as possible. To this end, a hierarchy

[5] Here and only here *jerk* is the third derivative of position, the rate of change of acceleration. Elsewhere in this thesis, 'Jerk' refers to a sudden input.

[6] While it is possible to use Rao-Blackwellised particle filters with systems with singular covariance matrices, as stated earlier, the advantages of using Rao-Blackwellisation are greatest when using non-singular covariance matrices. Similarly, while it is possible that specific applications will not permit the use of a Rao-Blackwellised particle filter, there are many applications for which such filters can offer significant improvements in performance.

of systems of stochastic differential equations were discretised using Itô calculus, the 'Jerk' model (often used in the tracking community) and Laplace Transforms. This discretisation process is complicated by both nonlinearity and dimensionality. An approach based on linearisation together with the use of Laplace transforms is the key contribution. This approach closely resembles that developed by other researchers[92, 110] and provides a method for obtaining high fidelity discretisation while maintaining the accessibility of the approach to the tracking community. Using Laplace transforms is also useful when deriving tractable approximations to the matrix exponential. The models derived using the approach were well matched to those already in the literature and included models for which no discrete time model appears to be documented. For one system considered, a slight modification of the approach of previous workers[92, 110] also made it possible to incorporate third order effects.

Both the approach advocated here and that of [92, 110] are based on a linearisation. In a filtering environment, the model would probably be linearised about the mean state estimate. One avenue of future work is to investigate the use of fixed-lag smoothed estimates of the state to re-linearise the differential equations and so improve the approximation to the differential equations and hence tracking performance.

Semi-Markov Models

4

4.1 Introduction

When tracking a manoeuvring target, one needs models that can cater for each of the different regimes that can govern the target's evolution. The transitions between these regimes are often (either explicitly or implicitly) taken to evolve according to a Markov model. At each time epoch there is a probability of being in one discrete state given that the system was in another discrete state. Such Markov switching models result in an exponentially distributed sojourn time; the time for which the system remains in a given discrete state. Semi-Markov models (also known as renewal processes[24]) are a generalisation of Markov models that explicitly model the (discrete state dependent) distribution over sojourn time. At each time epoch there is a probability of being in one discrete state given that the system was in another discrete state and how long it has been in that discrete state. Such models offer the potential to better describe the behaviour of manoeuvring targets.

However, it is believed that the full potential of semi-Markov models has not yet been realised. In [20], sojourns were restricted to end at discrete epochs and filtered mode probabilities were used to deduce the parameters of the time-varying Markov process, equivalent to the semi-Markov process. In [116], the sojourns were taken to be Gamma distributed with integer shape parameters such that the Gamma variate could be expressed as a sum of exponential variates; the semi-Markov model could then be expressed as a (potentially highly dimensional) Markov model. This chapter proposes an approach that does not rely on the sojourn time distribution being of a given form and so is capable of capitalising on all available model fidelity regarding this distribution.

This chapter further considers the problem of both tracking and classifying targets. As discussed in [43], joint tracking and classification is complicated by the fact that sequentially updating a distribution over class membership necessarily results in an accumulation of errors. This is because, when tracking, errors are forgotten. In this context, the capacity to not forget, memory, is a measure of how rapidly the distribution over states becomes increasingly diffuse, making it difficult to predict where the target will be given where it was. Just as the system forgets where it was, so any algorithm that mimics the system forgets any errors that are introduced. So, while one cannot say that if the algorithm forgets any errors, it must converge, if the errors do not accumulate, that will not cause the filter to diverge. In the case of classification, this diffusion does not take place; if one knew the class at one point it would be known

for all future times. As a result, when conducting joint tracking and classification, it becomes not just pragmatically attractive but essential that the tracking process introduces as few errors as possible. This means that the accumulation of errors that necessarily takes place has as little impact as possible on the classification process.

This chapter considers the challenging problem of classifying targets which differ only in terms of their similar sojourn time distributions; the set of dynamic models used to model the different regimes are taken to be the same for all the classes. Were one using a Markov model, all the classes would have the same mean sojourn time and so the same best-fitting Markov model. Hence, it is only possible to classify the targets because semi-Markov models are being used.

Since the semi-Markov models are non-linear and non-Gaussian, the particle filtering methodology is adopted for solving this joint tracking and classification problem. This application of the particle filter is a special case of the generic framework developed concurrently by other researchers[120]. The particle filter represents uncertainty using a set of samples. Here, the samples each represent different hypotheses for the sojourns times and state transitions. Conditional on the sojourn times being known, the system is linear and Gaussian. So, the Kalman filter is used to deduce the parameters of the uncertainty over target state given the hypothesised history of sojourns.

A crucial element of the particle filter is the proposal distribution, the method by which each new sample is proposed from the old samples. If as few errors are to be introduced as possible, it is crucial that this sampling is well matched to the true system. Hence, the set of samples is divided into a number of strata, each of which had a proposal that was well matched to one of the classes. Whatever the proposal distribution, it is possible to calculate the probability of every class. So, to minimise the errors introduced, for each particle (and so hypothesis for the history of state transitions and sojourn times), the probability of all the classes is calculated. So each particle uses a proposal matched to one class, but calculates the probability of the target being a member of every class.

So, the particles are used to estimate the manoeuvres and a Kalman filter used to track the target. The particles are split into strata each of which is well suited to tracking one of the classes and the strata of particles used to classify the target on the basis of the target's manoeuvrability.

This chapter is structured as follows: Section 4.2 begins by introducing the notation and the semi-Markov model structure that is used. Section 4.3 describes how a particle filter can be applied to the hard parts of the problem; the estimation of the semi-Markov process' states. Some theoretical concerns relating to robust joint tracking and identification are discussed in section 4.4. Then, in section 4.5, efficient and robust particle filter architectures are proposed as solutions for the joint tracking and classification problem. Finally, an exemplar problem is considered in section 4.6 and some conclusions drawn in section 4.7.

Notation	Definition
τ_k	continuous time relating to kth measurement
τ_t	continuous time relating to tth sojourn time
t_k	integer index of sojourn prior to kth measurement; so that $\tau_{t_k} \leq \tau_k \leq \tau_{t_k+1}$
k_t	integer index of measurement prior to tth sojourn; so that $\tau_{k_t} \leq \tau_t \leq \tau_{k_t+1}$
s_t	integer index of manoeuvre regime for $\tau_t < \tau < \tau_{t+1}$

Table 4.1: *Table defining notation*

4.2 Model

When using semi-Markov models, there is a need to distinguish between continuous time, the indexing of the measurements and the indexing of the sojourns. Here, continuous time is taken to be τ, measurements are indexed by k and manoeuvre regimes (or *sojourns*) are indexed by t. The continuous time when the kth measurement was received is τ_k. The time of the onset of the sojourn is τ_t. t_k is then the index of the sojourn during which the kth measurement was received. Similarly, k_t is the most recent measurement prior to the onset of the tth sojourn. This is summarised in table 4.1 while figure 4.1 illustrates the relationship between such quantities as $(t_k + 1)$ and t_{k+1}.

The model corresponding to sojourn t is s_t. s_t is a discrete semi-Markov process with transition probabilities $p(s_t|s_{t-1})$ that are known; note that since, at the sojourn end, a transition must occur, so $p(s_t|s_{t-1}) = 0$ if $s_t = s_{t-1}$.

$$p(s_t|s_{t-1}) = p(s_t|s_{1:t-1}) \tag{4.1}$$

where $s_{1:t-1}$ is the history of states for the first to the $(t-1)$th regime and similarly, $y_{1:k}$ will be used to denote the history of measurements up to the kth measurement.

For simplicity, the transition probabilities are here considered invariant with respect to time once it has been determined that a sojourn is to end; ie. $p(s_t|s_{t-1})$ is not a function of τ. The sojourn time distribution that determines the length of time for which the process remains in state s_t is distributed as $g(\tau - \tau_t|s_t)$.

$$p(\tau_{t+1}|\tau_t, s_t) \triangleq g(\tau_{t+1} - \tau_t|s_t). \tag{4.2}$$

The s_t process governs a continuous time process, x_τ, which given s_t and a state at a time after the start of the sojourn $x_{\tau_{t+1}} > x_{\tau'} > x_{\tau_t}$ has a distribution $f(x_\tau|x_{\tau'}, s_t)$. So, the distribution of x_τ given the initial state at the start of the sojourn and the fact that the sojourn continues to time τ is:

$$p(x_\tau|x_{\tau_t}, s_t, \tau_{t+1} > \tau) \triangleq f(x_\tau|x_{\tau_t}, s_t). \tag{4.3}$$

If x_k is the history of states (in continuous time), then a probabilistic model exists for how each measurement, y_k, is related to the state at the corresponding continuous time:

$$p(y_k|x_k) = p(y_k|x_{\tau_k}). \tag{4.4}$$

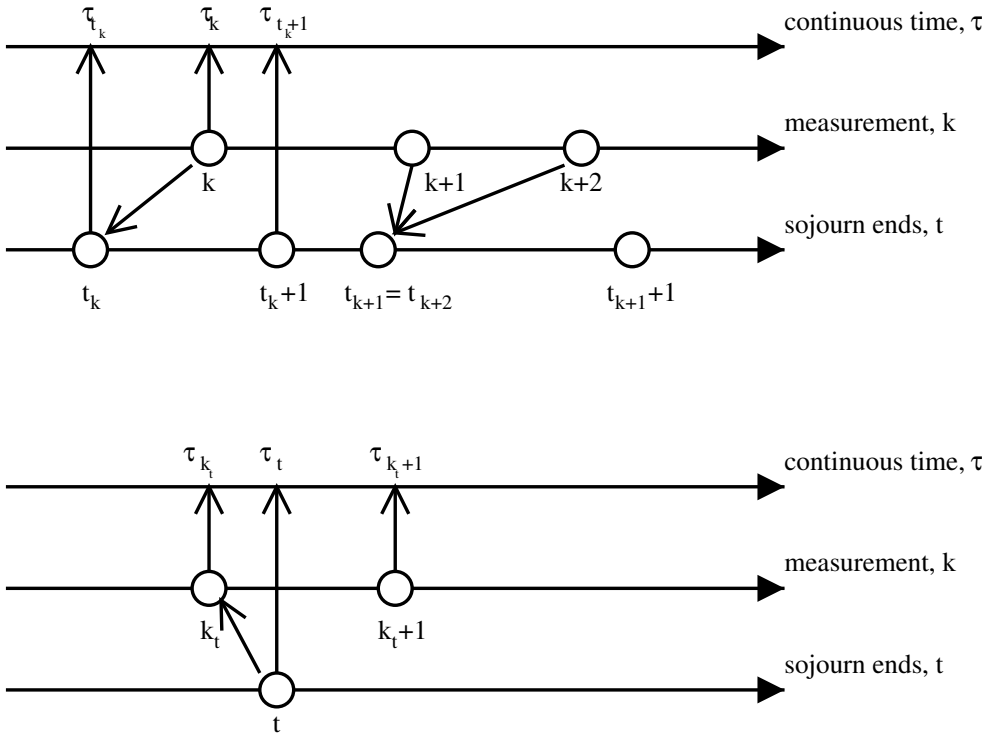


Figure 4.1: *Diagram showing the relationship between continuous time, the time when measurements were received and the time of sojourn ends. The circles represent the receipt of measurements or the start of a sojourn.*

This formulation makes it straightforward to then form a dynamic model for $s_{1:t_k}$ process and $\tau_{1:t_k}$ as follows:

$$p\left(s_{1:t_k}, \tau_{1:t_k}\right) = \left(\prod_{t'=2}^{t_k} p\left(s_{t'}|s_{t'-1}\right) p\left(\tau_{t'}|\tau_{t'-1}, s_{t-1}\right)\right) p\left(s_1\right) p\left(\tau_1\right) \tag{4.5}$$

where $p\left(s_1\right)$ is the initial prior on the state of the sojourn time (which we later assume to be uniform) and $p\left(\tau_1\right)$ is the prior on the time of the first sojourn end (which we later assume to be a delta function). This can then be made conditional on $s_{1:t_{k-1}}$ and $\tau_{1:t_{k-1}}$, which makes it possible to sample the semi-Markov process' evolution between measurements:

$$p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right) \propto \frac{p\left(s_{1:t_k}, \tau_{1:t_k}\right)}{p\left(s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right)} \tag{4.6}$$

$$= \frac{\left(\prod_{t'=2}^{t_k} p\left(s_{t'}|s_{t'-1}\right) p\left(\tau_{t'}|\tau_{t'-1}, s_{t-1}\right)\right) p\left(s_1\right) p\left(\tau_1\right)}{\left(\prod_{t'=2}^{t_{k-1}} p\left(s_{t'}|s_{t'-1}\right) p\left(\tau_{t'}|\tau_{t'-1}, s_{t-1}\right)\right) p\left(s_1\right) p\left(\tau_1\right)} \tag{4.7}$$

$$= \prod_{t'=t_{k-1}+1}^{t_k} p\left(s_{t'}|s_{t'-1}\right) p\left(\tau_{t'}|\tau_{t'-1}, s_{t-1}\right) \tag{4.8}$$

where $A \setminus B$ is the set A without the elements of the set B. Note that in this case $\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}$ could be the empty set in which case, $p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right) = 1$.

So, it is possible to write the joint distribution of the s_t and x_τ processes and the times of the sojourns, $\tau_{1:t_k}$, up to the time of the kth measurement, τ_k, as:

$$p\left(s_{1:t_k}, x_k, \tau_{1:t_k}|y_{1:k}\right) \propto p\left(s_{1:t_k}\tau_{1:t_k}\right) p\left(x_k, y_{1:k}|s_{1:t_k}, \tau_{1:t_k}\right) \tag{4.9}$$

$$= p\left(s_{1:t_k}\tau_{1:t_k}\right) p\left(x_k|s_{1:t_k}, \tau_{1:t_k}\right) p\left(y_{1:k}|x_k\right) \tag{4.10}$$

$$= p\left(s_{1:t_k}\tau_{1:t_k}\right) p\left(x_{\tau_k}|x_{\tau_{t_k}}, s_{t_k}\right) \left(\prod_{t'=2}^{t_k} p\left(x_{\tau_{t'}}|x_{\tau_{t'-1}}, s_{t'-1}\right)\right) p\left(x_{\tau_1}\right) \prod_{k'=1}^{k} p\left(y_{k'}|x_{\tau_{k'}}\right) \tag{4.11}$$

$$\propto \underbrace{p\left(s_{1:t_{k-1}}, x_{k-1}, \tau_{1:t_{k-1}}|y_{1:k-1}\right)}_{\text{The posterior at } k-1.} \underbrace{p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right)}_{\text{Evolution of semi-Markov model.}}$$

$$\times \underbrace{p\left(y_k|x_{\tau_k}\right)}_{\text{Likelihood}} \underbrace{\frac{p\left(x_{\tau_k}|x_{\tau_{t_k}}, s_{t_k}\right)}{p\left(x_{\tau_{k-1}}|x_{\tau_{t_{k-1}}}, s_{t_{k-1}}\right)}}_{\text{Effect on } x_\tau \text{ of incomplete regimes.}} \underbrace{\left(\prod_{t'=t_{k-1}+1}^{t_k} p\left(x_{\tau_{t'}}|x_{\tau_{t'-1}}, s_{t'-1}\right)\right)}_{\text{Effect on } x_\tau \text{ of sojourns between } k-1 \text{ and } k.} . \tag{4.12}$$

This is a recursive formulation of the problem. The annotations indicate the individual terms' relevance.

4.3 Application of Particle Filtering

Here, an outline of the form of particle filtering used is given so as to provide some context for the subsequent discussion and introduce notation. The reader who is unfamiliar with the subject is referred to the section 2.6 as well as the various tutorials and books available on the subject (two examples of which are [2] and [28]).

A particle filter is used to deduce the sequence of sojourn times, $\tau_{1:t_k}$, and the sequence of transitions, $s_{1:t_k}$, as a set of measurments are received. This is achieved by sampling N times from a proposal distribution of a form that extends the existing set of sojourn times and the s_t process with samples of the sojourns that took place between the previous and the current measurements:

$$\left\{\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}\right\}^i \sim q\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} \mid \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^i, y_k\right) \qquad i = 1 \ldots N. \tag{4.13}$$

A weight is then assigned according to the principle of importance sampling:

$$\bar{w}_k^i = w_{k-1}^i \frac{p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\}^i \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^i \mid \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^i\right) p\left(y_k \middle| \left\{s_{1:t_k}, \tau_{1:t_k}\right\}^i\right)}{q\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\}^i \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^i \mid \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^i, y_k\right)}. \tag{4.14}$$

These *unnormalised* weights are then normalised:

$$w_k^i = \frac{\bar{w}_k^i}{\sum_{i'=1}^N \bar{w}_k^{i'}} \tag{4.15}$$

and estimates of expectations calculated using the (normalised) weighted set of samples. When the weights become skewed, some of the samples dominate these expectations, so the particles are resampled; particles with low weights are probabilistically discarded and particles with high weights are probabilistically replicated in such a way that the expected number of offspring resulting from a given particle is proportional to the particle's weight. This resampling can introduce unnecessary errors. So, it should be used as infrequently as possible. To this end, a threshold can be put on the approximate effective sample size, so that when this effective sample size falls below a predefined threshold, the resampling step is performed. Recall that this approximate effective sample can be calculated as follows:

$$N_{\text{eff}} \approx \frac{1}{\sum_{i=1}^N \left(\bar{w}_k^i\right)^2}. \tag{4.16}$$

It is also possible to calculate the incremental likelihood:

$$p\left(y_k | y_{1:k-1}\right) \approx \frac{1}{N} \sum_{i=1}^N \bar{w}_k^i \tag{4.17}$$

which can be used to calculate the likelihood of the entire data sequence, which will be useful in later sections:

$$p\left(y_{1:k}\right) = p\left(y_1\right) \prod_{k'=2}^k p\left(y_{k'} | y_{1:k'-1}\right) \tag{4.18}$$

where $p\left(y_1\right) \triangleq p\left(y_1 | y_{1:0}\right)$, so can be calculated using (4.17).

4.4 Theoretical Concerns Relating To Joint Tracking and Classification

The proofs of convergence for particle filters rely on the ability of the dynamic models used to forget the errors introduced by the Monte-carlo integration [60]. If errors are forgotten then the errors cannot accumulate and so the algorithm must converge on the true uncertainty relating to the path through the state space.

Conversely, if the system does not forget then errors will accumulate and this will eventually cause the filter to diverge. This applies to sequential algorithms in general, including Kalman filters[1], which accumulate finite precision errors, though such errors are often sufficiently small that such problems rarely arise and have even less rarely been noticed.

For a system to forget, its model needs to involve the states changing with time; it must be *ergodic*[2]. There is then a finite probability of the system being in any state given that it was in any other state at some point in the past; so, it is not possible for the system to get stuck in a state. Models for classification do not have this ergodic property since the class is constant for all time; such models have infinite memory. Approaches to classification (and other long memory problems) have been proposed in the past based on both implicit and explicit modifications of the model that reduce the memory of the system by introducing some dynamics. Here, the emphasis is on using the models in their true form.

However, if the model's state is discrete, as is the case with classification, there is a potential solution described in this context in [43]. The idea is to ensure that all probabilities are calculated based on the classes remaining constant and to run a filter for each class; these filters cannot be reduced in number when the probability passes a threshold if the system is to be robust. In such a case, the overall filter is conditionally ergodic. The approach is similar to that advocated for classification alone whereby different classifiers are used for different classes[5].

The preceding argument relates to the way that the filter forgets errors. This enables the filter to always be able to visit every part of the state space; and the approach advocated makes it possible to recover from a misclassification. However, this does not guarantee that the filter can calculate classification probabilities with any accuracy. The problem is the variation resulting from different realisations of the errors caused in the inference process. In a particle filter context, this variation is the Monte Carlo variation and is the result of having sampled one of many possible different sets of particles at a given time. Put more simply; performing the sampling step twice would not give the same set of samples.

(4.18) means that, if each iteration of the tracker introduces errors, in this environment, the classification errors necessarily accumulate[3]. There is nothing that can be done about this. All that can be

[1] It is well documented that Extended Kalman filters can accumulate linearisation errors which can cause filter divergence, but here the discussion relates to Kalman filtering with linear Gaussian distributions such that the Kalman filter is an analytic solution to the problem of describing the pdf.

[2] Technically, it is the optimal filter, rather than the dynamic model that needs to be ergodic. However, here the discussion relates to the model since this aids understanding and, for the cases considered here, the two concepts are equivalent.

[3] Note that the argument here and throughout the surrounding text is aimed at the reader unfamiliar to the literature

done is to attempt to minimise the errors that are introduced such that the inevitable accumulation of errors will not impact performance on a time scale that is of interest.

So, to be able to classify targets based on their dynamic behaviour, all estimates of probabilities must be based on the classes remaining constant for all time and the errors introduced into the filter must be minimised. As a result, classification performance is a good test of algorithmic performance.

4.5 Efficient and Robust Classification

The previous section asserts that to be robust, it is essential to estimate probabilities based on all the classes always remaining constant. However, to be efficient, the filter should react to the classification estimates and focus its effort on the most probable classes (this could equally be the class with the highest expected cost according to some non-uniform cost function but this is not considered here).

To resolve these two seemingly contradictory requirements of robustness twinned with efficiency, the structure of the particle filter can be capitalised upon. The particle filter distinguishes between the proposal used to sample the particles' paths and the weights used to reflect the disparity between the proposal and the true posterior. So, it is possible for the proposal to react to the classification probabilities and favour proposals well suited to the more probable classes while calculating the weights for the different classes; this is equivalent to Rao-Blackwellising the discrete distribution over class for each particle.

One could enable the system to react to the classification probabilities while remaining robust to misclassification by each particle sampling the importance function from a set of importance samplers according to the classification probabilities. Each importance sampler would be well-suited to the corresponding class and each particle would calculate the weights with respect to all the classes given its sampled values of the state.

However, here a different architecture is advocated; the particles are divided into strata, such that the different strata each use an importance function well-suited to one of the classes. For any particle in the jth stratum, S_j, and in the context of the application of particle filtering to semi-Markov models, the importance function is then of the form $q\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} \mid \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}, y_k, S_j\right)$. The strata then each have an associated weight and these weights sum to unity across the strata. If each particle calculates the probability of all the classes given its set of hypotheses, then the architecture will be robust. It is then possible to make the architecture efficient by adding a decision logic that reacts to the weights on the strata; one might add and remove strata on the basis of the classification probabilities. The focus here is not on designing such a decision logic, but to propose an architecture that permits the use of such logic.

To use this architecture, it is necessary to manipulate strata of particles and so to be able to calculate the total weight on a class or equally on a stratum. To this end, the relations that enable this to happen are now outlined.

on convergence proofs; the statements therefore hold in the context of tracking, but do not necessarily generalise to more generic applications of particle filters.

The classes are indexed by c, particles indexed by i and the strata indexed by j. The model used to calculate the weights is M and the stratum is S. So, the unnnormalised weight for the ith particle in stratum S_j, using model M_c, is $\bar{w}_k^{(i,j,c)}$.

The weight on a stratum, $p(S_j|y_{1:k})$, can be deduced from:

$$p(S_j|y_{1:k}) \propto p(y_{1:k}|S_j)\, p(S_j) \tag{4.19}$$

where $p(S_j)$ is the (probably uniform) prior across the strata. This leads to the following recursion:

$$p(S_j|y_{1:k}) \propto p(y_k|y_{1:k-1}, S_j)\, p(S_j|y_{1:k-1}) \tag{4.20}$$

where $p(y_k|y_{1:k-1}, S_j)$ can be estimated using a minor modification of (4.17) as follows:

$$p(y_k|y_{1:k-1}, S_j) \approx \sum_{i,c} \bar{w}_k^{(i,j,c)}. \tag{4.21}$$

Similarly, for the classes:

$$p(M_c|y_{1:k}) \propto p(y_k|y_{1:k-1}, M_c)\, p(M_c|y_{1:k-1}) \tag{4.22}$$

where

$$p(M_c|y_{1:k}) = \sum_j p(S_j, M_c|y_{1:k}) = \sum_j p(S_j|y_{1:k})\, p(M_c|S_j, y_{1:k}) \tag{4.23}$$

and

$$p(M_c|S_j, y_{1:k}) \propto \sum_i \bar{w}_k^{(i,j,c)}. \tag{4.24}$$

To implement this recursion, the weights of the classes are normalised such that they sum to unity over the particle in the strata:

$$w_k^{(c|i,j)} \triangleq \frac{\bar{w}_k^{(i,j,c)}}{\bar{w}_k^{(i,j)}} \tag{4.25}$$

where $\bar{w}_k^{(i,j)}$ is the total unnormalised weight of the particle:

$$\bar{w}_k^{(i,j)} \triangleq \sum_c \bar{w}_k^{(i,j,c)}. \tag{4.26}$$

These weights are then normalised such that they sum to unity within each strata:

$$w_k^{(i|j)} \triangleq \frac{\bar{w}_k^{(i,j)}}{\bar{w}_k^{(j)}} \tag{4.27}$$

where $\bar{w}_k^{(j)}$ is the total unnormalised weight of the stratum:

$$\bar{w}_k^{(j)} \triangleq \sum_i \bar{w}_k^{(i,j)}. \tag{4.28}$$

These weights are also normalised such that they sum to unity across the strata:

$$w_k^{(j)} \triangleq \frac{\bar{w}_k^{(j)}}{\sum_i \bar{w}_k^{(i,j)}}. \tag{4.29}$$

The skewness of each stratum is then used to assess whether that stratum has degenerated and so if resampling is necessary for the set of particles in that stratum. This means that the weight relating to M_c for the ith particle within the jth stratum is:

$$w_k^{(i,j,c)} \propto w_k^{(j)} w_k^{(i|j)} w_k^{(c|i,j)}. \tag{4.30}$$

4.6 Example

4.6.1 Model

The classification of targets which differ solely in terms of the semi-Markov model governing the s_t process is considered. The classes have different gamma distributions for their sojourn times but all have the same mean value for the sojourn time, and so the same best fitting Markov model.

The x_τ process is taken to be a constant velocity model, an integrated diffusion process:

$$f\left(x_{\tau+\Delta}|x_\tau, s\right) = \mathcal{N}\left(x_{\tau+\Delta}; A\left(\Delta\right) x_\tau, Q_s\left(\Delta\right)\right) \tag{4.31}$$

where $\mathcal{N}\left(x; m, C\right)$ denotes a Gaussian distribution for x, with mean, m, and covariance, C, and where:

$$A\left(\Delta\right) = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \tag{4.32}$$

$$Q_s\left(\Delta\right) = \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma_s^2 \tag{4.33}$$

where the discrete state, s_t, takes one of two values which differ in terms of σ_s^2; $\sigma_1^2 = 0.001$ and $\sigma_2^2 = 100$.

The measurements are linear Gaussian measurements of position:

$$p\left(y_k|x_{\tau_k}\right) = \mathcal{N}\left(y_k; H x_{\tau_k}, R\right) \tag{4.34}$$

where

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.35}$$

and $R = 0.1$. The measurements are received at regular intervals such that $\tau_k - \tau_{k-1} = 0.5$ for all $k > 1$.

The three classes' sojourn distributions are:

$$g\left(\tau - \tau_t | s_t, M_c\right) = \begin{cases} \mathcal{G}\left(\tau - \tau_t; 2, 5\right) & s_t = 1, c = 1 \\ \mathcal{G}\left(\tau - \tau_t; 10, 1\right) & s_t = 1, c = 2 \\ \mathcal{G}\left(\tau - \tau_t; 50, 0.2\right) & s_t = 1, c = 3 \\ \mathcal{G}\left(\tau - \tau_t; 10, 0.1\right) & s_t = 2, \forall c \end{cases} \tag{4.36}$$

where $\mathcal{G}\left(x; \alpha, \beta\right)$ is a gamma distribution over x, with shape parameter α and scale parameter β. Figure 4.2 shows these different sojourn time distributions. Note that since the mean of the gamma distribution is $\alpha\beta$, all the sojourn distributions for $s_t = 1$ have the same mean. Hence, the exponential

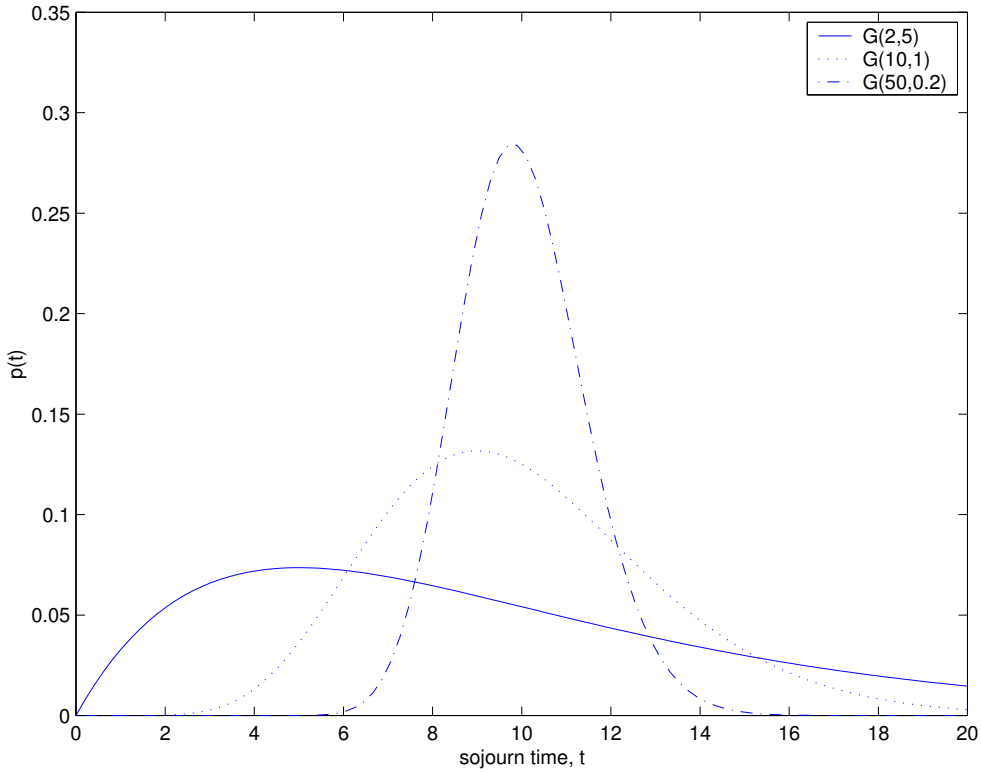


Figure 4.2: *Sojourn time distributions for $s_t = 1$ for the different classes.*

distribution (which only has a single parameter that defines the mean) for all three classes would be the same.

Since there are only two discrete states, the state transition probabilities are simple:

$$p\left(s_t|s_{t-1}\right) = \begin{cases} 0 & s_t = s_{t-1} \\ 1 & s_t \neq s_{t-1} \end{cases}. \tag{4.37}$$

This means that, given the initial discrete state, the sojourn ends define the discrete state sequence.

$p\left(s_1\right)$ is taken to be uniform across the two models and $p\left(\tau_1\right) = \delta\left(\tau_1 - 0\right)$, so it assumed known that there was a transition at time 0. x_0 is initialised at zero as follows:

$$x_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{4.38}$$

4.6.2 Tracking of Manoeuvring Targets

A target from the first class is considered. A Rao-Blackwellised particle filter is used. The particle filter samples the sojourn ends and then, conditional on the sampled sojourn ends and state transitions, uses a Kalman filter to exactly describe the uncertainty relating to x_τ and a discrete distribution over class to exactly describe the classification probabilities (as described previously).

For the proposal in the particle filter, (4.8), the dynamic prior for the s_t process, is used, with a minor

modification:

$$q\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} \mid \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}, y_k\right)$$
$$\triangleq p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}, \tau_{t_k+1} > \tau_k, M_j\right) \tag{4.39}$$
$$= \int p\left(\left\{s_{1:t_k}, \tau_{1:t_k+1}\right\} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}, \tau_{t_k+1} > \tau_k, M_j\right) d\tau_{t_k+1} \tag{4.40}$$

ie. when sampling up to time τ_k, the s_t process is extended to beyond τ_k, but the sample of the final sojourn time is integrated out (so forgotten); the proposal simply samples that the next sojourn is after the time of the measurement, not what time it actually took place. This exploits some structure in the problem since τ_{t_k+1} has no impact on the estimation up to time τ_k and so classification on the basis of $y_{1:k}$. The weight update equation simplifies since the dynamics are used as the proposal:

$$\bar{w}_k^i = w_{k-1}^i p\left(y_k | \left\{s_{1:t_k}, \tau_{1:t_k}\right\}^i\right) \tag{4.41}$$

where $p\left(y_k | \left\{s_{1:t_k}, \tau_{1:t_k}\right\}^i\right)$ can straightforwardly be calculated by a Kalman filter with a time varying process model (with model transitions at the sojourn ends) and measurement updates at the times of the measurements.

Having processed the kth measurement, the ith particle then needs to store the time of the hypothesised last sojourn, $\tau_{t_k}^{(i)}$, the current state, $s_{t_k}^{(i)}$, a mean and covariance for x_{τ_k} and a weight, $w_k^{(i)}$.

25 particles are used[4] and initialised with samples from $p(s_1)$ and $p(\tau_1)$ (so all the same τ_1). Each particles' initial value for the Kalman filter's mean is the true initial state. The initial value for the covariance is then defined as P:

$$P = \begin{bmatrix} 100 & 0 \\ 0 & 10 \end{bmatrix}. \tag{4.42}$$

The weights are all initialised as equal for all the particles. Resampling takes place if the approximate effective sample size given in (4.16) falls below 12.5.

The true trajectory through the x_τ space is shown in figure 4.3. The errors in the estimation of this trajectory conditional on the first particle's path is shown in figure 4.4. The true trajectory though the discrete space is given in figure 4.5. The hypothesis for the trajectory through the discrete space for some of the particles is shown in figure 4.6.

Note that the target has been successfully tracked and that, as a result of the resampling, the particles all have the same hypothesis for the majority of the trajectory through the discrete space, which is well matched (for the most part) to the true trajectory. The diversity of the particles represents the uncertainty over the later part of the state sequence with the particles representing different hypothesised times and numbers of recent regime switches.

[4]This number is smaller than one might use in practical situations, but is chosen to be consistent with the subsequent subsection when the choice makes it possible to illustrate the effect of the accumulation of errors.

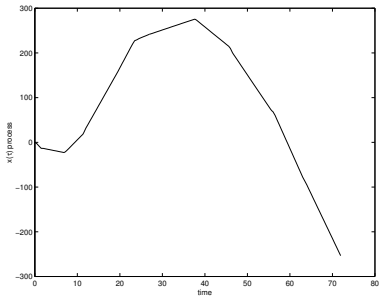


Figure 4.3: *True trajectory for target through x_τ state-space.*

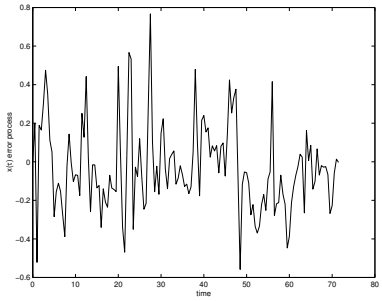


Figure 4.4: *Estimation error in the x_τ state space.*

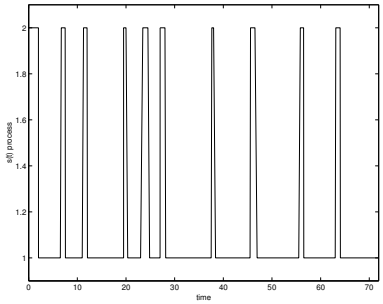
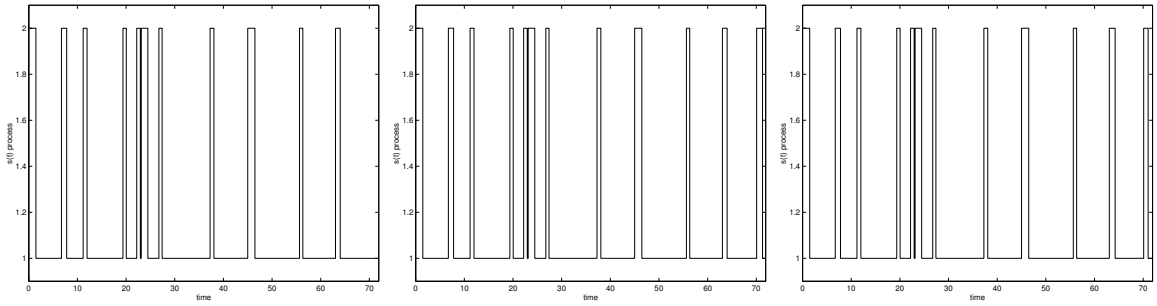
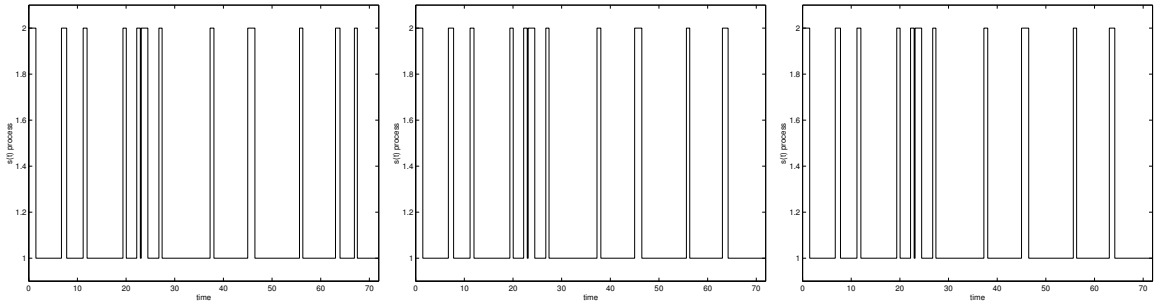


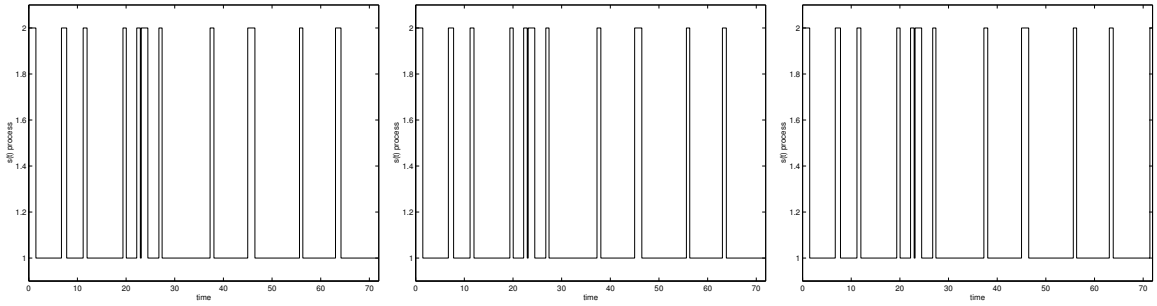
Figure 4.5: *True trajectory for target through s_t state space.*



(a) Particle 1 (b) Particle 2 (c) Particle 3



(d) Particle 4 (e) Particle 5 (f) Particle 6



(g) Particle 7 (h) Particle 8 (i) Particle 9

Figure 4.6: *A subset of the particles' hypothesised trajectories through s_t space.*

4.6.3 Classification on the Basis of Manoeuvrability

The proposals that are well-suited to each class each use the associated class' prior as their proposal:

$$q\left(\left\{s_{1:t_k},\tau_{1:t_k}\right\}\setminus\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\}\mid\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\},y_k,S_j\right) \tag{4.43}$$

$$\triangleq p\left(\left\{s_{1:t_k},\tau_{1:t_k}\right\}\setminus\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\}\mid\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\},M_j\right). \tag{4.44}$$

The weight update equation is then:

$$\bar{w}_k^{(i,j,c)} = w_{k-1}^{(i,j,c)}\frac{p\left(y_k\middle|\left\{s_{1:t_k},\tau_{1:t_k}\right\}^{(i,j)}\right)p\left(\left\{s_{1:t_k},\tau_{1:t_k}\right\}^{(i,j)}\setminus\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\}^{(i,j)}\middle|\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\}^{(i,j)},M_c\right)}{p\left(\left\{s_{1:t_k},\tau_{1:t_k}\right\}^{(i,j)}\setminus\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\}^{(i,j)}\middle|\left\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\right\}^{(i,j)},M_j\right)}. \tag{4.45}$$

Having processed the k measurement, the ith particle in the jth stratum stores the time of the hypothesised last sojourn, $\tau_{t_k}^{(i,j)}$, the current state, $s_{t_k}^{(i,j)}$, a mean and covariance for x_{τ_k}, a weight for each class, $w_k^{(c|i,j)}$, and a weight, $w_k^{(i|j)}$. Each stratum also stores $w_k^{(j)}$.

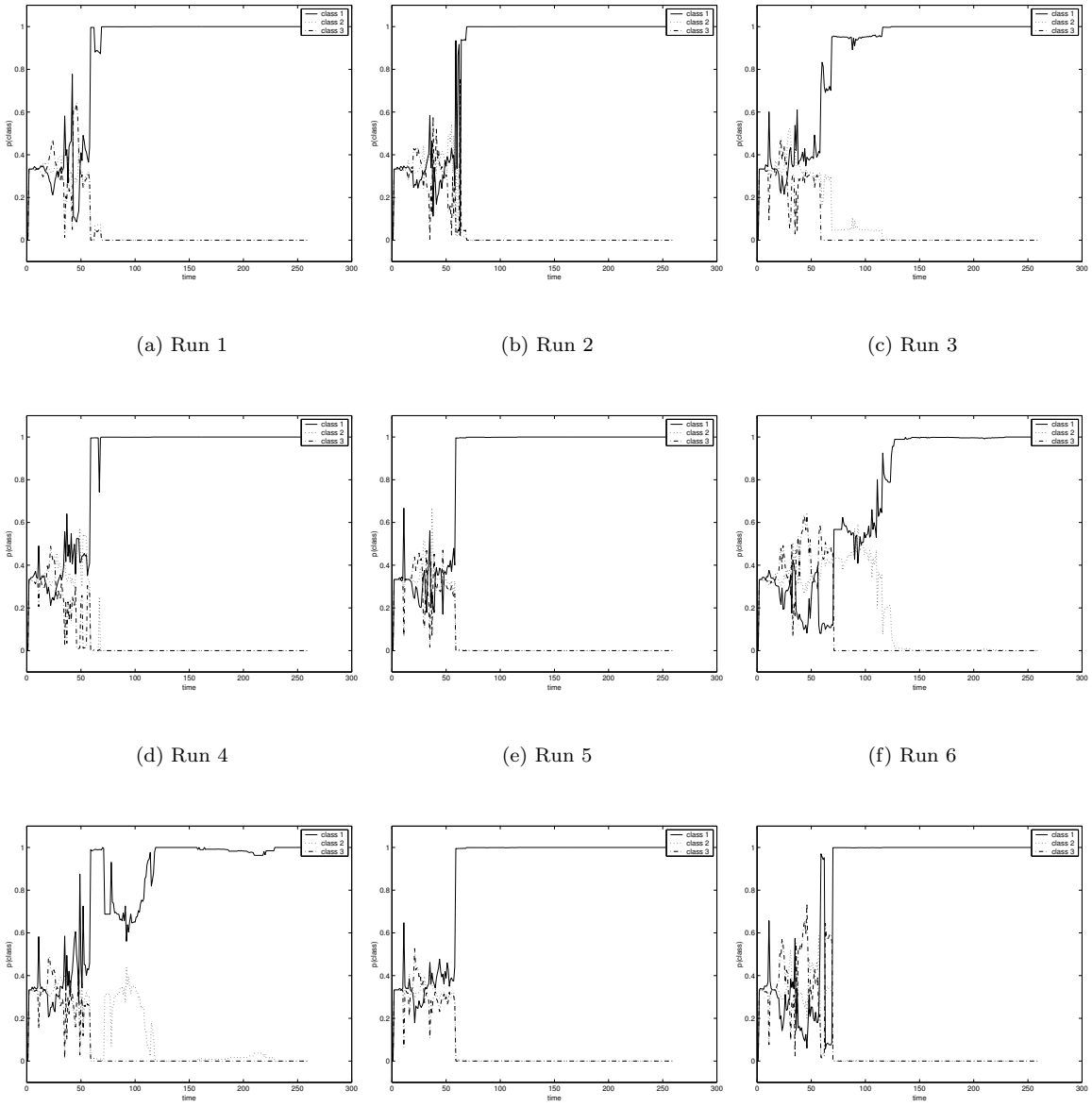
25 particles are used per stratum, each is initialised as previously with a uniform distribution over the classes and with the weights on the strata initialised as being equal. Resampling for a given stratum takes place if the approximate effective sample size given in (4.16) for the stratum falls below 12.5.

Nine runs were conducted with data simulated according to each of the three models. The classification results are shown in figures 4.7, 4.8 and 4.9 for data simulated from models 1, 2 and 3 respectively. In the majority of cases, the classification is correct at the end of the run (such that the classification is based on all the data). It is also evident that there is significant variation between runs; the errors are indeed accumulating. As one might expect, the targets that are of the first class are rapidly classified; the correct classification probabilities rapidly rise to unity and remain there. The classification probabilities for targets of the other two classes change less rapidly and it appears, as one might expect, that it is hardest to correctly classify targets of class 2.

The weights on the different strata are shown in figures 4.10, 4.11 and 4.12 for the same runs and for models 1, 2 and 3 respectively. It is worth noting that in contrast to the classification probabilities, the weights on the strata often change suddenly. It is also worth noting that the stratum with the highest weight will often change without a corresponding change to the classification probabilities; run 7 for class 1 is a clear example of this. Finally, note that the stratum with the highest weight is not as frequently the true class as the classification results; this is particularly noticeable for class 2 for which the classification performance is promising, but for which the stratum with the highest weight appears to rarely be that corresponding with class 2. This implies that the choice of importance function is less important than calculating the probabilities of all the classes for every sample.

4.7 Conclusions

Particle filtering has been applied to the use of semi-Markov models for tracking manoeuvring targets. An architecture has been proposed that enables particle filters to be both robust and efficient when

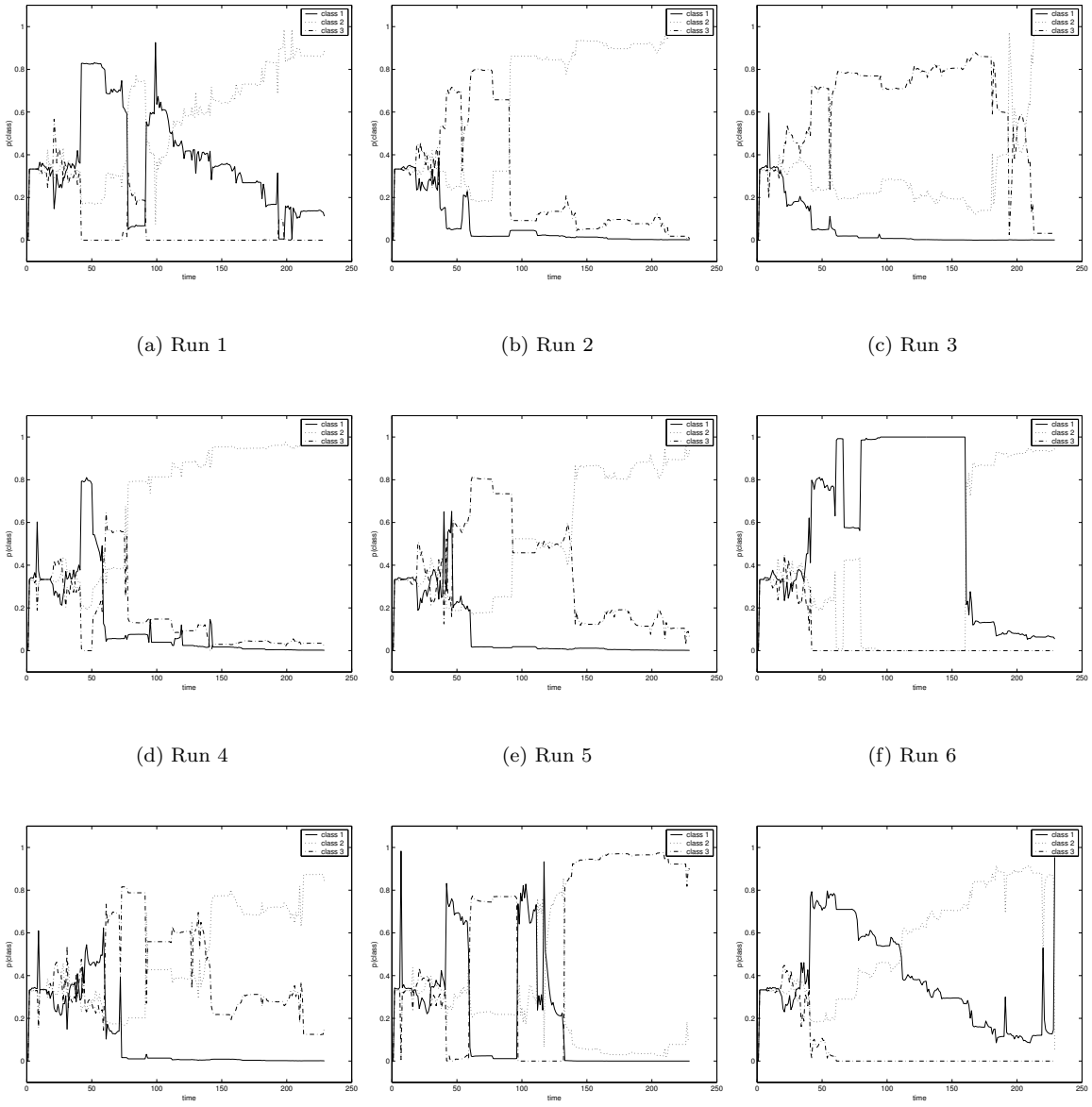


(a) Run 1 (b) Run 2 (c) Run 3

(d) Run 4 (e) Run 5 (f) Run 6

(g) Run 7 (h) Run 8 (i) Run 9

Figure 4.7: *Nine sets of sequential classification results with the true class as class 1.*

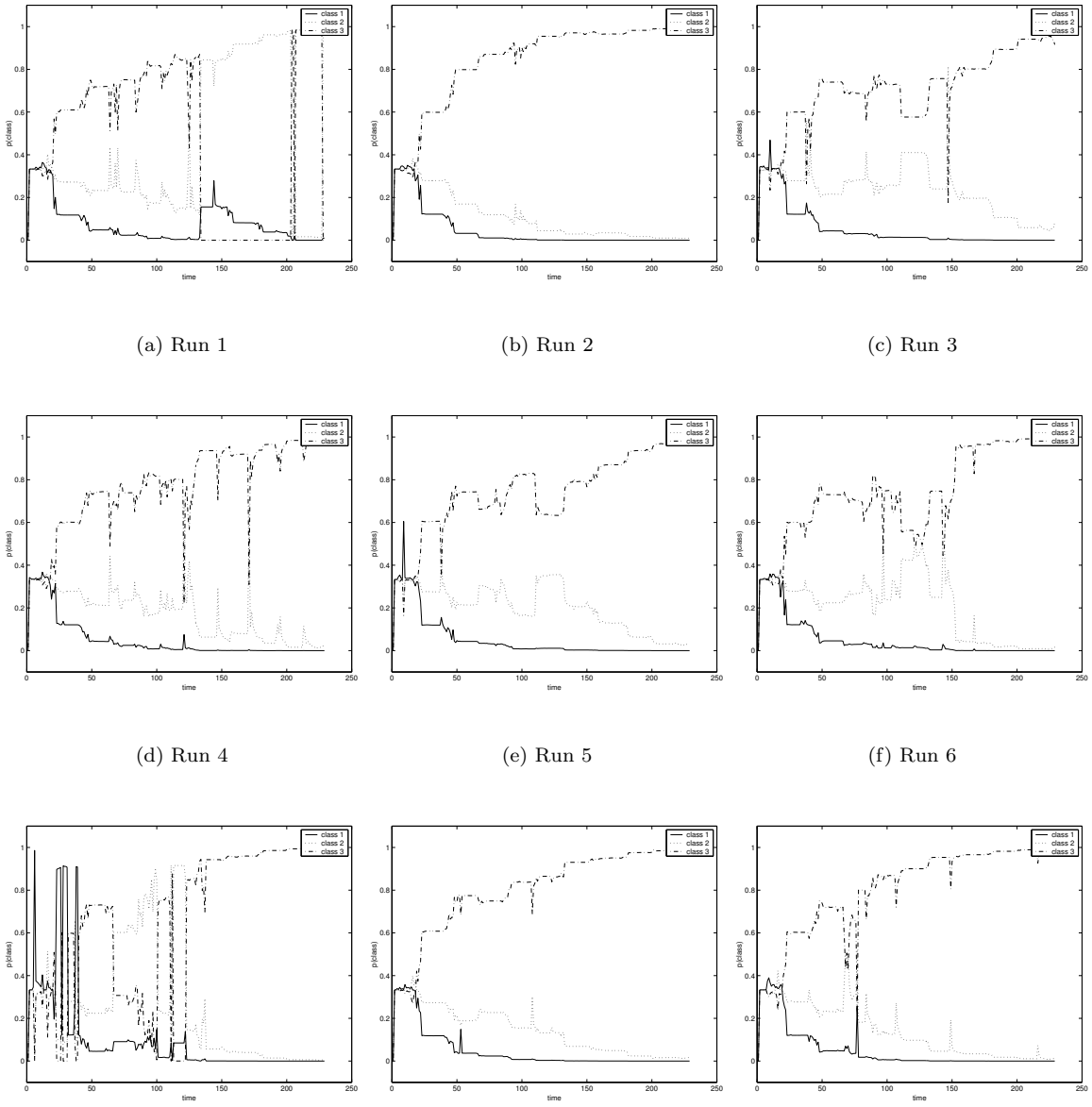


(a) Run 1 (b) Run 2 (c) Run 3

(d) Run 4 (e) Run 5 (f) Run 6

(g) Run 7 (h) Run 8 (i) Run 9

Figure 4.8: *Nine sets of sequential classification results with the true class as class 2.*

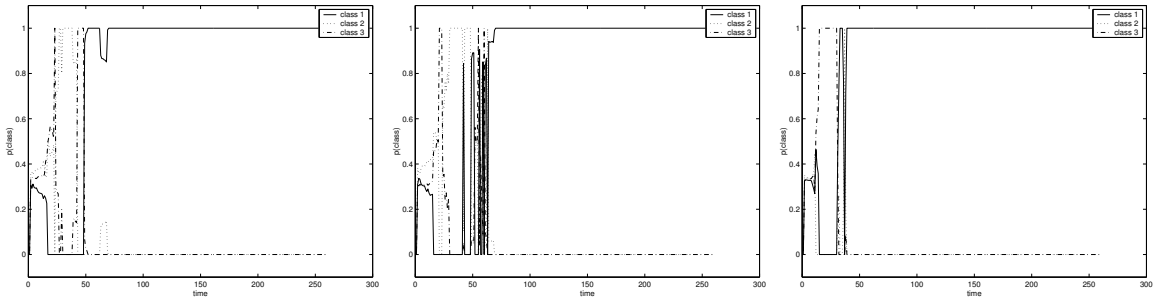


(a) Run 1 (b) Run 2 (c) Run 3

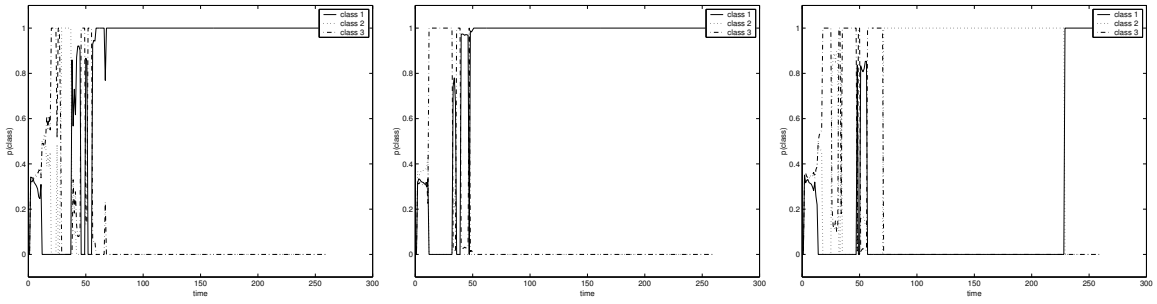
(d) Run 4 (e) Run 5 (f) Run 6

(g) Run 7 (h) Run 8 (i) Run 9

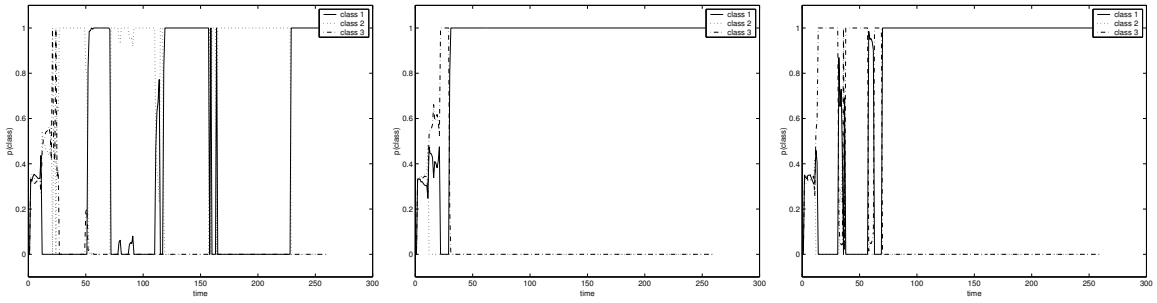
Figure 4.9: *Nine sets of sequential classification results with the true class as class 3.*



(a) Run 1 (b) Run 2 (c) Run 3

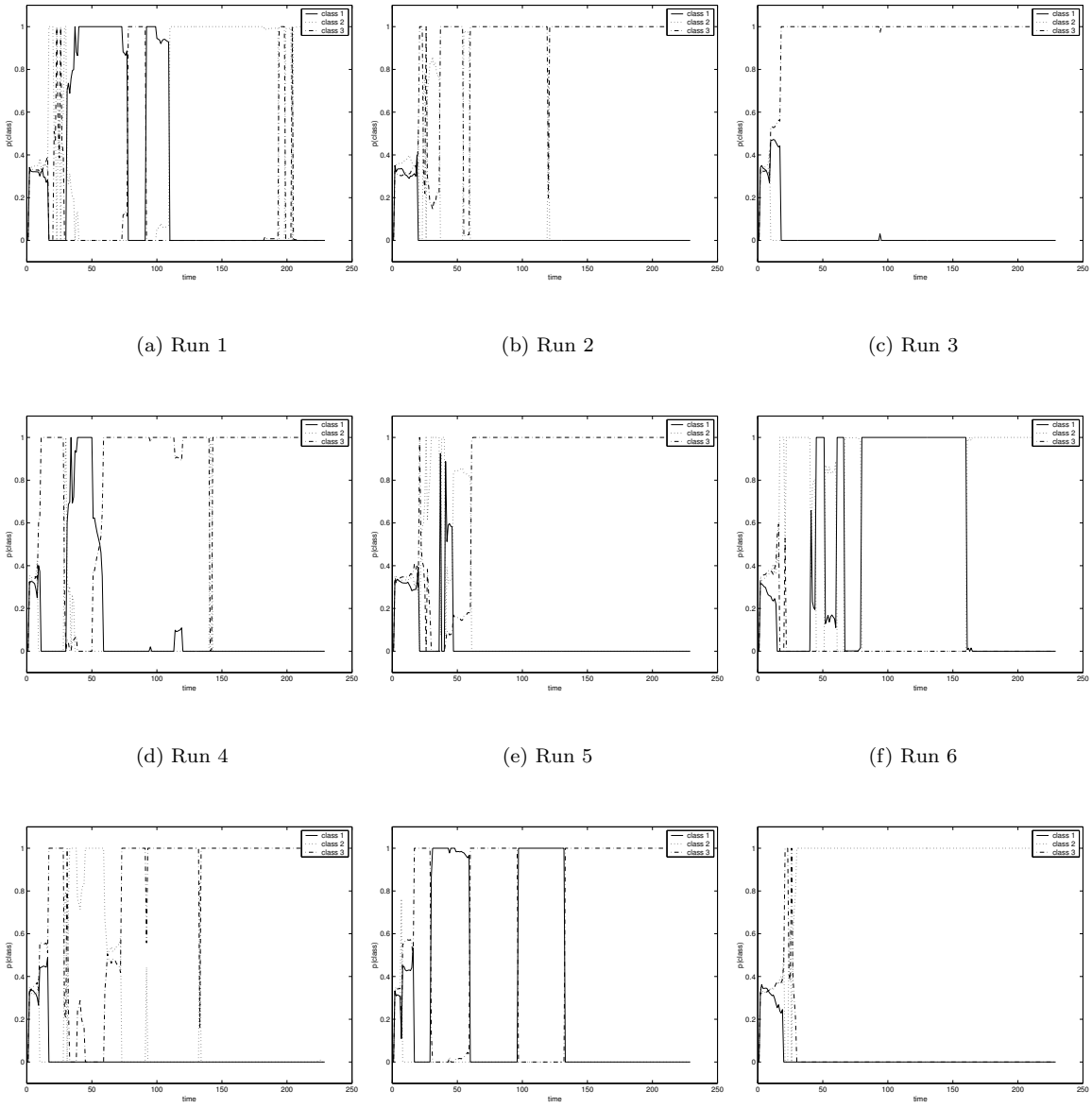


(d) Run 4 (e) Run 5 (f) Run 6



(g) Run 7 (h) Run 8 (i) Run 9

Figure 4.10: *Nine sets of strata weights with the true class as class 1.*

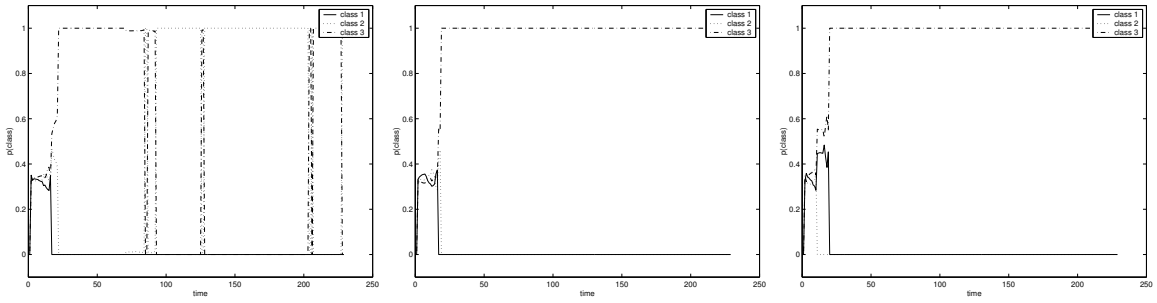


(a) Run 1 (b) Run 2 (c) Run 3

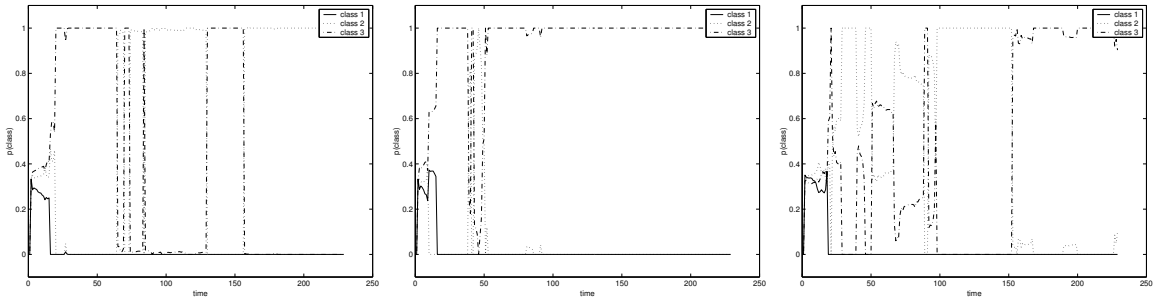
(d) Run 4 (e) Run 5 (f) Run 6

(g) Run 7 (h) Run 8 (i) Run 9

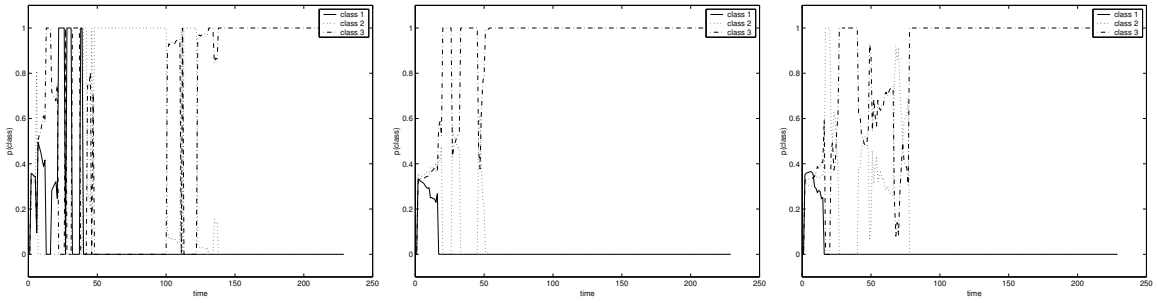
Figure 4.11: *Nine sets of strata weights with the true class as class 2.*



(a) Run 1 (b) Run 2 (c) Run 3



(d) Run 4 (e) Run 5 (f) Run 6



(g) Run 7 (h) Run 8 (i) Run 9

Figure 4.12: *Nine sets of strata weights with the true class as class 3.*

classifying targets on the basis of their dynamic behaviour. It has been demonstrated that it is possible to jointly track such manoeuvring targets and classify their manoeuvrability.

Efficient Data Association 5

IMPORTANT NOTICE

This chapter describes an approach patented by QinetiQ[75]. Implementing this approach without a licence will infringe this patent and is illegal. A free evaluation licence for a MATLAB implementation of the patent can be obtained by emailing <ehm@signal.qinetiq.com>.

5.1 Introduction

Data association is the primary source of complexity in multi-target tracking; at each time step one receives several candidate measurements and one needs to update all the targets' distributions while enforcing the constraint that each measurement relates to zero or one targets and each target relates to zero or one measurements. If one defines a *joint event* as a valid combination of measurements being associated with targets, then the number of such joint events grows exponentially with the number of targets. Ideally one would treat the joint event as a nuisance parameter and integrate it out and so include the effect of all the joint events on each targets' distribution. One can then represent the multi-target posterior as a product of marginal distributions for each target. This is called Mutual Exclusion[66] in the particle filtering literature. The same idea is used in the Kalman Filter tracking literature under the name of the Joint Probabilistic Data Association Filter, JPDAF[38]. Since the JPDAF was proposed, the approach has also been generalised so that each target's distribution is modelled as a mixture of Gaussians rather than a single Gaussian[93].

The exponential growth of the number of joint events with the number of targets and number of measurements has meant that these approaches have hitherto been presumed too computationally expensive to be used with large numbers of targets. As a result, research has either approximated the soft decision (averaging over the joint event) with a hard decision[94, 98, 102] (finding best joint event, usually at some point in the past) or used Monte-Carlo sampling to approximate the summation involved in the soft decision[49, 77, 90, 106]. The approach described here is to exactly calculate the sum over the exponentially growing number of joint events in sub-exponential time[72, 75]. Note that the approach is exact and fast

and is not iterative[1] and is patented by QinetiQ.

The idea is to explicitly exploit the same redundancy that must be being implicitly exploited by algorithms that find the best (often maximum probability) assignment of targets to measurements. Just as one does not need to consider all joint events when finding the best assignment, so one might assert that one does not need to consider all joint events when calculating sums over the joint events. This redundancy is exploited by noticing that it is possible for different assignments of some of the targets to the measurements to leave the same set of measurements for the remaining targets to be assigned to. Consider that the targets are assigned measurements in turn in all such ways that the constraints are not violated. The enumeration of joint events can then be considered as a tree, where that nodes in the tree correspond to assignments of some of the targets to some of the measurements. This tree has repeated structure. In some sense, nodes in the tree are equivalent; multiple nodes have the same subtrees below them. This idea was exploited by other authors previously[127]. However, the approach proposed in [127] only exploits some of the redundancy; two nodes are considered to be equivalent if the same set of measurements are used by the targets considered up to a node. This exploits the permutation symmetry of the assignment problem; the impact on the assignment for subsequent targets isn't altered by the order in which the measurements were assigned to the targets considered up to the node. The approach proposed here differs in that two nodes are considered equivalent if the set of measurements available for the subsequent targets are the same. Two equivalent nodes can then correspond to a different set of measurements being assigned to the targets up to the nodes but these sets having the same intersection with the measurements to which the subsequent targets can be assigned. The impact is large since while the number of sets of measurements used (and so nodes in the tree and computational cost of the algorithm in [127]) can only grow as more targets are considered albeit substantially slower than the number of joint events, the number of sets of measurements available can reduce as more targets are considered. As a result, the approach of [127] has a computational cost that grows as (slow) exponential while the approach proposed here has a computational cost that while difficult to define, appears to be pseudo-polynomial (for realistic tracking scenarios for which the hard decision logic based algorithms would converge).

In section 5.2 mutual exclusion is described using an explicit sum over the associations of targets with measurements. In section 5.3, a fast method for calculating the sum is discussed. In section 5.4, two illustrative examples are used to demonstrate the significant reduction in computational cost of implementing mutual exclusion using the approach. Finally, section 5.5 concludes.

[1] To the knowledge of the author all of the many alternative schemes are either approximate - in all but some very specific cases - and fast (for example [22, 83, 99]) or do not scale as well as the proposed approach with the number of targets and are exact (for example [127]).

5.2 Mutual Exclusion

The aim here is to formulate the problem faced by mutual exclusion as it relates to the summation over joint events in the context of particle filtering. Hence, only a concise recap is given of particle filtering, of the association of targets with measurements and of how the two relate.

5.2.1 Particle Filtering

Recall that a single target particle filter uses the diversity of a set of N samples to represent the uncertainty of the state of the target. The samples are hypotheses for the evolution of the target state, $x_{1:t} = x_1 \dots x_t$, consistent with the history of measurements, $y_{1:t} = y_1 \dots y_t$. The kth particle consists of a hypothesis for the state sequence, $x^k_{1:t}$, and an associated weight, w^k_t. At each iteration, this state sequence is extended using a convenient *proposal* distribution which often takes the convenient form $q\left(x_t|x_{t-1}, y_t\right)$, such that the particles only need to remember the hypothesis for the current state.

$$x^k_t \sim q\left(x_t|x^k_{t-1}, y_t\right) \tag{5.1}$$

The weights are then updated to reflect the disparity between this proposal and the posterior from which one is (presumably) unable to draw samples from, $p\left(x_t|x_{t-1}, y_t\right)$.

$$w^k_t \propto w^k_{t-1} \frac{p\left(y_t|x^k_t\right) p\left(x^k_t|x^k_{t-1}\right)}{q\left(x^k_t|x^k_{t-1}, y_t\right)} \tag{5.2}$$

The weights are normalised to sum to unity. The fact that the proposal and posterior are different means that the weights necessarily become skewed over time. To counter this, a *resampling* step is used to replicate those particles with large weights and discard those particles with small weights such that the total number of samples remains constant, but the computational load is spread more evenly across the posterior.

To enable the focus to be on mutual exclusion and avoid unnecessary obfuscation, the proposal used is assumed to be the prior:

$$q\left(x_t|x_{t-1}, y_t\right) \triangleq p\left(x_t|x_{t-1}\right). \tag{5.3}$$

The reason that using the prior can often constitute a convenient choice of proposal distribution is that the weight update simplifies considerably:

$$w^k_t \propto w^k_{t-1} p\left(y_t|x^k_t\right). \tag{5.4}$$

5.2.2 Association of Targets to Measurements

There are typically multiple measurements at each time step, the ith of the N_M such measurements being y^i_t. To model the fact that each target may be undetected, another hypothesis is used. Hence, a dummy measurement is introduced so that when $i = 0$, the target is undetected and when $1 \leq i \leq N_M$, the target is assigned to the ith measurement.

The assignment of targets to measurements is assumed to be decomposed into an assignment for each of the N_T targets. The i_jth measurement is assigned to the jth target. Gating is used to select a subset of

measurement hypotheses for each target which are considered candidates for the assignment; the subset is made up of the dummy and those measurements that lie in high probability regions of the target's likelihood. In any case, this assignment can be represented as an indicator vector, Ω_j, with $N_M + 1$ elements ω_{ji}:

$$\Omega_j = [\omega_{ji}] \tag{5.5}$$

where

$$\omega_{ji} = \begin{cases} 1 & i = i_j \\ 0 & i \neq i_j \end{cases} \tag{5.6}$$

The assignment for all the targets can then be represented as an indicator matrix, Ω, where each row is such an indicator vector:

$$\Omega = \left[{\Omega_1}^T \ldots {\Omega_{N_T}}^T\right]^T = [\omega_{ji}] \tag{5.7}$$

where, as before:

$$\omega_{ji} = \begin{cases} 1 & i = i_j \\ 0 & i \neq i_j \end{cases} \tag{5.8}$$

Since it is assumed that each measurement is assigned to zero or one targets and that each target is assigned to zero or one measurements, there are constraints on the elements of this matrix:

$$\sum_{j=1}^{N_T} \omega_{ji} \leq 1 \qquad i = 1 \ldots N_M \tag{5.9}$$

since each measurement is assigned to at most one target and

$$\sum_{i=0}^{N_M} \omega_{ji} = 1 \qquad \forall j \tag{5.10}$$

since each target is assigned to exactly one measurement hypothesis (either a hypothesis relating to a real measurement or to the dummy measurement).

It is easy to impose the second constraint by assigning one measurement hypothesis to each of the N_T targets in turn. The resulting matrix representation of the assignment will then potentially violate the first constraint. V is defined to be the set of valid assignments which do not violate the constraint in this way. So, if $\Omega \in V$, then the constraints are satisfied. Since it is possible (but computationally expensive) to enumerate all the possible assignment matrices, and since it is possible to test each resulting matrix for being in the set V, it is possible to enumerate the members of the set of valid assignments V.

An alternative approach to including assignment matrices that violate the constraints is to build up the assignment matrices using a tree of associations possible for each target. So, the topmost nodes in the tree represent the valid choices of assignments for the first target. Similarly, the second layer of nodes represent the valid assignments for the first and second targets. The jth layer of nodes then represents the valid assignments for the first to jth targets. Each node then needs to store the measurements that have been *used* by the targets down to the target under consideration.

To enumerate the set of valid assignment matrices, each target is considered in turn and the assignment matrices built up target-by-target. For each node for the previous target, a new node for the current target is created for each gated measurement hypotheses that does not result in any violation of the constraints. So, the tree represents an enumeration of the valid sequences of assignments for the targets by considering each target in turn.

It should be observed that while only enumerating those association matrices that satisfy the constraints does avoid some computations, the computational expense is still prohibitive for large numbers of targets.

5.2.3 Integrating out the Association Matrix

When considering multiple targets, one would like to have a set of particles for each target so the kth particle for the jth target then consists of a hypothesis, $x_t^{k_j}$, and a weight, $w_t^{k_j}$. If the assignment is known, we could update all the particle weights using the approach outlined in the previous section. For the jth target, the associated measurement, i_j, would be used in equation (5.4):

$$w_t^{k_j} \propto w_{t-1}^{k_j} p\left(y_t^{i_j} | x_t^{k_j}\right) \tag{5.11}$$

where k_j is the kth particle for the jth target.

However, the assignment is typically unknown and so can either be estimated jointly with the state or integrated out. Here, the assignment is integrated out. Hence for the jth target, (5.4) becomes:

$$\begin{aligned} w_t^{k_j} \propto & w_{t-1}^{k_j} \sum_{i \in G_j} p\left(y_t^i, i | x_t^{k_j}\right) & (5.12) \\ = & w_{t-1}^{k_j} \sum_{i \in G_j} p\left(y_t^i | x_t^{k_j}, i\right) p\left(i|j\right) & (5.13) \end{aligned}$$

where G_j is the gated set of measurement hypotheses for the jth target.

$p\left(i|j\right)$ is the probability of the ith measurement being assigned to the jth target, so these probabilities sum to unity over the measurements and are calculated by marginalising over the valid set of assignment matrices:

$$\sum_{i=0}^{N_M} p\left(i|j\right) = 1 \tag{5.14}$$

$$\begin{aligned} p\left(i|j\right) &= \frac{\sum_{\Omega \in V, \omega_{ji}=1} p\left(\Omega\right)}{\sum_{\Omega \in V} p\left(\Omega\right)} & (5.15) \\ &= \frac{\sum_{\Omega \in V, \omega_{ji}=1} p\left(\Omega\right)}{\sum_{i=0}^{N_M} \left(\sum_{\Omega \in V, \omega_{ji}=1} p\left(\Omega\right)\right)} & (5.16) \end{aligned}$$

where the numerator is a sum over all valid association matrices that have $\omega_{ji} = 1$ and the denominator is a sum of this quantity over all the gated association hypotheses (which ensures that the normalisation condition of (5.14) holds).

It is assumed that (for valid association matrices such that $\Omega \in V$) the joint association likelihood can be expressed as a product of terms for each target; this is the same approach taken by popular hard decision logic algorithms such as global nearest neighbour[8]:

$$p(\Omega) = \prod_{j=1}^{N_T} p\left(\omega_{ji_j}\right) \tag{5.17}$$

$p\left(\omega_{ji_j}\right)$ is the likelihood based on a single-target model where $i_j = 1$, which can be calculated by marginalising over the particles:

$$p\left(\omega_{ji_j}\right) = \sum_{k_j} w_t^{k_j} p\left(y_t^{i_j} | M^{k_j}\right) \tag{5.18}$$

where M^{k_j} is the parameter set for the kth particle for the jth target.

In the context of particles, $M^{k_j} \triangleq x_t^{k_j}$; the parameter set is just the sampled state of the particle. In an approach based on a Gaussian mixture model, such as that in [93], the sum would be over the mixture components, so the parameter set for each component is the mean and covariance of the relevant Gaussian component. In the case of the JPDAF, there is only a single Gaussian in the mixture, so only a single term in the summation. The idea common to all these algorithms is to sum over all the valid association matrices.

It is worth noting that the formulation of the JPDAF often described in the literature does not involve joint likelihoods with the same form as (5.18), but joint likelihoods that include another term which is dependent on the number of false alarms (and so number of targets that are associated with the dummy measurement)[10]. The assertion made here is that this restriction is justifiable on the basis that its use is widespread in other algorithms (such as the global nearest neighbour algorithm). Furthermore, the author believes that it is the summation over joint events rather than the form of the joint likelihood that provides the key benefit of using the JPDAF. As will be discussed, the restriction makes it possible to realistically implement JPDAF for large numbers of targets, providing further strong motivation for the restriction.

5.3 Fast Mutual Exclusion

The problem with the approach described in the previous section is that an enumeration of the valid association matrices in (5.16) results in a combinatorial explosion; the number of valid association matrices grows exponentially with the number of targets and measurements. The aim in this section is to present an approach to calculating exactly the numerator (and so denominator) in (5.16) in sub-exponential time.

5.3.1 Tree Formation

As described to this point, mutual exclusion enumerates all possible assignment matrices that satisfy the constraints of (5.9) and (5.10).

The key idea of the approach described here is for the nodes for the jth target to enumerate the valid associations possible for the $N_T - j$ remaining targets and not the associations of the j targets considered up to the jth target. The reason for the huge computational saving that results is that several parent nodes for one target can result in the same child node.

So, an additional step is introduced that, when creating a new node for the jth target checks to see if there are any existing nodes for the jth target that can result in the same set of valid associations possible for the $N_T - j$ remaining targets. The *identity* of a given node for the jth target defines the set of valid associations possible for the node and is taken to be the intersection of the measurements used by the j targets up to the node and the *subsequent associations possible* for the jth target. The subsequent associations possible is the union of the gated hypotheses for the remaining $N_T - j$ targets. This quantity can be calculated for all targets in a preprocessing step that recurses through the list of targets in reverse order. The identity of a child node can be calculated by taking the intersection of this quantity and the union of the identity of the parent and the measurement corresponding to the relationship between the parent and child node.

For each parent node for a given target, candidate child nodes are then created by considering all gated hypotheses that are in the identity for the node. An identity is then calculated for the candidate child node. If a node with the same identity already exists then this existing node is added to the list of children of the parent node and the parent added to the child's list of parents. The parent-child relationship is then labelled with the association hypothesis that gave rise to the connection; hence there can be multiple connections between the same pair of parent and child nodes as a result of different choices of measurement hypotheses that give rise to the same set of valid possible associations for the remaining targets. Only if no node exists with the same identity (and so the same set of valid associations possible for the $N_T - j$ remaining targets) is a new child node actually created. The result is that the tree is replaced by a net of interconnected nodes.

The leaf nodes (the nodes for the N_Tth target) in the tree that enumerates association matrices are then equivalent to different descents through the net. The fact that descents through the net explicitly share common structure means that the net provides a succinct and exact representation of the problem. Indeed, the use of this net opens the door to summing the exponentially increasing number of terms in (5.16) in sub-exponential time as will be discussed in the sequel.

5.3.2 Probability Calculations

It still remains to calculate the quantities in the numerator of (5.16) from this net. The sum is over all descents of the net that include a given element in the assignment matrix being unity. Since all descents must pass through one of the nodes in the net for the jth target, the sum over the valid hypotheses can be replaced with a sum over the nodes:

$$\sum_{\Omega \in V, \omega_{ji}=1} p(\Omega) = \sum_{n_j=1}^{N_{n_j}} p_T(i|n_j) \tag{5.19}$$

where there are N_{n_j} nodes for the jth target and where $p_T(i|n_j)$ is the sum of the joint likelihoods for descents that include $\omega_{ji} = 1$ that go through the n_jth node (for the jth target).

If the restriction of (5.18) is imposed then $p_T(i|n_j)$ can be efficiently calculated as a product of three terms: a term relating to the descents to those of the nodes' parents that have the ith measurement associated to the parent-child relationship, a term relating to the connection of the node to its parent (ie. $p(\omega_{ji})$) and a term relating to the descents possible for the remaining $N_T - j$ targets:

$$p_T(i|n_j) = p_U(n_j)\, p(\omega_{ji})\, p_D^{\star}(i, n_j) \tag{5.20}$$

where

$$p_D^{\star}(i, n_j) = \sum_{n'_{j-1} \in P(n_j), R\left(n'_{j-1}, n_j\right) = i} p_D\left(n'_{j-1}\right) \tag{5.21}$$

and where $P(n_j)$ is the list of parents of node n_j and $R\left(n'_{j-1}, n_j\right)$ is the index of the measurement associated with the parent-child relationship of the n_jth node (for the jth target) and the n'_{j-1}th node (for the $(j-1)$th target).

To calculate $p_D(n_j)$ and $p_U(n_j)$, a similar approach can be used to the two steps constituting the forward-backward algorithm used in Hidden Markov Models, HMMs[100]:

$$p_D(n_j) = \sum_{n'_{j-1} \in P(n_j)} p\left(\omega_{jR\left(n'_{j-1}, n_j\right)}\right) p_D\left(n'_{j-1}\right) \tag{5.22}$$

$$p_U(n_j) = \sum_{n'_{j+1} \in C(n_j)} p\left(\omega_{(j+1)R\left(n_j, n'_{j+1}\right)}\right) p_U\left(n'_{j+1}\right) \tag{5.23}$$

where it should be noted that the same node can appear more than once in the node's lists of parents, $P(n_j)$, and children, $C(n_j)$.

So, to calculate the probabilities, the downwards and upwards probabilities to each node are calculated. These probabilities are combined with the probability of each association hypothesis to give the sum of probabilities of all descents through the net that pass though each node (and are related to the given hypothesis). Hence it is possible to calculate the quantities of interest from the net.

5.3.3 Ordering of Targets

The approach described to this point will drastically reduce the computational expense associated with the calculation of the numerator of (5.16) given an ordering of the targets. However, the ordering of the targets will have an effect on the computational expense. To minimise the computational expense the targets need to be ordered such that the number of nodes in the net is minimised.

One simple step is to adopt the same approach as described in [125]; the targets are divided into clusters. The clusters are defined such that the cluster's set of measurements has no intersection with the other clusters' sets of measurements. These sets of measurements for each cluster are the union of all the gated measurements for all the targets within the cluster. The sums in (5.16) then only need consider the cluster containing the given target since all the joint likelihoods (for all the targets) factorise with one term for each of the clusters.

This clustering can be thought of as an ordering of the targets according to the cluster to which each target belongs. The extension to ordering the targets within the clusters is non-trivial to solve optimally, so a simple heuristic approach, that has been found to be effective, is advocated here.

The idea is to start with N_T lists of targets in a cluster where the jth list initially contains just the jth target. At each iteration of the ordering algorithm, the union of the gated measurements for all the targets in each list is calculated. Then the pair of lists with the largest number of measurements common to both lists' unions is found. This pair of lists is then combined. The combination process considers potentially reversing the order of each of the lists and appending the lists in either order: a appended with b; a reversed and appended with b; b appended with a; b appended with a reversed. The combination is chosen that minimises the intersection between the newly adjoining targets' sets of gated measurements. This new combined list then replaces the two constituent lists. This process is repeated N_T times until there is one list of targets.

5.4 Results

To demonstrate the gains offered by the approach, two exemplar data association problems are considered. In the first example, the number of targets is small enough to make it feasible to use an approach based on enumerating the valid association matrices. This enables a comparison of the proposed approach with that previously proposed. In the second case, the number of targets in the cluster is large enough to make it infeasible to use mutual exclusion without the proposed approach.

To generate the problems, 80 targets and 80 measurements were sampled uniformly across an area such that $0 < x < 1$ and $0 < y < 1$. The gates for each target were circular regions centered on each target such that a measurement was considered within the gate if the squared distance of the measurement from the target was less than d. For the first example, $d = 0.005$. For the second example, $d = 0.01$. In both cases, the largest resulting cluster is that considered.

Small Number of Targets

The first cluster is that shown in figure 5.1; there are 9 measurements and 9 targets. The large circles represent the gates for each target and the crosses the measurements.

The tree resulting from enumerating the valid association matrices is that in figure 5.2. The number of leaf nodes is 10200. This is less than the 81920 association matrices that satisfy (5.10), but still large.

The net resulting from the proposed approach is shown in figure 5.3. The maximum width of the net is 8 nodes. The quantities calculated using the net are within machine precision errors of those calculated using the tree. The time taken by the two approaches is of a similar order of magnitude; while using the tree does involve a larger computational cost, the simpler implementation makes the tree-based approach applicable in scenarios of this (small) size. The combinatorial explosion associated with mutual association is not problematic in this scenario.

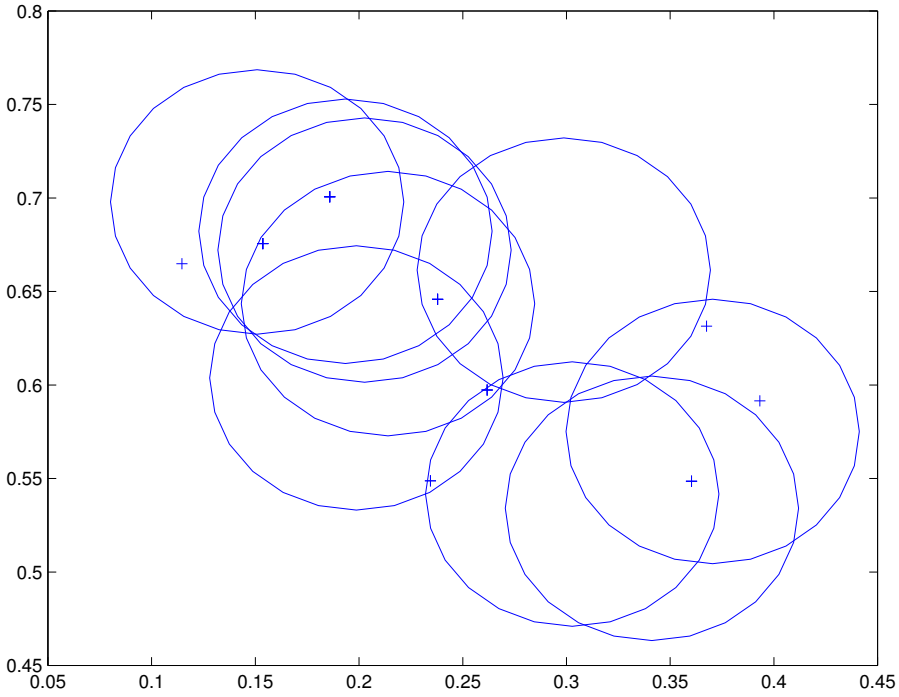


Figure 5.1: *Illustration of gating arrangement for a difficult multi-target tracking data-association example.*

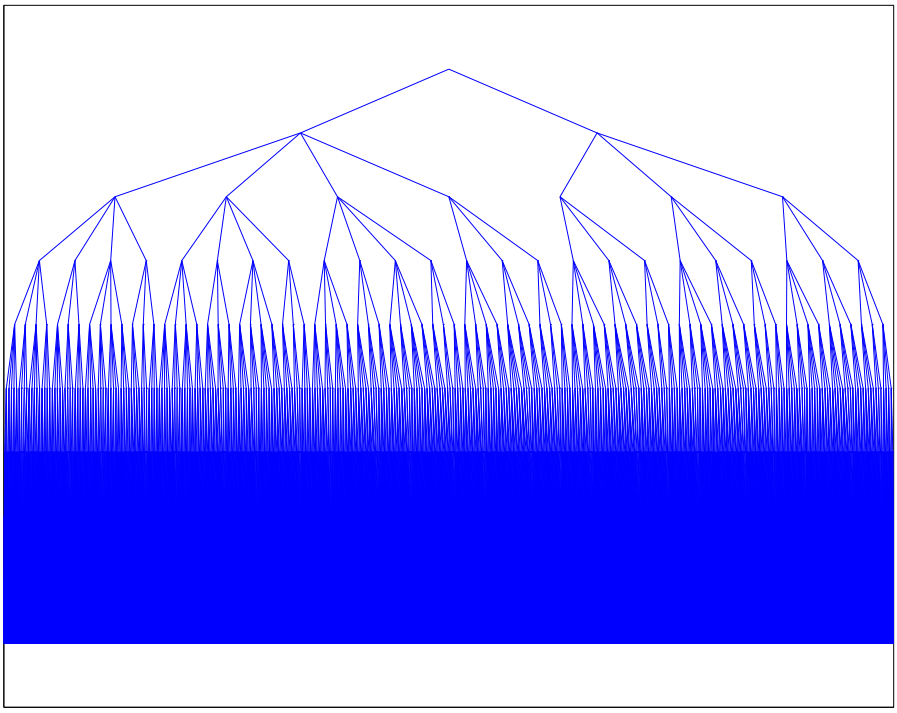


Figure 5.2: *Tree resulting from enumeration of valid association matrices for example involving a small number of targets.*

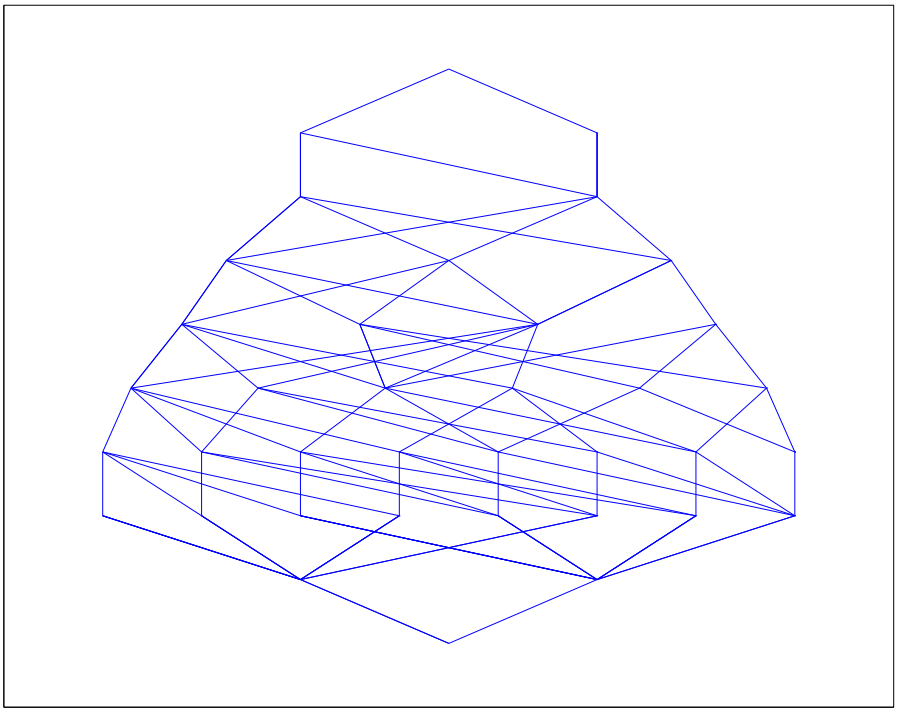


Figure 5.3: *Net resulting from proposed approach for example involving a small number of targets.*

Large Number of Targets

The second cluster consists of 33 measurements and 41 targets and is shown in figure 5.4.

The total number of association matrices that satisfy (5.10) is greater than 10^{22}. While the number of valid association matrices will be smaller, it is unlikely to be a very different order of magnitude. Indeed, the combinatorial explosion results in the tree based approach being unable to be implemented in a useful timescale.

The net resulting from the proposed approach is shown in figure 5.5 and, at its widest, is 64 nodes wide. As a result the terms in (5.16) can be calculated in a few seconds on a desktop PC.

5.5 Conclusions

A fast method has been described that can be used to exactly calculate the sum over an exponentially growing number of joint events in substantially less than exponential time. It has been shown that it is now possible to use mutual exclusion to track a large number of targets.

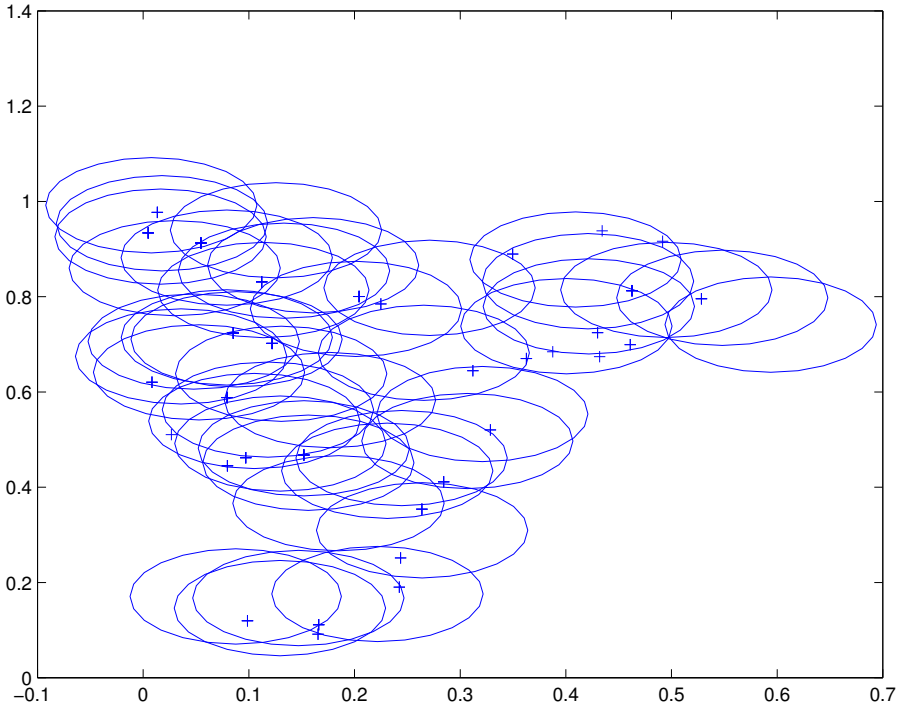


Figure 5.4: *Illustration of gating arrangement for example involving a large number of targets.*

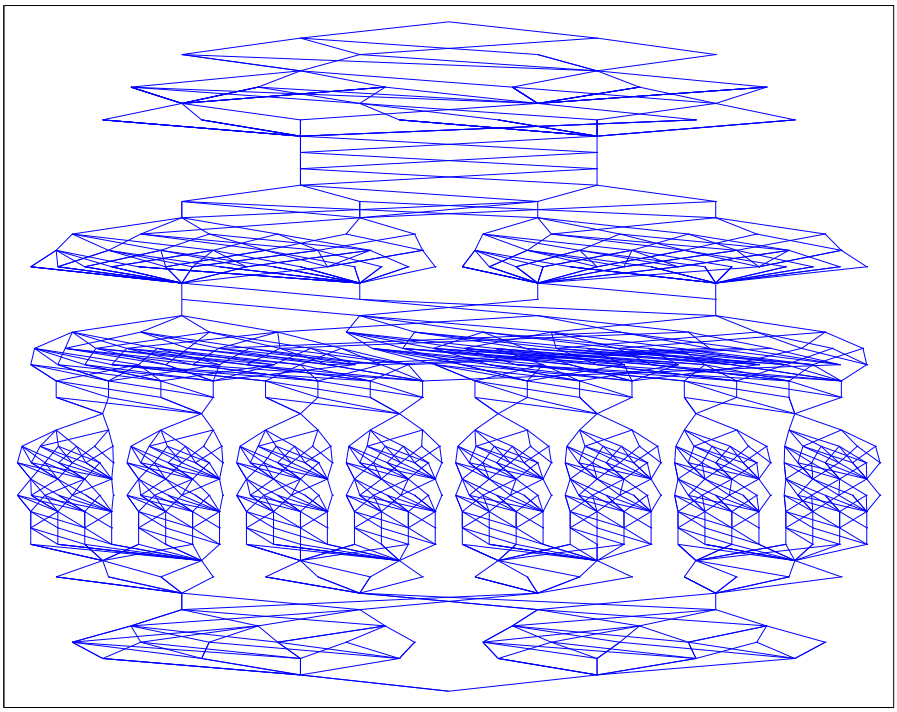


Figure 5.5: *Net Multi-target tracking example using computationally efficient approach*

Images 6

6.1 Introduction

Time series can be efficiently filtered using the Kalman filter[126] and associated sequential algorithms. In the cases when the assumptions of such time series algorithms hold, the algorithms are able to completely characterise the joint distribution of the state at all L points in a time series in $O(L)$ time. This expediency is a result of being able to order the time epochs and choose a model such that, given a state at a particular time, the states at later times are independent of the states at previous times.

Images cannot be ordered in the same way; one cannot say whether one pixel was after another. As a result, to use Bayesian inference to analyse images, one typically has to resort to numerical approaches such as MCMC, which is computationally expensive, or to using template matching, which relies on low amounts of noise being present.

Previous work has exploited the fact that images can be decomposed into a time series, where each element of the time series is a row in the image[59, 104]. For near square images, the computational expense associated with inference is then dominated by the within row calculations. With J columns in the image, these calculations involve the inversion of a $J \times J$ matrix and so are $O(J^3)$. If there are I rows, then the computational expense of the algorithm is $O(IJ^3)$ and so if $I \approx J$ (ie. if the image is near square), then the computational expense is approximately quadratic in the number, IJ, of pixels.

This chapter exploits the fact that the within row calculations themselves can be computed using sequential algorithms. Hence the within row calculations can be computed in $O(J)$ time. This is achieved by imposing an ordering within the rows. The resulting algorithm has a computational cost that is $O(IJ)$ and so linear in the number of pixels, making it viable to perform exact Bayesian inference in large images. The approach is illustrated by generalising the Kalman Smoother to the analysis of images rather than of time series. Just as the Kalman Smoother is exact, so this approach is exact. The approach generalises to analysis using other time series algorithms.

Section 6.2 describes the model used for images while 6.3 describes how images described by this model can be decomposed into rows and how the analysis of image can then be performed using a (vertical) Kalman Smoother. Section 6.4 then describes how all the operations on the rows can themselves be performed using (horizontal) Kalman Smoothers. Finally, section 6.6 draws some conclusions.

6.2 Image Model

Consider some data consisting of a measurement at every point in a regular grid of I rows by J columns. At each location, $\{i,j\} \in \{1,2,\ldots,I\} \times \{1,2,\ldots,J\}$ there is a measurement, z_i^j, which is related to the state of the system x_i^j:

$$p\left(z_i^j | x_{1:I}^{1:J}\right) = p\left(z_i^j | x_i^j\right) \tag{6.1}$$

Consider a causal dynamic model[1] that factorises into a (v)ertical, (h)orizontal and (s)tationary[2] term when conditioned on the states from a corner up to (but excluding) a point:

$$p\left(x_i^j | x_{1:i-1}^{1:j-1}, x_i^{1:j-1}, x_{1:i-1}^j\right) = p\left(x_i^j | x_{i-1}^j, x_i^{j-1}\right) = v\left(x_i^j | x_{i-1}^j\right) h\left(x_i^j | x_i^{j-1}\right) s\left(x_i^j\right) \quad i>1, j>1 \tag{6.4}$$

with special cases along the edges and in the corner, in which cases some of the terms in the conditioning don't exist:

$$p\left(x_i^j | x_{1:i-1}^{1:j-1}, x_i^{1:j-1}, x_{1:i-1}^j\right) = p\left(x_i^j | x_{i-1}^j\right) = v\left(x_i^j | x_{i-1}^j\right) s\left(x_i^j\right) \quad i>1, j=1 \tag{6.5}$$

$$p\left(x_i^j | x_{1:i-1}^{1:j-1}, x_i^{1:j-1}, x_{1:i-1}^j\right) = p\left(x_i^j | x_i^{j-1}\right) = h\left(x_i^j | x_i^{j-1}\right) s\left(x_i^j\right) \quad i=1, j>1 \tag{6.6}$$

$$p\left(x_i^j | x_{1:i-1}^{1:j-1}, x_i^{1:j-1}, x_{1:i-1}^j\right) = p\left(x_i^j\right) = s\left(x_i^j\right) \quad i=1, j=1 \tag{6.7}$$

It is also assumed that all the models are linear Gaussian model of the following forms:

$$h\left(x_i^j | x_i^{j-1}\right) = \mathcal{N}\left(x_i^j; A_H x_i^{j-1}, Q_H\right) \tag{6.8}$$

$$v\left(x_i^j | x_{i-1}^j\right) = \mathcal{N}\left(x_i^j; A_V x_{i-1}^j, Q_V\right) \tag{6.9}$$

$$s\left(x_i^j\right) = \mathcal{N}\left(x_i^j; m_s, C_s\right) \tag{6.10}$$

$$p\left(z_i^j | x_i^j\right) = \mathcal{N}\left(z_i^j; H_z x_i^j, R\right) \tag{6.11}$$

[1]The discussion here relates to causal dynamic models for images. By making the state partially observed it is possible to formulate non-causal models as partially observed causal models. An analogous approach can be used for time series where the partially observed parts of the state at one time relate to the future values of the non-causal process. As an example, consider the following non-causal model:

$$x_t = \frac{x_{t+1} - x_{t-1}}{2} + \omega_t^x \tag{6.2}$$

which can be re-expressed as the following partially observed causal process:

$$\bar{x}_t = \begin{bmatrix} x_{t+2} \\ x_{t+1} \\ x_t \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x_{t+1} \\ x_t \\ x_{t-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega_t^x \end{bmatrix} = A\bar{x}_{t-1} + \bar{\omega}_t^x. \tag{6.3}$$

While A^{-1} does not exist in such cases, the discussion in this chapter will assume that A^{-1} does exist; extensions of this work to remove this assumption would be straightforward. Furthermore, it should be pointed out that the discussion of models given here is intentionally simplistic since the emphasis of the work is on deriving a methodology for analysing images and not on the models that should be used to conduct such analysis. A more thorough treatment analogous to that conducted in chapter 3 for time but for spatial models would complement the work of this chapter.

[2]The stationary term is used since, while there is a concept of a correct ordering of a time series and so an obvious point (in time) to define the prior, there is no such concept in images. This choice of prior is chosen to facilitate rotational invariance.

The reverse vertical dynamics can be defined using the discussion in section 2.3.3, which does necessitate evaluation of the priors at every point in the image[3]:

$$v^b\left(x_{i-1}^j|x_i^j\right) = \mathcal{N}\left(x_{i-1}^j; b_{V,i}^j + A_{V,i}^{b,j}x_i^j, P_{V,i}^{b,j}\right). \tag{6.12}$$

where $b_{V,i}^j$, $A_{V,i}^{b,j}$ and $P_{V,i}^{b,j}$ are assumed to have been pre-calculated such that they are consistent with the forwards dynamics above.

Similarly, the reverse horizontal dynamics can be similarly defined:

$$h^b\left(x_i^{j-1}|x_i^j\right) = \mathcal{N}\left(x_i^{j-1}; b_{H,i}^j + A_{H,i}^{b,j}x_i^j, P_{H,i}^{b,j}\right). \tag{6.13}$$

where $b_{H,i}^j$, $A_{H,i}^{b,j}$ and $P_{H,i}^{b,j}$ are again assumed to have been consistently pre-calculated.

Row to Row Dependence

The model is such that the dependence of one row on the previous row can be expressed as follows:

$$V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) = \prod_{j=1}^{J} v\left(x_{i+1}^j|x_i^j\right) \tag{6.14}$$

and a convenient term can also be defined which includes the effect of the model on the rows independent of the other rows:

$$H\left(x_i^{1:J}\right) = \prod_{j=2}^{J} h\left(x_i^j|x_i^{j-1}\right)\prod_{j=1}^{J} s\left(x_i^j\right). \tag{6.15}$$

The dependence of one row on all the rows below can then be expressed as:

$$\begin{aligned}
p\left(x_{i+1}^{1:J}|x_{1:i}^{1:J}\right) =& p\left(x_{i+1}^{1:J}|x_i^{1:J}\right) && (6.16)\\
=& p\left(x_{i+1}^1|x_i^1\right)\prod_{j=2}^{J} p\left(x_{i+1}^j|x_i^j, x_i^{j-1}\right) && (6.17)\\
=& v\left(x_{i+1}^1|x_i^1\right) s\left(x_{i+1}^1\right)\prod_{j=2}^{J} v\left(x_{i+1}^j|x_i^j\right) h\left(x_{i+1}^j|x_{i+1}^{j-1}\right) s\left(x_{i+1}^j\right) && (6.18)\\
=& \prod_{j=1}^{J} v\left(x_{i+1}^j|x_i^j\right)\prod_{j=2}^{J} h\left(x_{i+1}^j|x_{i+1}^{j-1}\right)\prod_{j=1}^{J} s\left(x_{i+1}^j\right) && (6.19)\\
=& V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) H\left(x_{i+1}^{1:J}\right). && (6.20)
\end{aligned}$$

With the dependence of one row on all the rows above similarly defined:

$$\begin{aligned}
p\left(x_i^{1:J}|x_{i+1:I}^{1:J}\right) =& p\left(x_i^{1:J}|x_{i+1}^{1:J}\right) && (6.21)\\
=& V^b\left(x_i^{1:J}|x_{i+1}^{1:J}\right) H\left(x_i^{1:J}\right). && (6.22)
\end{aligned}$$

[3]It would be reasonably straightforward to extend the analysis to avoid the need for reverse dynamics and such extensions aren't explored here.

6.3 Vertical Kalman Smoother

As pointed out by Lavine[59], if one can divide the field up into rows so that conditional on $x_i^{1:J}$, $x_{i-1}^{1:J}$ and $x_{i+1}^{1:J}$ are independent, one can use a Kalman filter[126] to process the rows in order to efficiently obtain $p\left(x_i^{1:J}|z_{1:i}^{1:J}\right)$ recursively through i. One can then recurse in reverse order through i to get the parameters of $p\left(x_{i:I}^{1:J}|z_{i:I}^{1:J}\right)$ such that one ends up with the parameters of $p\left(x_{1:I}^{1:J}|z_{1:I}^{1:J}\right)$. Here, the concern is only to efficiently compute $p\left(x_i^j|z_{1:I}^{1:J}\right)$ exactly, so all the cross-covariance terms that would completely parameterise $p\left(x_{1:I}^{1:J}|z_{1:I}^{1:J}\right)$ are not calculated. However, it is straightforward to see that these terms could be calculated if required through reasonably minor modifications of the algorithm.

As pointed out by [59], it is possible to exploit the fact that the precision matrix for the posterior of all the states stacked ontop of one another is block tridiagonal and, since the publication of [59], this has been exploited to devise an efficient scheme for obtaining samples from such a field by Rue[104]. The key realisation in improving the efficiency is that, because of the structure of the model, $p\left(x_{1:I}^{1:J}|z_{1:I}^{1:J}\right)$ can be expressed as a product as follows:

$$
\begin{aligned}
p\left(x_{1:I}^{1:J}|z_{1:I}^{1:J}\right) &= p\left(x_1^{1:J}|z_{1:I}^{1:J}\right)\prod_{i=2}^{I} p\left(x_i^{1:J}|x_{1:i-1}^{1:J}, z_{1:I}^{1:J}\right) && (6.23)\\
&= p\left(x_1^{1:J}|z_{1:I}^{1:J}\right)\prod_{i=2}^{I} p\left(x_i^{1:J}|x_{i-1}^{1:J}, z_{1:I}^{1:J}\right) && (6.24)\\
&= p\left(x_1^{1:J}|z_{1:I}^{1:J}\right)\prod_{i=2}^{I} \frac{p\left(x_i^{1:J}, x_{i-1}^{1:J}|z_{1:I}^{1:J}\right)}{p\left(x_{i-1}^{1:J}|z_{1:I}^{1:J}\right)} && (6.25)
\end{aligned}
$$

So, all that is needed to describe $p\left(x_{1:I}^{1:J}|z_{1:I}^{1:J}\right)$ are the parameters of these factoring Gaussian densities:

$$
\begin{aligned}
m_i^{1:J}(z_{1:I}^{1:J}) &= \mathbb{E}\left[x_i^{1:J}|z_{1:I}^{1:J}\right] && i \in \{1, 2, \ldots I\} && (6.26)\\
P_{i,i}^{1:J,1:J}(z_{1:I}^{1:J}) &= \mathbb{E}\left[x_i^{1:J} x_i^{1:J\,T}|z_{1:I}^{1:J}\right] && i \in \{1, 2, \ldots I\} && (6.27)\\
P_{i,i-1}^{1:J,1:J}(z_{1:I}^{1:J}) &= \mathbb{E}\left[x_i^{1:J} x_{i-1}^{1:J\,T}|z_{1:I}^{1:J}\right] && i \in \{2, 3, \ldots I\} && (6.28)
\end{aligned}
$$

which is now posed as a sequential problem and solved using a Kalman Smoother.

6.3.1 Kalman Filter

Prediction

So, since i indexes the rows, the Kalman filter prediction step, at each step in the recursion through the rows, obtains $p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}\right)$ as follows:

$$
p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}\right) = \int p\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}\right) dx_i^{1:J} \qquad (6.29)
$$

Update

To then calculate $p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}\right)$, the update step of a Kalman filter is used:

$$
p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}\right) = \frac{p\left(z_{i+1}^{1:J}|x_{i+1}^{1:J}\right) p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}\right)}{p\left(z_{i+1}^{1:J}|z_{1:i}^{1:J}\right)} \qquad (6.30)
$$

(6.29) and (6.30) can then be combined:

$$p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}\right) = \frac{p\left(z_{i+1}^{1:J}|x_{i+1}^{1:J}\right)}{p\left(z_{i+1}^{1:J}|z_{1:i}^{1:J}\right)} \int p\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.31}$$

The model of the conditional dependence can be decomposed into the (V)ertical and a (H)orizontal factors in (6.20) as follows, such that the horizontal term is independent of $dx_i^{1:J}$ and so can appear as a factor outside the integral in (6.29):

$$p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}\right) = \int p\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.32}$$

$$= \int H\left(x_{i+1}^{1:J}\right) V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.33}$$

$$= H\left(x_{i+1}^{1:J}\right) \int V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}\right) dx_i^{1:J}. \tag{6.34}$$

Comparing (6.31) and (6.34), it is clear that the factor in (6.34) could have resulted from using a system without the horizontal conditional dependence but with processing a *pseudo-measurement*, $y_i^{1:J}$, that corrected for this; the pseudo-measurement and its likelihood are chosen to generate a term that is $H(x_{i+1}^{1:J})$. For the example considered by [59], this greatly simplifies the matrix manipulations involved with implementing the Kalman filter since the vertical interactions can be integrated out and then the horizontal dependencies can be incorporated in at a second stage. This equates to making the following relation:

$$p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) = \int V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.35}$$

$$p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i+1}^{1:J}\right) = \frac{p\left(y_{i+1}^{1:J}|x_{i+1}^{1:J}\right)}{p\left(y_{i+1}^{1:J}|y_{1:i}^{1:J}\right)} \int V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.36}$$

$$= H\left(x_{i+1}^{1:J}\right) \int V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) dx_i^{1:J}. \tag{6.37}$$

where it is assumed that the integrals necessary to calculate $p\left(y_{i+1}^{1:J}|x_{i+1}^{1:J}\right)$ and $p\left(y_{i+1}^{1:J}|y_{1:i}^{1:J}\right)$ exist.

By way of explanation, $p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)$ is then a prediction based only on the vertical (and not the horizontal) interactions.

6.3.2 Kalman Smoother

The Kalman filter runs recursively from $i = 1$ to $i = I$ to obtain $p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)$. In [59], a forwards filter backwards-smoother is used to recursively (backwards through the rows) calculate $p\left(x_i^{1:J}|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right)$. Here, the two-filter form of the smoother is used to calculate $p\left(x_i^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right)$ and then $p\left(x_i^{1:J}|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right)$ and finally $p\left(x_i^j|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right)$:

$$p\left(x_i^{1:J}|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right) \propto \frac{p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_i^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right)}{p\left(x_i^{1:J}\right)} \tag{6.38}$$

$$p\left(x_i^j|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right) = \int p\left(x_i^{1:J}|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right) dx_i^{1:j-1} dx_i^{j+1:J}, \tag{6.39}$$

where it has been assumed that $p\left(x_i^{1:J}\right)$ and the reverse dynamics needed to compute $p\left(x_i^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right)$ are available such that it is possible to compute:

$$p\left(x_i^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) = V^b\left(x_i^{1:J}|x_{i+1}^{1:J}\right) p\left(x_{i+1}^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right). \tag{6.40}$$

6.4 Horizontal Kalman Smoothers

The image model is such that x_i^{j-1} and x_i^{j+1} are independent when conditioned on $x_{i+1}^{1:J}$, $x_{i-1}^{1:J}$ and x_i^j. This makes it possible for a horizontal Kalman smoother to efficiently run along the rows and obtain the parameters of the distributions for $p\left(x_i^{1:J}|z_{1:i}^{1:J}\right)$ and $p\left(x_i^{1:J}|z_{i+1:I}^{1:J}\right)$ which can then be used in the vertical Kalman smoother. The multiplication of these two distributions can also be implemented using a Kalman smoother.

So, this section describes how to implement the various parts of the vertical Kalman smoother described in the previous section using horizontal Kalman smoothers that run along the rows. This is possible since all the distributions that are of interest can be completely described as follows:

$$p\left(x_i^{1:J}|\mathbb{I}\right) = p\left(x_i^1|\mathbb{I}\right)\prod_{j=2}^{J} p\left(x_i^j|x_i^{j-1},\mathbb{I}\right) \tag{6.41}$$

$$= p\left(x_i^1|\mathbb{I}\right)\prod_{j=2}^{J}\frac{p\left(x_i^{j-1:j}|\mathbb{I}\right)}{p\left(x_i^{j-1}|\mathbb{I}\right)} \tag{6.42}$$

where $\mathbb{I}$ is some conditioning.

So, all that is needed to completely describe $p\left(x_i^{1:J}|\mathbb{I}\right)$ are the parameters of these factorising Gaussian densities:

$$m_i^j(\mathbb{I}) = \mathbb{E}\left[x_i^j|\mathbb{I}\right] \qquad j \in \{1,2,\dots J\} \tag{6.43}$$

$$P_{i,i}^{j,j}(\mathbb{I}) = \mathbb{E}\left[x_i^j {x_i^j}^T|\mathbb{I}\right] \qquad j \in \{1,2,\dots J\} \tag{6.44}$$

$$P_{i,i}^{j,j-1}(\mathbb{I}) = \mathbb{E}\left[x_i^j {x_i^{j-1}}^T|\mathbb{I}\right] \qquad j \in \{2,3,\dots J\}\,. \tag{6.45}$$

6.4.1 Kalman Filter

Prediction

In the case of the prediction within the vertical Kalman filter, the distributions of interest are $p\left(x_i^{1:J}|z_{1:i}^{1:J},y_{1:i}^{1:J}\right)$ and $p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J},y_{1:i}^{1:J}\right)$. If $p\left(x_i^{1:J}|z_{1:i}^{1:J},y_{1:i}^{1:J}\right)$ is assumed to be of the form of (6.42), it is possible to show

that $p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)$ must also be of the same form:

$$p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) = \int V\left(x_{i+1}^{1:J}|x_i^{1:J}\right) p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.46}$$

$$= \int \prod_{j=1}^{J} v\left(x_{i+1}^{j}|x_i^{j}\right) p\left(x_i^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) \prod_{j=2}^{J} p\left(x_i^{j}|x_i^{j-1}, z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) dx_i^{1:J} \tag{6.47}$$

$$= \int p\left(x_i^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) v\left(x_{i+1}^{1}|x_i^{1}\right) \prod_{j=2}^{J} p\left(x_i^{j}|x_i^{j-1}, z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) v\left(x_{i+1}^{j}|x_i^{j}\right) dx_i^{1:J} \tag{6.48}$$

$$= \int p\left(x_i^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) v\left(x_{i+1}^{1}|x_i^{1}\right) \prod_{j=2}^{J} \frac{p\left(x_i^{j-1:j}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)}{p\left(x_i^{j-1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)} \frac{v\left(x_{i+1}^{j}|x_i^{j}\right) v\left(x_{i+1}^{j-1}|x_i^{j-1}\right)}{v\left(x_{i+1}^{j-1}|x_i^{j-1}\right)} dx_i^{1:J} \tag{6.49}$$

$$= p\left(x_{i+1}^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) \prod_{j=2}^{J} \frac{p\left(x_{i+1}^{j-1:j}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)}{p\left(x_{i+1}^{j-1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)} \tag{6.50}$$

Then, because $p\left(x_i^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)$ is Gaussian and of the form of (6.42), $p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right)$ is also of the form of (6.42) with the following parameters:

$$m_{i+1}^{j}(z_{1:i}^{1:J}, y_{1:i}^{1:J}) = A_V m_i^{j}(z_{1:i}^{1:J}, y_{1:i}^{1:J}) \qquad j \in \{1, 2, \dots J\} \tag{6.51}$$

$$P_{i+1,i+1}^{j,j}(z_{1:i}^{1:J}, y_{1:i}^{1:J}) = A_V P_{i,i}^{j,j}(z_{1:i}^{1:J}, y_{1:i}^{1:J}) A_V^T + Q_V \qquad j \in \{1, 2, \dots J\} \tag{6.52}$$

$$P_{i+1,i+1}^{j,j-1}(z_{1:i}^{1:J}, y_{1:i}^{1:J}) = A_V P_{i,i}^{j,j-1}(z_{1:i}^{1:J}, y_{1:i}^{1:J}) A_V^T \qquad j \in \{2, 3, \dots J\} \tag{6.53}$$

Hence, the (vertical) prediction step can be conducted independently for all the (neighbouring pairs of) pixels in the row; there is no need for a (horizontal) Kalman Smoother to run along the row to implement the prediction step in the vertical Kalman Filter.

Update

In the update step, it is possible to show that if $p\left(x_{i+1}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) = p\left(x_{i+1}^{1:J}|D\right)$, where D is the set of (D)ownward measurements, $z_{1:i}^{1:J}$ and $y_{1:i}^{1:J}$, is of the form of 6.42, then $p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}, y_{1:i+1}^{1:J}\right)$ must also be

of the same form:

$$p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J},y_{1:i+1}^{1:J}\right)\propto p\left(z_{i+1}^{1:J}|x_{i+1}^{1:J}\right)p\left(x_{i+1}^{1:J}|D\right)H\left(x_{i+1}^{1:J}\right) \tag{6.54}$$

$$=\prod_{j=1}^{J}p\left(z_{i+1}^{j}|x_{i+1}^{j}\right)p\left(x_{i+1}^{1}|D\right)\prod_{j=2}^{J}p\left(x_{i+1}^{j}|x_{i+1}^{j-1},D\right)\prod_{j=2}^{J}h\left(x_{i+1}^{j}|x_{i+1}^{j-1}\right)\prod_{j=1}^{J}s\left(x_{i+1}^{j}\right) \tag{6.55}$$

$$=\prod_{j=1}^{J}p\left(z_{i+1}^{j}|x_{i+1}^{j}\right)p\left(x_{i+1}^{1}|D\right)\prod_{j=2}^{J}\frac{p\left(x_{i+1}^{j-1:j}|D\right)}{p\left(x_{i+1}^{j-1}|D\right)}\prod_{j=2}^{J}h\left(x_{i+1}^{j}|x_{i+1}^{j-1}\right)\prod_{j=1}^{J}s\left(x_{i+1}^{j}\right) \tag{6.56}$$

$$\begin{aligned}=&\prod_{j=1}^{j'}p\left(z_{i+1}^{j}|x_{i+1}^{j}\right)p\left(x_{i+1}^{1}|D\right)\prod_{j=2}^{j'}\frac{p\left(x_{i+1}^{j-1:j}|D\right)}{p\left(x_{i+1}^{j-1}|D\right)}\prod_{j=2}^{j'}h\left(x_{i+1}^{j}|x_{i+1}^{j-1}\right)\prod_{j=1}^{j'}s\left(x_{i+1}^{j}\right)\\&\times\prod_{j=j'+1}^{J}p\left(z_{i+1}^{j}|x_{i+1}^{j}\right)\prod_{j=j'+1}^{J}\frac{p\left(x_{i+1}^{j-1:j}|D\right)}{p\left(x_{i+1}^{j-1}|D\right)}\prod_{j=j'+1}^{J}h\left(x_{i+1}^{j}|x_{i+1}^{j-1}\right)\prod_{j=j'+1}^{J}s\left(x_{i+1}^{j}\right)\end{aligned} \tag{6.57}$$

$$\propto\frac{p\left(x_{i+1}^{1:j'}|D,z_{i+1}^{1:j'},y_{i+1}^{1:j'}\right)p\left(x_{i+1}^{j':J}|D,z_{i+1}^{j'+1:J},y_{i+1}^{j'+1:J}\right)}{p\left(x_{i+1}^{j'}|D\right)} \tag{6.58}$$

where $p\left(x_{i+1}^{1:j'}|D,z_{i+1}^{1:j'},y_{i+1}^{1:j'}\right)$ and $p\left(x_{i+1}^{j':J}|D,z_{i+1}^{j'+1:J},y_{i+1}^{j'+1:J}\right)$ are of the form of 6.42, making it possible to completely define $p\left(x_{i+1}^{1:J}|D,z_{i+1}^{1:J},y_{i+1}^{1:J}\right)$ by calculating $p\left(x_{i+1}^{j-1:j}|z_{1:i+1}^{1:j},y_{1:i+1}^{1:j}\right)$ and $p\left(x_{i+1}^{j-1:j}|z_{1:i+1}^{j:J},y_{1:i+1}^{j:J}\right)$ for every $2\leq j\leq J$.

Horizontal Recursion for Implementation of Update

$p\left(x_{i+1}^{j}|D,z_{i+1}^{1:j},y_{i+1}^{1:j}\right)$ can be obtained from $p\left(x_{i+1}^{j-1:j}|D,z_{i+1}^{1:j},y_{i+1}^{1:j}\right)$ by marginalising:

$$p\left(x_{i+1}^{j}|D,z_{i+1}^{1:j},y_{i+1}^{1:j}\right)=\int p\left(x_{i+1}^{j-1:j}|D,z_{i+1}^{1:j},y_{i+1}^{1:j}\right)dx_{i+1}^{j-1} \tag{6.59}$$

A prediction step can then deduce $p\left(x_{i+1}^{j:j+1}|D,z_{i+1}^{1:j},y_{i+1}^{1:j+1}\right)$ from $p\left(x_{i+1}^{j}|D,z_{i+1}^{1:j},y_{i+1}^{1:j}\right)$ based on the *dynamics* for the horizontal interaction (including the static term and the term resulting from the processing of the row below):

$$p\left(x_{i+1}^{j:j+1}|D,z_{i+1}^{1:j},y_{i+1}^{1:j+1}\right)=p\left(x_{i+1}^{j}|z_{1:i+1}^{1:J},y_{1:i+1}^{1:j}\right)h\left(x_{i+1}^{j+1}|x_{i+1}^{j}\right)\frac{p\left(x_{i+1}^{j:j+1}|D\right)}{p\left(x_{i+1}^{j}|D\right)}s\left(x_{i+1}^{j+1}\right) \tag{6.60}$$

Moving from left to right in (6.60), the effect of the term $h\left(x_{i+1}^{j+1}|x_{i+1}^{j}\right)$ can be implemented using the normal Kalman filter prediction step (but without the usual integral):

$$\mathcal{N}\left(x_1;m_1,C_1\right)\mathcal{N}\left(x_2;Ax_1,Q\right)=\mathcal{N}\left(\begin{bmatrix}x_1\\x_2\end{bmatrix};m,C\right) \tag{6.61}$$

where

$$m = \begin{bmatrix} m_1 \\ Am_1 \end{bmatrix} \tag{6.62}$$

$$C = \begin{bmatrix} C_1 & AC_1 \\ C_1A^T & AC_1A^T + Q \end{bmatrix} \tag{6.63}$$

To multiply the resulting distribution by $p\left(x_{i+1}^{j:j+1}|D\right)$, one can use the standard result for multiplying two Gaussians:

$$\mathcal{N}(x; m_1, C_1)\mathcal{N}(x; m_2, C_2) = \mathcal{N}(x; m, C) \tag{6.64}$$

where

$$m = C\left(C_1^{-1}m_1 + C_2^{-1}m_2\right) \tag{6.65}$$

$$C^{-1} = C_1^{-1} + C_2^{-1} \tag{6.66}$$

Multiplication by the static term $s\left(x_{i+1}^{j+1}\right)$ in (6.60) can be considered as a partial observation of (half) the joint state. Hence, one can use the standard Kalman filter update equations to update the joint distribution of x_i^{j+1} and x_i^j:

$$\mathcal{N}(y; Hx, R)\mathcal{N}(x; m_1, C_1) \propto \mathcal{N}(x; m, C) \tag{6.67}$$

where

$$m = m_1 + CH^TR^{-1}(y - Hm_1) \tag{6.68}$$

$$C^{-1} = C_1^{-1} + H^TR^{-1}H \tag{6.69}$$

The division by $p\left(x_{i+1}^j|D\right)$ can then be thought of as removing the effect of a partial observation of (half) the joint state. Considering the exponential term in the definition of a Gaussian, multiplication of two Gaussian distributions results in addition of two Mahalanobis distances. Division therefore results in a subtraction of one Mahalanobis distance from another. Noting that $a-b = a+(-b)$, one can substitute $-R$ for R in the above. Hence, one can use a small modification of the standard Kalman filter update equations to update the distribution of x_i^{j+1} and x_i^j:

$$\frac{\mathcal{N}(x; m_1, C_1)}{\mathcal{N}(y; Hx, R)} \propto \mathcal{N}(x; m, C) \tag{6.70}$$

where

$$m = m_1 - CH^TR^{-1}(y - Hm_1) \tag{6.71}$$

$$C^{-1} = C_1^{-1} - H^TR^{-1}H \tag{6.72}$$

To complete the recursion, the update step for the measurement is simply implemented using the usual Kalman update equations.

$$p\left(x_{i+1}^{j:j+1}|D, z_{i+1}^{1:j+1}, y_{i+1}^{1:j+1}\right) \propto p\left(x_{i+1}^{j:j+1}|D, z_{i+1}^{1:j}, y_{i+1}^{1:j+1}\right) p\left(z_{i+1}^{j+1}|x_{i+1}^{j+1}\right) \tag{6.73}$$

This then gives rise to a recursion for $p\left(x_{i+1}^{j:j+1}|D, z_{i+1}^{1:j+1}, y_{i+1}^{1:j+1}\right)$.

Completely Parameterising the Updated Distribution

The same recursion can be used (backwards across the row and with a reverse horizontal dynamic to obtain the parameters of $p\left(x_{i+1}^{j:j+1}|D, z_{i+1}^{j+1:J}, y_{i+1}^{j+1:J}\right)$ and so $p\left(x_{i+1}^{j}|D, z_{i+1}^{j+1:J}, y_{i+1}^{j+1:J}\right)$, such that the parameters of $p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}, y_{1:i+1}^{1:J}\right)$ are calculated using a similar relation to that in the two-filter form of the Kalman Smoother:

$$p\left(x_{i+1}^{j-1:j}|z_{1:i+1}^{1:J}, y_{1:i+1}^{1:J}\right) = \int\int p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}, y_{1:i+1}^{1:J}\right) dx_{i+1}^{1:j-2} dx_{i+1}^{j+1:J} \tag{6.74}$$

$$\propto \frac{1}{p\left(x_{i+1}^{j}|D\right)} \int p\left(x_{i+1}^{1:j}|D, z_{i+1}^{1:j}, y_{i+1}^{1:j}\right) dx_{i+1}^{1:j-2} \int p\left(x_{i+1}^{j:J}|D, z_{i+1}^{j+1:J}, y_{i+1}^{j+1:J}\right) dx_{i+1}^{j+1:J} \tag{6.75}$$

$$\propto \frac{1}{p\left(x_{i+1}^{j}|D\right)} p\left(x_{i+1}^{j-1:j}|D, z_{i+1}^{1:j}, y_{i+1}^{1:j}\right) p\left(x_{i+1}^{j}|D, z_{i+1}^{j+1:J}, y_{i+1}^{j+1:J}\right) \tag{6.76}$$

To implement (6.76), one again uses the Kalman update equations. This results in all the parameters of $p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J}, y_{1:i+1}^{1:J}\right)$ which can then be used in the processing of the next row.

6.4.2 Kalman Smoother

The above method can be used to derive the parameters of $p\left(x_{i}^{1:J}|z_{1:i}^{1:J}, y_{1:I}^{1:J}\right)$. A backwards (vertically) filter can then be used to derive the parameters of $p\left(x_{i}^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right)$. Such a backward filter would be identical in its operation except that it would iterate down from the Ith row to the 1st row and would use the reversed (vertical) dynamics.

The remaining task is then to obtain the parameters of $p\left(x_{i}^{1:J}|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right)$, which can be carried out as follows:

$$p\left(x_{i}^{1:J}|z_{1:I}^{1:J}, y_{1:I}^{1:J}\right) \propto p\left(x_{i}^{1:J}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{1:J}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) \tag{6.77}$$

$$= p\left(x_{i}^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) \prod_{j=2}^{J} p\left(x_{i}^{j}|x_{i}^{j-1}, z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{1}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) \prod_{j=2}^{J} p\left(x_{i}^{j}|x_{i}^{j-1}, z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) \tag{6.78}$$

$$= p\left(x_{i}^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{1}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) \prod_{j=2}^{J} p\left(x_{i}^{j}|x_{i}^{j-1}, z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{j}|x_{i}^{j-1}, z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) \tag{6.79}$$

$$= p\left(x_{i}^{1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{1}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right) \prod_{j=2}^{J} \frac{p\left(x_{i}^{j-1:j}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{j-1:j}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right)}{p\left(x_{i}^{j-1}|z_{1:i}^{1:J}, y_{1:i}^{1:J}\right) p\left(x_{i}^{j-1}|z_{i+1:I}^{1:J}, y_{i+1:I}^{1:J}\right)} \tag{6.80}$$

This distribution is of the following form where the conditioning is based on the (U)pward measure-

ments, $z_{i+1:I}^{1:J}$ and $y_{i+1:I}^{1:J}$, and the (D)ownward measurements:

$$p\left(x_i^{1:J}|U,D\right) \propto p\left(x_i^1|U\right)p\left(x_i^1|D\right)\prod_{j=2}^{J}\frac{p\left(x_i^{j-1:j}|U\right)p\left(x_i^{j-1:j}|D\right)}{p\left(x_i^{j-1}|U\right)p\left(x_i^{j-1}|D\right)} \tag{6.81}$$

$$= p\left(x_i^1|U\right)p\left(x_i^1|D\right)\prod_{j=2}^{j'}\frac{p\left(x_i^{j-1:j}|U\right)p\left(x_i^{j-1:j}|D\right)}{p\left(x_i^{j-1}|U\right)p\left(x_i^{j-1}|D\right)}\prod_{j=j'+1}^{J}\frac{p\left(x_i^{j-1:j}|U\right)p\left(x_i^{j-1:j}|D\right)}{p\left(x_i^{j-1}|U\right)p\left(x_i^{j-1}|D\right)} \tag{6.82}$$

$$\propto \frac{p\left(x_i^{1:j'}|U,D\right)p\left(x_i^{j':J}|U,D\right)}{p\left(x_i^{j'}|U\right)p\left(x_i^{j'}|D\right)} \tag{6.83}$$

To calculate the parameters of this distribution, the expression is integrated and (A)nother pseudo-measurement used to ease the notation:

$$p\left(x_i^{j-1:j}|U,D\right) = \int\int p\left(x_i^{1:J}|U,D\right)dx_i^{1:j-2}dx_i^{j+1:J} \tag{6.84}$$

$$\propto \frac{1}{p\left(x_i^j|U\right)p\left(x_i^j|D\right)}\int p\left(x_i^{1:j}|U,D\right)dx_i^{1:j-2}\int p\left(x_i^{j:J}|U,D\right)dx_i^{j+1:J} \tag{6.85}$$

$$= \frac{1}{p\left(x_i^j|U\right)p\left(x_i^j|D\right)}p\left(x_i^{j-1:j}|A_i^{1:j-2},U,D\right)p\left(x_i^j|A_i^{j+1:J},U,D\right) \tag{6.86}$$

A recursion can then calculate $p\left(x_i^{j-1:j}|A_i^{1:j-2},U,D\right)$ by first using a marginalisation step:

$$p\left(x_i^j|A_i^{1:j-2},U,D\right) = \int p\left(x_i^{j-1:j}|A_i^{1:j-2},U,D\right)dx_i^{j-1} \tag{6.87}$$

This is then followed by a multiplication and division step:

$$p\left(x_i^{j:j+1}|A_i^{1:j-1},U,D\right) = p\left(x_i^{j:j+1}|U\right)p\left(x_i^{j:j+1}|D\right)p\left(x_i^j|A_i^{1:j-2},U,D\right)\frac{1}{p\left(x_i^j|U\right)p\left(x_i^j|D\right)} \tag{6.88}$$

Again, the operations are performed from left to right. Since $p\left(x_i^{j:j+1}|U\right)$ and $p\left(x_i^{j:j+1}|D\right)$ are both Gaussian, the standard result for multiplying two Gaussians can be used. If one considers the joint state of x_i^{j+1} and x_i^j, then multiplication by the term $p\left(x_i^j|A_i^{1:j-2},U,D\right)$ in (6.88) can be considered as a partial observation of (half) the joint state. Hence, one can use the standard Kalman filter update equations to update the distribution of x_i^{j+1} and x_i^j. The division by each of the terms $p\left(x_i^j|U\right)$ and $p\left(x_i^j|D\right)$ in (6.88) can then be thought of as removing the effect of a partial observation of (half) the joint state and so the modified version of the standard Kalman filter update equations can the be used. This then gives rise to a recursion for $p\left(x_i^{j-1:j}|A_i^{1:j-2},U,D\right)$. The same recursion can be used (backwards across the row) to obtain $p\left(x_{i+1}^{1:J}|z_{1:i+1}^{1:J},y_{1:i+1}^{1:J}\right)$. To implement (6.86), one again uses the Kalman update equations, which results in the parameters of $p\left(x_i^{1:J}|z_{1:I}^{1:J},y_{1:I}^{1:J}\right)$.

6.5 Results

To demonstrate the approach, an edge-detection model is considered as an exemplar model using which one might wish to analyse an image. The aim of this example is not to demonstrate an improved edge

detector, but to demonstrate the algorithmic performance. The model is based on the constant velocity model used extensively in the tracking literature[10]. The state at every location in the image consists of an intensity and a vertical and a horizontal *velocity*. Only the intensity is observed at each location, so the state is partially observed. The parameters of the model are then as follows:

$$A_V = \begin{bmatrix} 1 & T & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.89}$$

$$Q_V = \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} & 0 \\ \frac{T^2}{2} & T & 0 \\ 0 & 0 & 0 \end{bmatrix} \sigma_{VV}^2 + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & T \end{bmatrix} \sigma_{VH}^2 \tag{6.90}$$

$$A_H = \begin{bmatrix} 1 & 0 & T \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.91}$$

$$Q_H = \begin{bmatrix} \frac{T^3}{3} & 0 & \frac{T^2}{2} \\ 0 & 0 & 0 \\ \frac{T^2}{2} & 0 & T \end{bmatrix} \sigma_{HH}^2 + \begin{bmatrix} 0 & 0 & 0 \\ 0 & T & 0 \\ 0 & 0 & 0 \end{bmatrix} \sigma_{HV}^2 \tag{6.92}$$

$$m_s = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{6.93}$$

$$C_s = \begin{bmatrix} \sigma_C^2 & 0 & 0 \\ 0 & \sigma_C^2 & 0 \\ 0 & 0 & \sigma_C^2 \end{bmatrix} \tag{6.94}$$

$$H_z = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \tag{6.95}$$

$$R = \sigma_R^2 \tag{6.96}$$

In the tracking literature, T is the time between measurements and so defines the units of the velocity. Here, that choice is taken to be arbitrary and so $T = 1$.

The raw image considered is shown in figure 6.1(a). The intensity of the image pixels took integer values in the range $0 - 255$. The values used for the parameters of the model were $\sigma_{HH}^2 = \sigma_{VV}^2 = \sigma_{HV}^2 = \sigma_{VH}^2 = \sigma_C^2 = 10$. For edge detection, the magnitude of the vector composed of the vertical and horizontal velocities is assumed to be the quantity of interest. An image of the value of this quantity for the mean state estimate at every point in the image is shown in figure 6.1(b). The edges were inferred assuming that the measurement noise was Gaussian with the same variance as a uniform error (due to quantisation) across an integer value for the intensity of each pixel (such that $\sigma_R^2 = \frac{1}{12}$). White noise with standard deviation of $\sqrt{50}$ was then added to the original image; this generated the noisy image shown in figure 6.2(a). Then assuming the measurement noise was $\sigma_R^2 = 50$, resulted in the image of edges shown in figure 6.2(b). The images are made up of 256 pixels by 256 pixels and the processing was

carried out on a desktop PC running MATLAB in approximately 10 minutes for each such image. While this is much more than would be required by standard image processing algorithms, the key contribution of this chapter is the fact that the proposed algorithm's complexity grows linearly with the number of pixels and, because it is an analytically exact Bayesian solution to the problem posed, the algorithm gives good results in environments with a low Signal to Noise ratio. Alternative implementations could well improve the speed significantly.

6.6 Conclusions

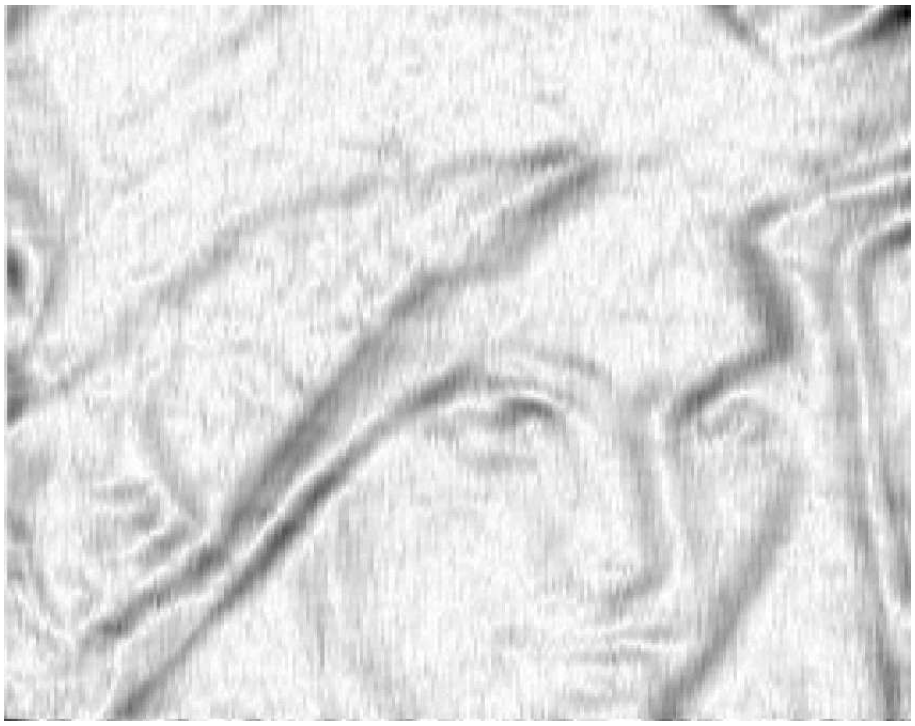
An efficient scheme was proposed for analytically exact inference in conditionally Gaussian Markov random fields. The cost of the approach is linear in the number of pixels in an image. To demonstrate the application of the approach, a large noisy image was processed using the devised algorithm. This opens up opportunities for the analysis of spatial (and volumetric) data using algorithms from the time series literature. Future work will look into how to extend the analysis to consider nonlinear and non-Gaussian models using Gaussian mixtures and Monte Carlo algorithms and to the analysis of data residing in higher dimensional spaces.

(a) Clean image.

(b) Image of edges deduced from clean image.

Figure 6.1: *Analysis of Clean Image.*

(a) Image plus noise.



(b) Image of edges deduced from image plus noise.

Figure 6.2: *Analysis of Noisy Image.*

Conclusions 7

There exists a wealth of algorithms that have been developed for the analysis of sequential problems. There are a number of areas where these algorithms could be extended so as to minimise the impact of the approximations that inevitably must be introduced to make the solutions practicable: mechanisms for efficiently and robustly tracking a large uncertain number of targets in ambiguous environments; efficient methods for ensuring that data association algorithms solve as easy an assignment problem as possible; methods for making explicit the implicit approximations in a number of approaches.

To make best use of the wealth of these sequential algorithms, the models used need to be derived carefully. A method has been proposed that is both easy to understand and gives rise to models that can indeed make best use of these algorithms. The method is closely related to that proposed by another author[110], which gives the same results for simpler models, but is arguably less accessible to a more applied audience and less applicable to the larger systems that are often of interest. Further work is needed to exploit the close relations between these two bodies of work.

An architecture has been proposed for efficient and robust joint tracking and sequential classification. The approach was demonstrated through the application to semi-Markov models governing the behaviour of a manoeuvring target. Generalisation of the model and algorithm to the analysis of other data with sequential structure, such as images, is to be pursued.

The tracking of large numbers of targets is complicated by the ambiguous assignment of measurements to targets. The number of candidate assignments grows exponentially with the number of targets. An exact and efficient method has been proposed to marginalising out this assignment. The solution considers a sufficient statistic of the assignment of some of the targets to the measurements in terms of the impact this has on the future targets. This concept of a sufficient statistic is exactly that employed in a sequential context where the sufficient statistic represents the dependence of the future on the past. Fast mechanisms for defining an ordering of the targets such that the sequential processing of the targets has maximum benefit need to be devised.

Images can be considered as a sequence of columns. Each column can be considered as a sequence of pixels. Hence it has been possible to devise an efficient and robust algorithm that processes images in a time that scales linearly with the number of pixels, while avoiding any need for large amounts of computation or a requirement of low noise. While the focus was on linear Gaussian models and two-dimensional data, since other high dimensional data can be recursively considered as sequential lower dimensional problems, this makes it possible for future work to generalise seminal sequential algorithms to the efficient and robust analysis of volumes of data evolving in time and so on.

It is the problem that dictates the structure of an efficient solution. A large number of problems

have structure that can be thought of as sequential. Indeed, this thesis has shown that it is possible to construct efficient sequential solutions to problems with sequential structure whether or not they might traditionally be considered sequential problems.

Bibliography

[1] C Andrieu, M Davy, and A Doucet. Efficient particle filtering for jump Markov systems. In *Proceedings of IEEE ICASSP 2002, Orlando, USA*, May 2002.

[2] A S Arulampalam, S Maskell, N J Gordon, and T Clapp. A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[3] S Arulampalam and B Ristic. Comparison of the particle filter with range parameterised and modified polar EKF's for angle-only tracking. In *SPIE: Signal and Data Processing of Small Targets*, volume 4048, pages 288–299, 2000.

[4] D Avitzour. Stochastic simulation Bayesian approach to multitarget tracking. *IEE Proceedings-Radar Sonar and Navigation*, 142(2):41–44, 1995.

[5] P M Baggesnstoss. The PDF projection theorem and the class-specific method. *IEEE Transactions on Signal Processing*, 51(3):672–687, March 2003.

[6] D Ballantyne, H Chan, and M Kouritzin. A branching particle-based nonlinear filter for multi-target tracking. In *Proceedings of 4th International Conference on Information Fusion*, August 2001.

[7] Y Bar-Shalom. Unbiased converted measurements for tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 34:1023–1027, 1998.

[8] Y Bar-Shalom and W D Blair, editors. *Multitarget-Multisensor Tracking: Applications and Advances Volume III*. Artech House, Boston/London, 2000.

[9] Y Bar-Shalom and A G Jaffer. Adaptive nonlinear filtering for tracking with measurements of uncertain origin. In *Proceedings of IEEE Conference on Decision and Control, New Orleans, LA*, December 1972.

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

[11] Y Bar-Shalom, H Shertukde, and K Pattipati. Precision target tracking for small extended objects. In *Optical Engineering*, volume 29(2), February 1989.

[12] Y Bar-Shalom, H Shertukde, and K Pattipati. Use of measurement from an imaging sensor for precision target tracking. In *IEEE Transactions on Aerospace and Electronic Systems*, volume 25(6), November 1989.

[13] R H Bishop and A C Antoulas. Nonlinear approach to aircraft tracking problem. In *Journal of Guidance, Control and Dynamics*, volume 17(5), pages 1124–1130, September 1994.

[14] S Blackman and R Popoli. *Deisgn and Analysis of Modern Tracking Systems*. Artech House, 1999.

[15] A Blake and M Isard. *Active Contours*. Springer-Verlag, 1998.

[16] H A P Blom. An efficient filter for abruptly changing systems. In *IEEE Proceedings-Decision and Control*, volume 23, pages 656–658, 1984.

[17] H A P Blom and Y Bar-Shalom. The interacting model algorithm for systems with Markovian switching coefficients. In *IEEE Transactions on Automatic Control*, volume 33, pages 780–783, 1988.

[18] Y Boers, J N Driessen, F Verschure, W P M H Heemels, and A Juloski. A multi target track before detect application. In *Proceedings of WOMOT 2003*, 2003.

[19] M Briers, S Maskell, and R Wright. A rao-blackwellised unscented Kalman filter. In *Proceedings of 6th International Conference on Information Fusion*, 2003.

[20] L Campo, P Mookerjee, and Y Bar-Shalom. State estimation for systems with a sojourn-time-dependent Markov switching model. *IEEE Transactions on Automatic Control*, 36:238–243, February 1991.

[21] J Carpenter, P Clifford, and P Fearnhead. Improved particle filter for non-linear problems. *IEE Proceedings on Radar and Sonar Navigation,*, 146(1):2–7, February 1999.

[22] L Chen, M Wainwright, M Cetin, and A Willsky. Multitarget-multisensor data association using the tree-reweighted max-product algorithm. In *Proceedings of SPIE Aerosense conference, Orlando, FL*, March 2003.

[23] T Clapp and S Godsill. Improvement strategies for monte carlo particle filters. In A Doucet, J F G de Freitas, and N J Gordon, editors, *Sequential Monte Carlo Methods in Practice*. Springer-Verlag: New York, 2001.

[24] D R Cox and V Isham. *Point Processes*. Chapman and Hall, 1980.

[25] D Crisan, P Del Moral, and T J Lyons. Non-linear filtering using branching and interacting particle systems. *Markov Processes and Related Fields*, 5(3):293–319, 1999.

[26] A Doucet and C Andrieu. Particle filtering for partially observed gaussian state space models. *J. Royal Statistical Society B (Methodological)*, 64(4):827–836, 2002.

[27] A Doucet, J F G de Freitas, and N J Gordon. An introduction to sequential Monte Carlo methods. In A Doucet, J F G de Freitas, and N J Gordon, editors, *Sequential Monte Carlo Methods in Practice*, pages 3–14. Springer-Verlag: New York, 2001.

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

[29] A Doucet, S Godsill, and C Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[30] A Doucet, N Gordon, and V Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49(3):613–624, 2001.

[31] A Doucet, N J Gordon, and V Krishnamurthy. Sequential simulation-basd estimation of jump Markov linear systems. In *Proceedings of IEEE Conference on Decision and Control*, volume 2, pages 1166–1171, December 2000.

[32] H Driessen and Y Boers. Multiple model multiple hypothesis filter for tracking maneouvring targets. In *Proceedings of SPIE; Signal Processing of Small Target*, August 2001.

[33] J Durbin and S J Koopman. Monte Carlo maximum likelihood estimation for non-Gaussian state space models. *Biometrika*, 84:1403–1412, 1997.

[34] J Durbin and S J Koopman. *Time series Analysis by State Space Methods*. Oxford University Press, 2001.

[35] A Einstein. *Investigations on the Theory of the Brownian Movement*. Dover, 1956.

[36] P Fearnhead. *Sequential Monte Carlo Methods in Filter Theory*. PhD thesis, Oxford University, 1998.

[37] G D Forney. The viterbi algorithm. In *Proceedings of the IEEE*, volume 61(3), pages 268–278, March 1973.

[38] T E Fortmann, Y Bar-Shalom, and M Scheffe. Multi-target tracking using joint probabilistic data association. In *Proceedings of the 1980 IEEE Conference on Decision and Control, Albuquerque*, pages 807–812, 1980.

[39] W R Gilks and C Berzuini. Following a moving target – Monte Carlo inference for dynamic baysian models. *Journal of the Royal Statistical Society, B*, 63:127–146, 2001.

[40] S Godsill, A Doucet, and M West. Maximum a posteriori sequence estimation using Monte Carlo particle filters. *Annals of the Institute of Statistical Mathematics*, 53(1):82–96, March 2001.

[41] N Gordon, J Percival, and Robinson M. The Kalman-Levy filter and heavy-tailed models for tracking manoeuvring targets. In *Proceedings of Fusion 2003*, 2003.

[42] N Gordon, D Salmond, and A Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F Radar, Sonar and Navigation*, 140(2):107–113, 1993.

[43] N J Gordon, S Maskell, and T Kirubarajan. Efficient particle filters for joint tracking and classification. In *Procedings of SPIE: Signal and Data Processing of Small Targets*, pages 439–449, 2002.

[44] F Gustafsson, F Gunnarsson, N Bergman, U Forssell, J Jansson, R Karlsson, and P-J Nordlund. Particle filers for positioning, navigation and tracking. *IEEE Transactions on Signal Processing*, 50(2):425–437, February 2002.

[45] J E Handschin and D Q Mayne. Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. In *International Journal of Control*, volume 9(5), pages 547–559, 1969.

[46] A C Harvey. *Forcasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press, Cambridge, 1989.

[47] R E Helmick, D Blair, and S A Hoffman. Fixed-interval smoothing for Markovian switching systems. In *IEEE Transactions on Information Theory*, volume 41(6), pages 1845–1855, 1995.

[48] Y C Ho and R C K Lee. A Bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, AC(9):333–339, 1964.

[49] C Hue, J-P Le-Cadre, and P Perez. Sequential monte carlo methods for multiple target tracking and data fusion. *IEEE Transactions on Signal Processing*, 50(2), February 2002.

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

[51] S Julier. A skewed approach to filtering. In *Proceedings of SPIE Conference on Signal Processing of Small Targets*, volume 3373, pages 271–282, 1998.

[52] S Julier. The scaled unscented transformation. To appear in Automatica, 2000.

[53] S J Julier and J K Uhlmann. A consistent, debiased method for converting between polar and cartesian coordinate systems. *SPIE Proceedings of AeroSense: Acquisition, Tracking and Pointing XI, vol. 3086*, page 110121, 1997.

[54] S J Julier and J K Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *SPIE Proceedings of AeroSense: Aerospace/Defence Sensing, Simulation and Controls*, 1997.

[55] R E Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–46, 1960.

[56] G Kitagawa. Monte Carlo filter and smoother for non-Gaussian non-linear state space models. *Journal Of Computational and Graphical Statistics*, 5(1):1–25, 1996.

[57] W Koch. Retrodiction for Bayesian multiple-hypothesis/multiple-target tracking in densely cluttered environment. In *SPIE: Signal and Data Processing of Small Targets*, pages 429–440, 1996.

[58] C V Kumar, R N Rajagopal, and R Kiran. Passive DOA tracking using unscented Kalman filter. *CRL Technical Journal Vol.3 No.3 December*, 2001.

[59] M Lavine. Another look at conditionally Gaussian Markov random fields. In *Bayesian Statistics: Proceedings of the Sixth International Meeting*, 1998.

[60] F Le Gland and N Oudjane. Stability and uniform approximation of nonlinear filters using the Hilbert metric, and application to particle filters. Technical Report RR-4215, INRIA, 2001.

[61] T Lefebvre, H Bruyninckx, and J De Schutter. Comment on a new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 2002.

[62] D Lerro and Y Bar-Shalom. Tracking with debiased consistent converted measurements vs. EKF. *IEEE Transactions on Aerospace and Electronic Systems*, 29(3):1015–1022, July 1993.

[63] X R Li and Y Bar-Shalom. Multiple-model estimation with variable structure. In *IEEE Transactions on Automatic Control*, volume AC-41(4), pages 478–493, April 1996.

[64] X R Li and V P Jilkov. A survey of maneuvering target tracking. In *SPIE: Signal and Data Processing of Small Targets*, volume 4048, April 2000.

[65] J S Liu and R Chen. Sequential monte carlo methods for dynamical systems. In *Journal of the American Statistical Association*, volume 93, pages 1032–1044, 1998.

[66] J MacCormick and A Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proc. International Conference on Computer Vision, Kerkyra, Corfu*, pages 572–578, 1999.

[67] R Mahler. An introduction to multisource/multitarget statistics and its applications. Technical report, Lockheed Martin, 2000.

[68] X Mao. *Stochastic Differential Equations and Application*. Horwood series in Mathematics and Applications, 1997.

[69] A Marrs, S Maskell, and Y Bar-Shalom. Expected likelihood for tracking in clutter with particle filters. In O Drummond, editor, *Proceedings of SPIE Conference on Signal Processing of Small Targets*, pages 230–239, 2002.

[70] F Martinerie and P Forster. Data association and tracking using hidden Markov models and dynamic programming. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, 1992.

[71] S Maskell. Using Laplace transforms to derive turning models for tracking. Submitted to IEEE Transactions on Aerospace and Electronic Systems, 2000.

[72] S Maskell. Efficient hypothesis management - with application to multi-target tracking. Technical report, DERA, June 2001. RESTRICTED - COMMERCIAL.

[73] S Maskell. Multi-Sensor Management, First Year PhD Report. Technical report, Cambridge University Engineering Department, June 2001.

[74] S Maskell. Basics of the particle filter. In N Shephard and A Harvey, editors, *State Space and Unobserved Component Models*. Cambridge University Press, 2003.

[75] S Maskell. Signal processing with reduced combinatorial complexity, July 2003. Patent Reference: 0315349.1.

[76] S Maskell. Tracking manoevring targets and classification of their maneovrability. Submitted to Special Issue of EURASIP Journal on Applied Signal Processing on Particle Filtering in Signal Processing, 2003.

[77] S Maskell, N Gordon, M Rollason, and D Salmond. Efficient multitarget tracking using particle filters. *Proceedings of SPIE Conference on Signal Processing of Small Targets*, pages 251–262, 2002.

[78] S Maskell, N Gordon, M Rollason, and D Salmond. Efficient multitarget tracking using particle filters. *Journal Image and Vision Computing*, 21(10):931–939, September 2003.

[79] S Maskell and N J Gordon. A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. In *Proceedings of IEE Colloquim*, 2002.

[80] S Maskell, M Orton, and N Gordon. Efficient inference for conditionally Gaussian Markov random fields. Technical report, Cambridge University Engineering Department, 2002.

[81] M Montemerlo, S Thrun, D Koller, and B Wegbreit. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002.

[82] M R Morelande and S Challa. A multitarget tracking alogorithm based on random sets. In *Proceedings of Fusion 2003*, 2003.

[83] D Musicki, R Evans, and B La Scala. Integrated track splitting suite of target tracking filters. In *Proceedings of Fusion 2003*, 2003.

[84] C Musso, Oudjane N, and F LeGland. Improving regularised particle filters. In A Doucet, J F G de Freitas, and N J Gordon, editors, *Sequential Monte Carlo Methods in Practice*, pages 247–271. Springer-Verlag: New York, 2001.

[85] H Niederreiter. *Random Number Genereation and Quasi-Monte Carlo Methods*. SIAM, Philadelphia, PA, 1992.

[86] H Niederreiter and P J Shiue, editors. *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*. Springer-Verlag New York Inc., 1995.

[87] B Øksendal. *Stochastic Differential Equations - An Introduction with Applications*. Springer, 5th edition, 1998.

[88] D Ormoneit, C Lemieux, and D Fleet. Lattice particle filters. In *Proceedings of Uncertainty in AI (UAI) 2001*, pages 395–402, 2001.

[89] M Orton and W Fitzgerald. A Bayesian approach to tracking multiple targets using sensor arrays and particle filters. *IEEE Transactions on Signal Processing*, 50(2):216–223, February 2002.

[90] M Orton and A Marrs. A Bayesian approach to multi-target tracking data fusion with out-of sequence measurements. Technical report, Cambridge University Engineering Department, 2001.

[91] N Oudjane and C Musso. Progressive correction for regularized particle filters. In *Proceedings of 3rd International Conference on Information Fusion*, 2000.

[92] T Ozaki. A local linearization approach to nonlinear filtering. *Int. J. on Control*, 57(1):75–96, 1993.

[93] L Y Pao. Multisensor multitarget mixture reduction algorithms for target tracking. *Journal of Guidance Control and Dynamics*, 17:1205–1211, 1994.

[94] K R Pattipati, Y Bar-Shalom, T Kiruba'rajan, and S Deb. A generalized s-d assignment algorithm for multisensor-multitarget state estimation. In *INFORMS*, October 1998.

[95] K R Pattipati, S Deb, Y Bar-Shalom, and R Washburn. A new relaxaton algorithm and passive sensor data association. In *IEEE Transactions on Automatic Control*, volume 37(2), February 1992.

[96] V Philomin, R Duraiswami, and L Davis. Quasi-random sampling for condensation. In *Proceedings of European Conference on Computer Vision*, 2000.

[97] M Pitt and N Shephard. Filtering via simulation: Auxiliary particle filters. In *Journal of the American Statistical Association*, volume 94(446), pages 590–599, 1999.

[98] A B Poore, N Rijavec, T N Barker, and M Munger. Data association problems posed as multi-dimensional assignment problems: Algorithm development. In *SPIE Proceedings*, pages 172–182, 1993.

[99] H A Quevedo, S S Blackman, T Nichols, R J Dempster, and R Wenski. Reducing mht computational requirements through use of cheap jpda methods. In *Proceedings of SPIE; Signal Processing of Small Target*, August 2001.

[100] L R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77(2), pages 257–285, February 1989.

[101] L R Rabiner and B H Juang. An introduction to hidden Markov models. In *IEEE Acoustics, Speech and Signal Processing Magazine*, pages 4–16, January 1986.

[102] B D Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control, vol. AC-24, no. 6*, pages 843–854, 1979.

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

[104] H Rue. Fast sampling of Gaussian Markov random fields with applications. Technical report, Norwegian University of Science and Technology, Trondheim, 2000. Technical Report.

[105] D J Salmond. Mixture reduction for target tracking in clutter. In *SPIE Vol. 1305, Signal and Data Processing of Small Targets*, pages 434–445, 1990.

[106] D Schulz, W Burgard, D Fox, and A Cremers. Tracking multiple moving objects with a mobile robot. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, December 2001.

[107] S R Searle. *Matrix Algebra Useful for Statistics*. Wiley, 1982.

[108] N Shephard and M Pitt. Likelihood analysis of non-Gaussian measurement time series. *Biometrika*, 84(3):653–667, 1997.

[109] H Shertukde and Y Bar-Shalom. Tracking of crossing targets with imaging sensors. In *IEEE Transactions on Aerospace and Electronic Systems*, volume 27(4), July 1991.

[110] I Shoji and T Ozaki. A statistical method of estimation and simulation for systems of stochastic differential equations. *Biometrika*, 85:240–243, 1998.

[111] R H Shumway and D S Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[112] H Sidenbladh. Multi-target particle filtering for the probability hypothesis density. In *Proceedings of Fusion 2003*, 2003.

[113] D Sornette and K Ide. The Kalman-Levy filter. *Physica D*, 151:142–174, 2001.

[114] R L Streit and R F Barrett. Frequency line tracking using hidden Markov models. In *IEEE Transactions Accoustics, Speech & Signal Processing*, volume 38(4), pages 586–598, April 1990.

[115] R L Streit and T E Luginbuhl. Probabilistic multi-hypothesis tracking. Submitted to: IEEE Transactions Automatic Control, 1999.

[116] D D Sworder, R Vojak, R G Hutchins, and M Kent. Renewal models for manoevring targets. *IEEE Transactions on Aerospace and Electronic Systems*, 31(1):168–150, January 1995.

[117] R van der Merwe, A Doucet, J F G de Freitas, and E Wan. The unscented particle filter. In *Advances in Neural Information Processing Systems (NIPS13)*, pages 584–590, December 2000.

[118] J R Van Zandt. A more robust unscented transform. *SPIE Proceedings of AeroSense: Signal and Data Processing of Small Targets*, 2001.

[119] J Vermaak, A Doucet, and P Perez. Maintaining multi-modality through mixture tracking. In *Proceedings of ICCV 2003*, volume II, pages 1110–1116, 2003.

[120] J Vermaak, S J Godsill, and A Doucet. Radial basis function regresssion using trans-dimensional sequential monte carlo. In *Proceedings of 12th IEEE Workshop on Statistical Signal Processing*, pages 525–528, 2003.

[121] B-N Vo, S Singh, and A Doucet. Sequential monte carlo implementation of the PHD filter for multi-target tracking. In *Proceedings of Fusion 2003*, 2003.

[122] E A Wan and R van der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC)*, 2000.

[123] E A Wan and R van der Merwe. *The Unscented Kalman Filter, to appear in Kalman Filtering and Neural Networks, Chapter 7, Edited by Simon Haykin*. Wiley Publishing, 2001.

[124] H Wang, T Kirubarajan, and Y Bar-Shalom. Precision large scale air traffic surveillance using IMM/assignment estimators. *IEEE Transactions on Aerospace and Electronic Systems*, January 1999.

[125] J R Werthmann. A step-by-step description of a computationally efficient version of multiple hypothesis tracking. In *SPIE Proceedings*, 1992.

[126] M West and J Harrison. *Bayesian Forecasting and Dynamic Models*. Springer-Verlag, New York, 2nd edition, 1997. Springer Series in Statistics.

[127] B Zhou and N Bose. An efficient algorithm for data association in multitarget tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 31(1):458–468, January 1995.