EURASIP Journal on Applied Signal Processing 2004:15, 2339–2350

Joint Tracking of Manoeuvring Targets and Classification of Their Manoeuvrability

Simon Maskell

QinetiQ Ltd, St. Andrews Road, Malvern, Worcestershire WR14 3PS, UK
Email: smaskell@signal.qinetiq.com

Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK

Received 30 May 2003; Revised 23 January 2004

Semi-Markov models are 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 models make it possible to define the distribution explicitly. Such models can be used to describe the behaviour of manoeuvring targets, and particle filtering can then facilitate tracking. An architecture is 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.

Keywords and phrases: tracking, classification, manoeuvring targets, particle filtering.

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 [1]) 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 [2], 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 [3], 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 paper 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. The author asserts that the restrictions of the aforementioned approaches currently limit the use of semi-Markov models in tracking systems and that the improved modelling (and so estimation) accuracy that semi-Markov models make possible has not been realised up to now.

This paper further considers the problem of both tracking and classifying targets. As discussed in [4], 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 knowledge of 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, if the algorithm forgets any errors, it must converge. 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.

There have been some previous approaches to solving the problem of joint tracking and identification that have been based on both grid-based approximations [5] and particle filters [6, 7]. An important failing of these implementations is that target classes with temporarily low likelihoods can end up being permanently lost. As a consequence of this same feature of the algorithms, these implementations cannot recover from any miscalculations and are not robust. This robustness issue has been addressed by stratifying the classifier [4]; one uses separate filters to track the target for each class (i.e., one might use a particle filter for one class and a Kalman filter for another) and then combines the outputs to estimate the class membership probabilities and so classification of the target. This architecture does enable different state spaces and filters to be used for each class, but has the deficiency that this choice could introduce biases and so systematic errors. So, the approach taken here is to adopt a single state space common to all the classes and a single (particle) filter, but to then attempt to make the filter as efficient as possible while maintaining robustness. This ability to make the filter efficient by exploiting the structure of the problem in the structure of the solution is the motivation for the use of a particle filter specifically.

This paper demonstrates this methodology by considering 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 nonlinear and non-Gaussian, the particle-filtering methodology [8] is adopted for solving this joint tracking and classification problem. The particle-filter represents uncertainty using a set of samples. Here, each of the samples represent different hypotheses for the sojourns times and state transitions. Since there is uncertainty over both how many transitions occurred and when they occurred, the particles represent the diversity over the number of transitions and their timing. Hence, the particles differ in dimensionality. This is different from the usual case for which the dimensionality of all the particles is the same. Indeed, this application of the particle filter is a special case of the generic framework developed concurrently by other researchers [9]. The approach described here exploits the specifics of the semi-Markov model, but the reader interested in the more generic aspects of the problem is referred to [9].

Since, if the sojourn times are known, the system is linear and Gaussian, the Kalman filter is used to deduce the parameters of the uncertainty over target state given the hypothesised history of sojourns. So, the particle filter is only used for the difficult part of the problem—that of deducing the timings of the sojourn ends—and the filter operates much like a multiple hypothesis tracker, with hypotheses in the (continuous) space of transition times. To make this more explicit, it should be emphasised that the complexity of the particle filter is not being increased by using semi-Markov models, but rather particle filters are being applied to the problem associated with semi-Markov models. The resulting computational cost is roughly equivalent to one Kalman filter per particle and in the example considered in Section 6 just 25 particles were used for each of the three classes.[1] The author believes that this computational cost is not excessive and that, in applications for which it is beneficial to capitalise on the use of semi-Markov models—which the author believes to be numerous—the approach is practically useful. However, this issue of the trade-off between the computational cost and the resulting performance for specific applications is not the focus of this paper; here the focus is on proposing the generic methodology. For this reason, a simple yet challenging, rather than necessarily practically useful, example is used to demonstrate that the methodology has merit.

A crucial element of the particle filter is the proposal distribution, the method by which each new sample is proposed from the old samples. Expedient choice of proposal distribution can make it possible to drastically reduce the number of particles necessary to achieve a certain level of performance. Often, the trade-off between complexity and performance is such that this reduction in the number of particles outweighs any additional computation necessary to use the more expedient proposal distributions. So, the choice of proposal distribution can be motivated as a method for reducing computational expense. Here, however, if as few errors as possible, are to be introduced as is critically important when conducting joint tracking and classification, it is crucial that the proposal distribution 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. Note that this calculation is not computationally expensive, but provides information that can be used to significantly improve the efficiency of the filter.

So, the particles are used to estimate the manoeuvres and a Kalman filter is 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. The motivation for this architecture is the need to simultaneously achieve robustness and efficiency.

This paper is structured as follows: Section 2 begins by introducing the notation and the semi-Markov model

[1] This number is small and one might use more in practical situations, but the point is that the number of particles is not large and so the computational expense is roughly comparable to other existing algorithms.

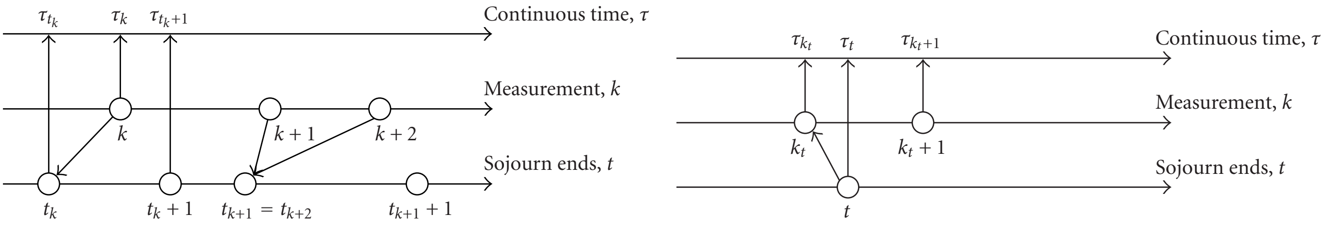


FIGURE 1: Diagram showing the relationship between continuous time, the time when measurements were received, and the time of sojourn ends. The circles represent the receipt of measurements or the start of a sojourn.

structure that is used. Section 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. Then, in Section 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 6 and some conclusions are drawn in Section 7.

TABLE 1: Definition of notation.

Notation	Definition
τ_k	Continuous time relating to kth measurement
τ_t	Continuous time relating to tth sojourn time
t_k	Sojourn prior to kth measurement; so that $\tau_{t_k} \leq \tau_k \leq \tau_{t_k+1}$
k_t	Measurement prior to tth sojourn; so that $\tau_{k_t} \leq \tau_t \leq \tau_{k_t+1}$
s_t	Manoeuvre regime for $\tau_t < \tau < \tau_{t+1}$

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 1 while Figure 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 $s_t = s_{t-1}$;

$$p(s_t|s_{t-1}) = p(s_t|s_{1:t-1}), \tag{1}$$

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; that is, $p(s_t|s_{t-1})$ 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 $g(\tau-\tau_t|s_t)$:

$$p(\tau_{t+1}|\tau_t, s_t) \triangleq g(\tau-\tau_t|s_t). \tag{2}$$

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

$$p(x_\tau|x_{\tau_t}, s_t, \tau_{t+1} > \tau) \triangleq f(x_\tau|x_{\tau_t}, s_t). \tag{3}$$

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:

$$p(y_k|x_k) = p(y_k|x_{\tau_{1:\tau_k}}) = p(y_k|x_{\tau_k}). \tag{4}$$

This formulation makes it straightforward to then form a dynamic model for $s_{1:t_k}$ process and $\tau_{1:t_k}$ as follows:

$$p(s_{1:t_k}, \tau_{1:t_k}) = \left(\prod_{t'=2}^{t_k} p(s_{t'}|s_{t'-1})p(\tau_{t'}|\tau_{t'-1}, s_{t-1})\right)p(s_1)p(\tau_1), \tag{5}$$

where $p(s_1)$ is the initial prior on the state of the sojourn time (which we later assume to be uniform) and $p(\tau_1)$ 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:

$$\begin{aligned}
p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}|s_{1:t_{k-1}}, \tau_{1:t_{k-1}}) \\
&\propto \frac{p(s_{1:t_k}, \tau_{1:t_k})}{p(s_{1:t_{k-1}}, \tau_{1:t_{k-1}})} \\
&= \frac{\left(\prod_{t'=2}^{t_k} p(s_{t'}|s_{t'-1})p(\tau_{t'}|\tau_{t'-1}, s_{t-1})\right)p(s_1)p(\tau_1)}{\left(\prod_{t'=2}^{t_{k-1}} p(s_{t'}|s_{t'-1})p(\tau_{t'}|\tau_{t'-1}, s_{t-1})\right)p(s_1)p(\tau_1)} \\
&= \prod_{t'=t_{k-1}+1}^{t_k} p(s_{t'}|s_{t'-1})p(\tau_{t'}|\tau_{t'-1}, s_{t-1}),
\end{aligned} \tag{6}$$

where $A \setminus B$ is the set A without the elements of the set B. Note that in this case $\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}$ could be the empty set in which case, $p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}) = 1$.

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

$$\begin{aligned}
& p(s_{1:t_k}, x_k, \tau_{1:t_k} | y_{1:k}) \\
& \quad \propto p(s_{1:t_k}, \tau_{1:t_k}) p(x_k, y_{1:k} | s_{1:t_k}, \tau_{1:t_k}) \\
& \quad = p(s_{1:t_k}, \tau_{1:t_k}) p(x_k | s_{1:t_k}, \tau_{1:t_k}) p(y_{1:k} | x_k) \\
& \quad = p(s_{1:t_k}, \tau_{1:t_k}) p(x_{\tau_k} | x_{\tau_{t_k}}, s_{t_k}) \left(\prod_{t'=2}^{t_k} p(x_{\tau_{t'}} | x_{\tau_{t'-1}}, s_{t'-1}) \right) \\
& \qquad \times p(x_{\tau_1}) \prod_{k'=1}^{k} p(y_{k'} | x_{\tau_{k'}}) \\
& \quad \propto \underbrace{p(s_{1:t_{k-1}}, x_{k-1}, \tau_{1:t_{k-1}} | y_{1:k-1})}_{\text{The posterior at } k-1} \\
& \qquad \times \underbrace{p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}})}_{\text{Evolution of semi-Markov model}} \\
& \qquad \times \underbrace{p(y_k | x_{\tau_k})}_{\text{Likelihood}} \underbrace{\frac{p(x_{\tau_k} | x_{\tau_{t_k}}, s_{t_k})}{p(x_{\tau_{k-1}} | x_{\tau_{t_{k-1}}}, s_{t_{k-1}})}}_{\text{Effect on } x_\tau \text{ of incomplete regimes}} \\
& \qquad \times \underbrace{\left(\prod_{t'=t_{k-1}+1}^{t_k} p(x_{\tau_{t'}} | x_{\tau_{t'-1}}, s_{t'-1}) \right)}_{\text{Effect on } x_\tau \text{ of sojourns between } k-1 \text{ and } k}.
\end{aligned} \tag{7}$$

This is a recursive formulation of the problem. The annotations indicate the individual terms' relevance.

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 various tutorials (e.g., [8]) and books (e.g., [10]) available on the subject.

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 measurements 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:

$$\begin{aligned}
& \{\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}\}^i \\
& \qquad \sim q\left(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i, y_k\right), \\
& \qquad\qquad i = 1, \dots, N.
\end{aligned} \tag{8}$$

A weight is then assigned according to the principle of importance sampling:

$$\begin{aligned}
\bar{w}_k^i = w_{k-1}^i & \frac{p\left(\{s_{1:t_k}, \tau_{1:t_k}\}^i \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i\right)}{q\left(\{s_{1:t_k}, \tau_{1:t_k}\}^i \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i, y_k\right)} \\
& \times p\left(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^i\right).
\end{aligned} \tag{9}$$

These *unnormalised* weights are then normalised:

$$w_k^i = \frac{\bar{w}_k^i}{\sum_{i'=1}^{N} \bar{w}_k^{i'}}, \tag{10}$$

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. This approximate effective sample can be calculated as follows:

$$N_{\text{eff}} \approx \frac{1}{\sum_{i=1}^{N} (\bar{w}_k^i)^2}. \tag{11}$$

It is also possible to calculate the incremental likelihood:

$$p(y_k | y_{1:k-1}) \approx \sum_{i=1}^{N} \bar{w}_k^i, \tag{12}$$

which can be used to calculate the likelihood of the entire data sequence, which will be useful in later sections:

$$p(y_{1:k}) = p(y_1) \prod_{k'=2}^{k} p(y_{k'} | y_{1:k'-1}), \tag{13}$$

where $p(y_1) \triangleq p(y_1 | y_{1:0})$, so can be calculated using (12).

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 [11, 12]. 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,[2] 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*. 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 [4]. 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 [13].

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.

Equation (13) means that, if each iteration of the tracker introduces errors, the classification errors necessarily accumulate. There is nothing that can be done about this. All that can be 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.

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 nonuniform 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 $q(\{s_{1:t_k},\tau_{1:t_k}\}\setminus\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\}\,|\,\{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\},y_k,S_j)$. 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.

The classes are indexed by c, particles by i, and the strata by j. The model used to calculate the weights is M and the stratum is S. So, the unnormalised 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

$$p(S_j|y_{1:k}) \propto p(y_{1:k}|S_j)p(S_j), \qquad (14)$$

where $p(S_j)$ is the (probably uniform) prior across the strata.

[2] 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.

This leads to the following recursion:

$$p(S_j|y_{1:k}) \propto p(y_k|y_{1:k-1},S_j)p(S_j|y_{1:k-1}), \tag{15}$$

where $p(y_k|y_{1:k-1},S_j)$ can be estimated using a minor modification of (12) as follows:

$$p(y_k|y_{1:k-1},S_j) \approx \sum_{i,c} \bar{w}_k^{(i,j,c)}. \tag{16}$$

Similarly, for the classes,

$$p(M_c|y_{1:k}) \propto p(y_k|y_{1:k-1},M_c)p(M_c|y_{1:k-1}), \tag{17}$$

where

$$\begin{aligned} p(M_c|y_{1:k}) &= \sum_j p(S_j,M_c|y_{1:k}) \\ &= \sum_j p(S_j|y_{1:k})p(M_c|S_j,y_{1:k}), \\ p(M_c|S_j,y_{1:k}) &\propto \sum_i \bar{w}_k^{(i,j,c)}. \end{aligned} \tag{18}$$

To implement this recursion, the weights of the classes are normalised such that they sum to unity over the particle in the strata:

$$w_k^{(c|i,j)} \triangleq \frac{\bar{w}_k^{(i,j,c)}}{\bar{w}_k^{(i,j)}}, \tag{19}$$

where $\bar{w}_k^{(i,j)}$ is the total unnormalised weight of the particle:

$$\bar{w}_k^{(i,j)} \triangleq \sum_c \bar{w}_k^{(i,j,c)}. \tag{20}$$

These weights are then normalised such that they sum to unity within each strata:

$$w_k^{(i|j)} \triangleq \frac{\bar{w}_k^{(i,j)}}{\bar{w}_k^{(j)}}, \tag{21}$$

where $\bar{w}_k^{(j)}$ is the total unnormalised weight of the stratum:

$$\bar{w}_k^{(j)} \triangleq \sum_i \bar{w}_k^{(i,j)}. \tag{22}$$

These weights are also normalised such that they sum to unity across the strata:

$$w_k^{(j)} \triangleq \frac{\bar{w}_k^{(j)}}{\sum_j \bar{w}_k^{(j)}}. \tag{23}$$

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

$$w_k^{(i,j,c)} \propto w_k^{(j)} w_k^{(i|j)} w_k^{(c|i,j)}. \tag{24}$$

```
For $j = 1 : N_M$
        Initialise: $w_0^{(j)} = 1/N_M$
        For $i = 1 : N_P$
                Initialise: $w_0^{(i|j)} = 1/N_P$
                Initialise: $x_0^{(i,j)} \sim p(x_0)$
                For $c = 1 : N_M$
                        Initialise: $w_0^{(c|i,j)} = 1/N_M$
                End For
        End For
End For
For $k = 1 : N_K$
        Implement recursion
End For
```

Algorithm 1

So, with N_P particles and N_M classes (and so N_M strata), running the algorithm over N_K steps can be summarised as follows in Algorithm 1. $p(x_0)$ is the initial prior on the state and Implement Recursion is conducted as in Algorithm 2 where V_j is the reciprocal of the sum of the squared weights, on the basis of which one can decide whether or not it is necessary to Resample. N_T is then the threshold on the approximate effective sample size which determines when to resample; $N_T \approx (1/2)N_P$ might be typical. Note that the resampling operation will result in replicants of a subset of some of the particles within the jth stratum, but that for each copy of the ith particle in the jth stratum, $w_k^{(c|i,j)}$ is left unmodified.

6. EXAMPLE

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. As stated in the introduction, this example is intended to provide a difficult to analyse, yet simple to understand, exemplar problem. The author does intend the reader to infer that the specific choice of models and parameters are well suited to any specific application.

The x_τ process is taken to be a constant velocity model; an integrated diffusion process

$$f(x_{\tau+\Delta}|x_\tau,s) = \mathcal{N}(x_{\tau+\Delta};A(\Delta)x_\tau,Q_s(\Delta)), \tag{25}$$

where $\mathcal{N}(x;m,C)$ denotes a Gaussian distribution for x, with mean, m, and covariance, C, and where

$$\begin{aligned} A(\Delta) &= \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix}, \\ Q_s(\Delta) &= \begin{bmatrix} \dfrac{\Delta^3}{3} & \dfrac{\Delta^2}{2} \\ \dfrac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma_s^2, \end{aligned} \tag{26}$$

```
Initialise $\bar{w}_k = 0$
For $j = 1 : N_M$
    Initialise $V_j = 0$
    Initialise output classification probabilities: $\bar{P}_k^c = 0$
    Initialise $\bar{w}_k^{(j)} = 0$
    For $i = 1 : N_P$
        Initialise $\bar{w}_k^{(i,j)} = 0$
        Sample $x_k^{(i,j)} \sim q(x_k | x_{k-1}^{(i,j)}, y_k, S_j)$
        For $c = 1 : N_M$
            $w_k^{(i,j,c)} = w_{k-1}^{(j)} w_{k-1}^{(i|j)} w_{k-1}^{(c|i,j)}$
            $\bar{w}_k^{(i,j,c)} = w_k^{(i,j,c)} (p(y_k | x_k^{(i,j)}, M_c)$
              $\times p(x_k^{(i,j,c)} | x_{k-1}^{(i,j)}, M_c) / q(x_k^{(i,j)} | x_{k-1}^{(i,j)}, y_k, S_j))$
            $\bar{w}_k^{(i,j)} = \bar{w}_k^{(i,j)} + \bar{w}_k^{(i,j,c)}$
            $\bar{w}_k^{(j)} = \bar{w}_k^{(j)} + \bar{w}_k^{(i,j,c)}$
            $\bar{w}_k = \bar{w}_k + \bar{w}_k^{(i,j,c)}$
            $\bar{P}_k^c = \bar{P}_k^c + \bar{w}_k^{(i,j,c)}$
        End For
    End For
End For
For $c = 1 : N_M$
    $P_k^c = \bar{P}_k^c / \bar{w}_k$, which can be output as necessary
For $j = 1 : N_M$
    $w_k^{(j)} = \bar{w}_k^{(j)} / \bar{w}_k$
    For $i = 1 : N_P$
        $w_k^{(i|j)} = \bar{w}_k^{(i,j)} / \bar{w}_k^{(j)}$
        For $c = 1 : N_M$
            $w_k^{(c|i,j)} = \bar{w}_k^{(c|i,j)} / \bar{w}_k^{(i,j)}$
        End For
        $V_j = V_j + (w_k^{(i|j)})^2$
    End For
    Resample $j$th stratum if $1/V_j < N_T$
End For
```

Algorithm 2

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 data are linear Gaussian measurements of position

$$p(y_k | x_{\tau_k}) = \mathcal{N}(y_k; H x_{\tau_k}, R), \tag{27}$$

where

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix}, \tag{28}$$

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

$$g(\tau - \tau_t | s_t, M_c) = \begin{cases} \mathcal{G}(\tau - \tau_t; 2, 5), & s_t = 1,\ c = 1, \\ \mathcal{G}(\tau - \tau_t; 10, 1), & s_t = 1,\ c = 2, \\ \mathcal{G}(\tau - \tau_t; 50, 0.2), & s_t = 1,\ c = 3, \\ \mathcal{G}(\tau - \tau_t; 10, 0.1), & s_t = 2,\ \forall c, \end{cases} \tag{29}$$

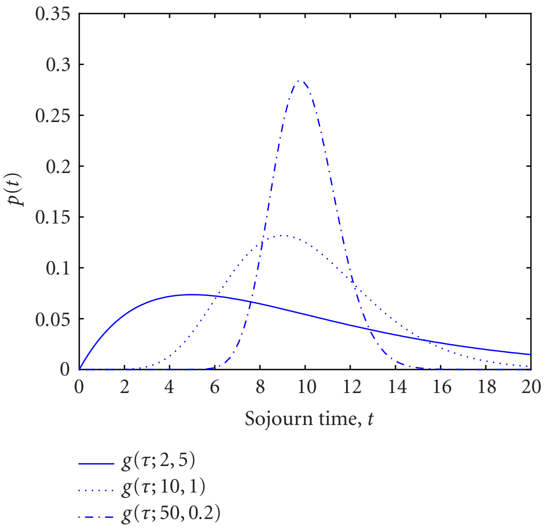


Figure 2: Sojourn time distributions for $s_t = 1$ for the different classes.

where $\mathcal{G}(x; \alpha, \beta)$ is a gamma distribution over x, with shape parameter α and scale parameter β. Figure 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 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:

$$p(s_t | s_{t-1}) = \begin{cases} 0, & s_t = s_{t-1}, \\ 1, & s_t \neq s_{t-1}. \end{cases} \tag{30}$$

This means that, given the initial discrete state, the sojourn ends define the discrete-state sequence.

$p(s_1)$ is taken to be uniform across the two models and $p(\tau_1) = \delta(\tau_1 - 0)$, so it assumed known that there was a transition at time 0. x_0 is initialised at zero as follows:

$$x_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{31}$$

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, (6), the dynamic prior for the s_t process is used, with a minor modification:

$$\begin{aligned} & q\left(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \mid \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, y_k\right) \\ & \triangleq p\left(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \mid s_{1:t_{k-1}}, \tau_{1:t_{k-1}}, \tau_{t_k+1} > \tau_k, M_j\right) \\ & = \int p\left(\{s_{1:t_k}, \tau_{1:t_k+1}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}\right. \\ & \qquad \left. \mid s_{1:t_{k-1}}, \tau_{1:t_{k-1}}, \tau_{t_k+1} > \tau_k, M_j\right) d\tau_{t_k+1}, \end{aligned} \tag{32}$$

that is, 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:

$$\bar{w}_k^i = w_{k-1}^i p\left(y_k \mid \{s_{1:t_k}, \tau_{1:t_k}\}^i\right), \tag{33}$$

where $p(y_k \mid \{s_{1:t_k}, \tau_{1:t_k}\}^i)$ 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 k 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, $w_k^{(i)}$.

Just $N_P = 25$ particles are used 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, m. The initial value for the covariance is then defined as C:

$$C = \begin{bmatrix} 100 & 0 \\ 0 & 10 \end{bmatrix}. \tag{34}$$

The weights are all initialised as equal for all the particles. Resampling takes place if the approximate effective sample size given in (11) falls below $N_T = 12.5$. Since each particle needs to calculate the parameters of a Kalman filter, the computational cost is roughly equivalent to that of a multiple hypothesis tracker [14] with 25 hypotheses; here the hypotheses (particles) are in the continuous space of the times of the sojourn ends rather than the discrete space of the associations of measurements with the track. The computational cost is therefore relatively low and the algorithm is therefore amenable to practical real-time implementation.

With N_P particles and N_K iterations, the algorithm is implemented as in Algorithm 3.

```
For $i = 1 : N_P$
        Initialise $w_0^i = 1/N_P$
        Initialise $\tau_1^i = 0$
        Initialise $s_1^i$ as 1 if $i > N_P/N_M$ or 2 with otherwise
        Initialise Kalman filter mean $m_0^i = m$
        Initialise Kalman filter covariance $C_0^i = C$

End For
For $k = 1 : N_K$
        Initialise $V = 0$
        Initialise $\bar{w}_k = 0$
        For $i = 1 : N_P$
                Sample $\{s_{1:t_k}, \tau_{1:t_k}\}^i \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i$
                  $\sim p(\{s_{1:t_k},\tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\} \mid \{s_{1:t_{k-1}},\tau_{1:t_{k-1}}\}^i)$
                Calculate $m_k^i$ and $C_k^i$ from $m_{k-1}^i$ and $C_{k-1}^i$
                  using $s_{1:t_k}^i \setminus s_{1:t_{k-1}}^i$
                Calculate $p(y_k \mid \{s_{1:t_k}, \tau_{1:t_k}\}^i)$ from $y_k$, $m_k^i$,
                  and $C_k^i$
                $\bar{w}_k^i = w_{k-1}^i p(y_k \mid \{s_{1:t_k}, \tau_{1:t_k}\}^i)$
                $\bar{w}_k = \bar{w}_k + \bar{w}_k^i$
        End For
        For $i = 1 : N_P$
                $w_k^i = \bar{w}_k^i/\bar{w}_k$
                $V = V + (w_k^i)^2$
        Resample if $1/V < N_T$

End For
```

Algorithm 3

The true trajectory through the discrete space is given in Figure 3. The hypothesis for the trajectory through the discrete space for some of the particles is shown in Figure 4. Note that, as a result of the resampling, all the particles 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.

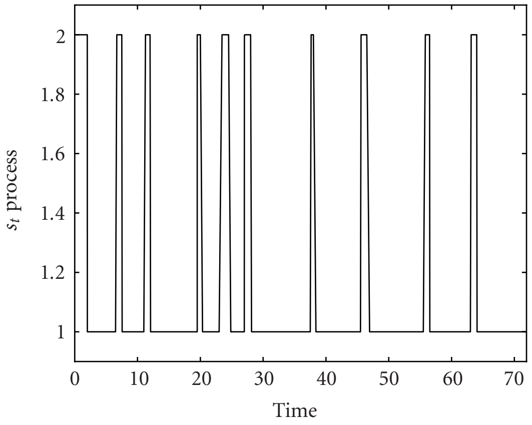


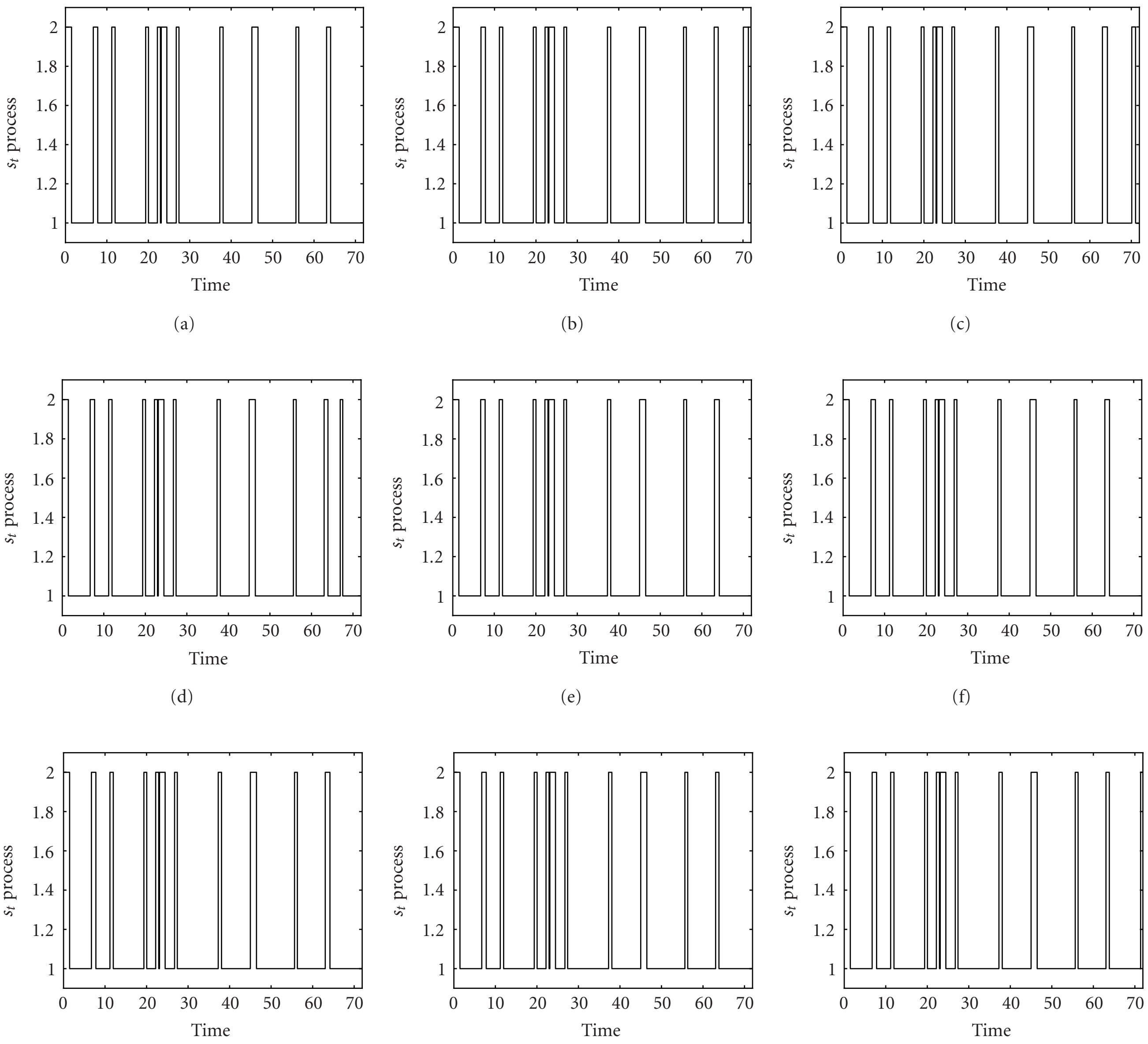
Figure 3: True trajectory for target through s_t state space.

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:

$$\begin{aligned} & q\left(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \mid \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, y_k, S_j\right) \\ & \qquad \triangleq p\left(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \mid \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, M_j\right). \end{aligned} \tag{35}$$

The weight update equation is then



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 4: A subset of the particles' hypothesised trajectories through s_t 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.

$$\bar{w}_k^{(i,j,c)} = w_{k-1}^{(i,j,c)} \frac{p\left(y_k \middle| \left\{s_{1:t_k}, \tau_{1:t_k}\right\}^{(i,j)}\right) p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\}^{(i,j)} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^{(i,j)} \middle| \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^{(i,j)}, M_c\right)}{p\left(\left\{s_{1:t_k}, \tau_{1:t_k}\right\}^{(i,j)} \setminus \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^{(i,j)} \middle| \left\{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\right\}^{(i,j)}, M_j\right)}. \quad (36)$$

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, $w_k^{(i|j)}$.

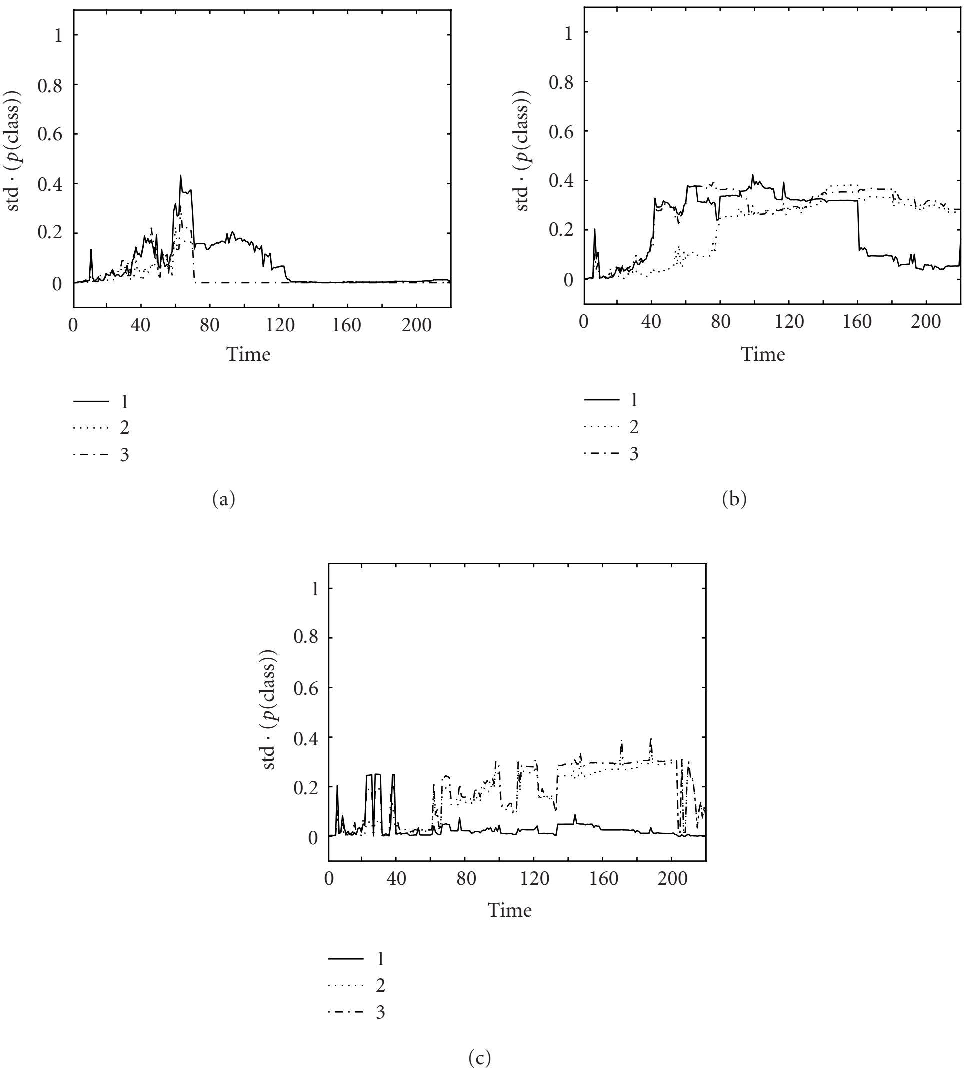


Figure 5: Standard deviation (std) of estimated classification probabilities, (p (class)), across ten filter runs for simulations according to each of the three models, labelled as 1, 2, and 3. (a) Data simulated from class 1. (b) Data simulated from class 2. (c) Data simulated from class 3.

Each stratum also stores $w_k^{(j)}$. The reader is referred to the preceding sections' summaries of the algorithms for the implementation details.

$N_P = 25$ particles are used per stratum, each is initialised as described 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 (11) for the stratum falls below $N_T = 12.5$. Since each of the $N_M = 3$ strata has $N_P = 25$ particles, the computational cost is approximately that of a multiple hypothesis tracker which maintains 75 hypotheses; the algorithm is practicable in terms of its computational expense.

However, it should be noted that, for this difficult problem of joint tracking and classification using very similar models, the number of particles used is small. This is intentional and is motivated by the need to look at the difference between the variance in the class membership probabilities and the variance of the strata weights.

Ten runs were conducted with data simulated according to each of the three models. The number of particles used is deliberately sufficiently small that the inevitable accumulation of errors causes problems in the time frame considered. This enables a comparison between the time variation in the variance across the runs of the classification probabilities and the variance across the runs of the strata weights. So, Figures 5 and 6 show the time variation in the variance across the runs of these two quantities. It is indeed evident that there is significant variation across the runs; the errors are indeed accumulating with time. It is also evident that this accumulation is faster for the importance weights than for the classification probabilities. This implies that the choice of importance function is less important, in terms of robustness of the estimation of the classification probabilities, than

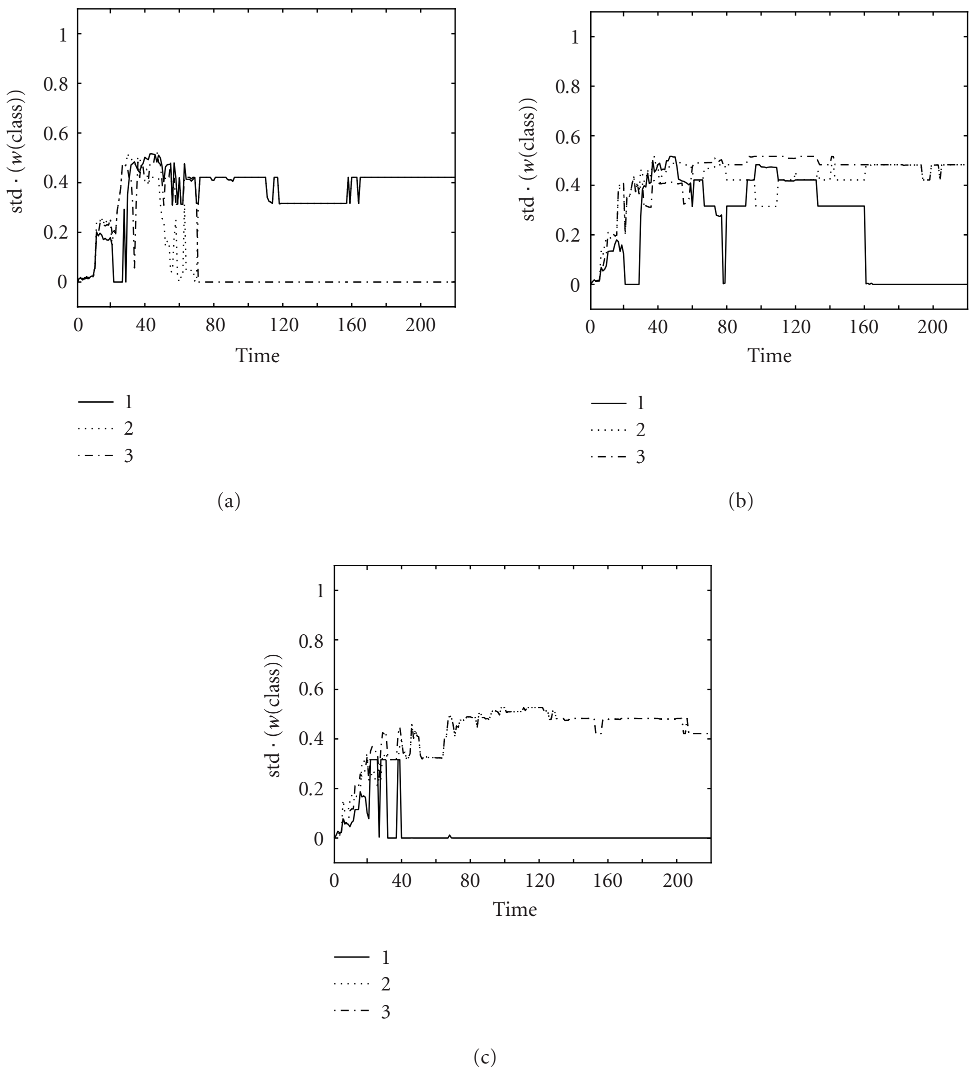


FIGURE 6: Variance of strata weights across ten filter runs for simulations according to each of the three models. (a) Data simulated from class 1. (b) Data simulated from class 2. (c) Data simulated from class 3.

calculating the probabilities of all the classes for every sample.

It is difficult to draw many conclusions from the variations across the true class. Since such issues are quite specific to the models and parameters, which are not the focus of this paper, this is not further investigated or discussed.

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 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.

ACKNOWLEDGMENTS

This work was funded by the UK MoD Corporate Research Programme. The author also gratefully acknowledges the award of his Industrial Fellowship by the Royal Commission for the Exhibition of 1851. The author would like to thank John Boyd and Dave Sworder for useful discussions on the subject of joint tracking and classification associated with the use of semi-Markov models and Arnaud Doucet and Matthew Orton for discussions of how particle filters can be used in such problems. The author also thanks the reviewers for their comments.

REFERENCES

[1] D. R. Cox and V. Isham, *Point Processes*, Chapman and Hall, New York, NY, USA, 1980.

[2] L. Campo, P. Mookerjee, and Y. Bar-Shalom, "State estimation for systems with sojourn-time-dependent Markov model

switching," *IEEE Trans. Automatic Control*, vol. 36, no. 2, pp. 238–243, 1991.

[3] D. D. Sworder, M. Kent, R. Vojak, and R. G. Hutchins, "Renewal models for maneuvering targets," *IEEE Trans. on Aerospace and Electronics Systems*, vol. 31, no. 1, pp. 138–150, 1995.

[4] N. J. Gordon, S. Maskell, and T. Kirubarajan, "Efficient particle filters for joint tracking and classification," in *Signal and Data Processing of Small Targets*, vol. 4728 of *Proceedings SPIE*, pp. 439–449, Orlando, Fla, USA, April 2002.

[5] S. Challa and G. Pulford, "Joint target tracking and classification using radar and ESM sensors," *IEEE Trans. on Aerospace and Electronics Systems*, vol. 37, no. 3, pp. 1039–1055, 2001.

[6] Y. Boers and H. Driessen, "Integrated tracking and classification: an application of hybrid state estimation," in *Signal and Data Processing of Small Targets*, vol. 4473 of *Proceedings SPIE*, pp. 198–209, San Diego, Calif, USA, July 2001.

[7] S. Herman and P. Moulin, "A particle filtering approach to FM-band passive radar tracking and automatic target recognition," in *Proc. IEEE Aerospace Conference*, vol. 4, pp. 1789–1808, BigSky, Mont, USA, March 2002.

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

[9] J. Vermaak, S. J. Godsill, and A. Doucet, "Radial basis function regression using trans-dimensional sequential Monte Carlo," in *Proc. 12th IEEE Workshop on Statistical Signal Processing*, pp. 525–528, St. Louis, Mo, USA, October 2003.

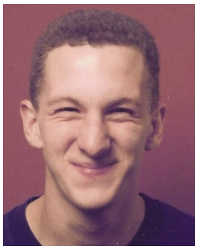
[10] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.

[11] F. Le Gland and N. Oudjane, "Stability and uniform approximation of nonlinear filters using the Hilbert metric, and application to particle filters," Tech. Rep. RR-4215, INRIA, Chesnay Cedex France, 2001.

[12] D. Crisan and A. Doucet, "Convergence of sequential Monte Carlo methods," Tech. Rep. CUED/F-INFENG/TR381, Department of Engineering, University of Cambridge, Cambridge, UK, 2000.

[13] P. M. Baggenstoss, "The PDF projection theorem and the class-specific method," *IEEE Trans. Signal Processing*, vol. 51, no. 3, pp. 672–685, 2003.

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



Simon Maskell is a Ph.D. degree graduand and holds a first-class degree with Distinction from Department of Engineering, University of Cambridge. His Ph.D. degree was funded by one of six prestigious Royal Commission for the Exhibition of 1851 Industrial Fellowships awarded on the basis of outstanding excellence to researchers working in British industry. At QinetiQ, he is a Lead Researcher for tracking in the Advanced Signal and Information Processing Group (ASIP). As such, he leads several projects and coordinates ASIP's tracking research while also supervising a number of other researchers. Simon has authored a number of papers, as well as several technical reports and a patent. He has also been the leading force behind the development of a QinetiQ product to provide a generic solution to all of QinetiQ's tracking problems.