Fixed Lag Particle Filtering for Target Tracking

Pete Bunch

August 2011

A report on the first year's progress

UNIVERSITY OF
CAMBRIDGE

Contents

1

Introduction

In a target tracking problem, the aim is to trace the trajectory of an object over time from a set of discrete observations. At first glance such a problem may appear no different from a standard case of state estimation to be solved with a Kalman filter (if linear) or particle filter (if not). The additional difficulty in target tracking is the nature of the observation process. Our imperfect sensors may not detect every target present in the scene in every scan. Furthermore, there may be some number of false alarms arising from sensor errors or clutter. Our task thus becomes three-fold: firstly to detect what targets are present in the scene, secondly to work out which observations were generated by each target, and finally to estimate the states of the targets.

Research in target tracking emerged from military applications such as radar and sonar. However, similar problems emerge in many diverse branches of science: the tracking of people, vehicles or animals in video sequences; of molecules or cells in microscopy data; or of notes in a piece of music. Each can be reduced to a similar underlying model.

In this work, a number of new algorithms have been developed for target tracking. These algorithms employ fixed-lag particle filters, which allow previous states to be re-proposed when later observations have been made. Examples have been implemented using both sequential importance sampling with resampling and Markov chain Monte Carlo.

This report is structured as follows:

In chapter 2 we review the relevant literature on tracking and particle filtering for nonlinear estimation.

In chapter 3 we review some mathematical basics which underlie the inference methods used

later and set out basic notation.

In chapter 4 we set out the tracking models used and investigate factorisation methods for efficient particle filter implementations.

In chapter 5 we introduce a framework for fixed-lag particle filtering, including the required proposal distributions.

In chapter 6 we present two implementations of fixed-lag particle filters for tracking, one using sequential importance sampling with resampling, the other using Markov chain Monte Carlo.

In chapter 7 we outline additions to the models and algorithms to handle joint detection and tracking of multiple targets.

In chapter 8 we present results and performance evaluations of the trackers.

In chapter 9 we examine a separate project on inference of intended destinations from mouse cursor data.

In chapter 10 a number of ideas for potential future research are set out.

2

Literature Review

Two spheres of literature will be of interest here: the developments of models and algorithms for tracking, and the studies of the numerical inference schemes which underly our method. We begin with the latter.

2.1 Particle filters

Filtering is the operation of infering the state of an evolving latent random process given a set of imperfect observations up until the current time. In a Bayesian estimation scheme, we would like to calculate a posterior probability distribution for such a state at each point in time. The difficulty with such a procedure is that the complexity of the state distribution will tend to compound over time. In the particular case where both the state transition and observation processes are linear transformations with Gaussian noise, it is possible to derive an analytic expression for the state distribution with constant complexity, the fabulous Kalman filter (KF) of (Kalman 1960). However, as many problems are ofttern nonlinear or non-Gaussian, and as no other such analytic and generally applicable cases have been discovered, (Daum 2005), we must resort to numerical techniques.

The particle filter (PF) approximates a probability distribution using a set of discrete samples or *particles* drawn from it. This will be no trivial task - the distribution cannot be sampled directly, just as it cannot be expressed analytically. Instead, at each time step, the cloud of particles representing the previous distribution is propagated forwards to approximate the next, taking into account the latest observation. The most common PF algorithm in use is known as Sequential Importance Sampling with Resampling (SISR) (or some permutation of

this initialism) - in fact the terms SISR and PF are often used interchangeably. Throughout this report we discriminate the term PF (any filter using particles) from SISR (a particular implementation) and we avoid the term Sequential Monte Carlo (SMC) which is often used as synonymous with either.

2.1.1 Sequential importance sampling

Importance sampling

The first modern implementation of the SISR algorithm by (Gordon, Salmond & Smith 1993). At the heart of the algorithm is an importance sampling step. Each particle from the old distribution is propagated forward by sampling from a proposal or 'importance' distribution. The particles are then assigned an "importance weight" to account for the discrepancy between the proposal distribution and the desired posterior. A significant strength of the algorithm is that we need only be able to calculate the posterior probability up to a normalising constant. This may be easily accounted for by normalising the weights of the discrete sample set.

Resampling

Over time, the variance in these weights is liable to grow, as unlikely particles deviate completely from the correct state and are assigned a negligible weight. The eventual result is a degenerate sample with all the weight on one or a few particles and most contributing nothing to the approximation. The solution to this degeneracy problem is to introduce a resampling step before the importance sampling, in which particles are sampled from the particle approximation. Thus, heavily weighted particles are multiplied while low weight particles are discarded. Many algorithms for resampling exist, a summary of which is given by (Doucet & Johansen 2009). This resampling step introduces a degree of Monte Carlo error, and so (Liu & Chen 1995) introduce a measure of particle diversity to assess the degree of degeneracy. This measure, the effective sample size (ESS), is an estimate of the equivalent number of equal-weighted particles in the approximation, and is based on a calculation of the sample weight variance. Resampling is only used if the ESS falls below a chosen threshold.

Importance distributions

The filter proposed by (Gordon et al. 1993), known as the bootstrap filter, used the state transition density as the importance distribution, as did many other early implementations, (Blake, Isard et al. 1998, Kitagawa 1996). However, this can result in poor performance if the

observation process is informative. In other words, if we know that the observation is always very close to the state, then there is no point in proposing particles far away which will have a very low weight. It was suggested in, amongst others, (Liu & Chen 1995) that a better proposal could be constructed by considering the position of the new observation. In particular, the "optimal" (in the sense that it minimises the weight variance) proposal is the conditional distribution of the new state given the state history and the new observation (see (Doucet, Godsill & Andrieu 2000) for proof). This is rarely available analytically or samplable, so Gaussian approximations may be used (Doucet et al. 2000).

Auxiliary particle filtering

In the description above, resampling was introduced as an additional procedure to rejeuvenate a degenerate particle distribution. However, it is possible to view it as a more integral part of the sampling procedure: At each step we construct a set of particles by proposing both a new state and a history for each. These histories are selected from the previous particle distribution. The resamping fulfils the role of a "history proposal". On a step with resampling we propose histories from the weighted particle distribution from the previous frame. On a step without resampling, we propose histories from the unweighted particle distribution, which can of course be done by simply keeping the same set of particles! In this new paradigm the obvious generalisation is to allow history proposals from the previous set of particles with arbitrary weights. A correction is made in the importance weight calculation to account for the proposal weights. This is the innovation of the auxliary particle filter introduced by (Pitt & Shephard 1999). The proposal weights may be assigned so as to favour particles which give rise to more likely states.

Resample-move

The objective when selecting the "optimal" proposal distribution or in choosing auxiliary proposal weights is to minimise the variance of the importance weights, and thus reduce the need for resampling. Another important particle filtering technique is the resample-move method of (Gilks & Berzuini 2001), which aims instead to replenish the particle diversity after resampling. Rather than beginning the next IS step with multiple copies of the same particle, the current and previous states of each are adapted using one or more Metropolis-Hastings moves. Although this method improves the representation of the state distribution, it requires significant additional computation for each particle.

2.1.2 Markov chain Monte Carlo

Metropolis-Hastings

Markov chain Monte Carlo (MCMC) algorithms, like SISR, make use of a set of samples to approximate a probability distribution. Unlike SISR, MCMC is not specifically designed to address problems of sequential inference, although, as we shall see, they may be applied to these problems as an MCMC particle filter.

The basic MCMC method was proposed by (Metropolis, Rosenbluth, Rosenbluth, Teller, Teller et al. 1953) and extended by (Hastings 1970), after which two the Metropolis-Hastings (MH) algorithm is named. Rather than producing independent samples from a distribution, MH uses a Markov chain. A reversible, ergodic Markov chain has a stationary distribution. By careful design of the transition properties of the chain, we can make the stationary distribution equal to some desired target distribution. For our inference problems, this target will be the state posterior.

A MH move is made by sampling a value of the state from a proposal distribution. With some probability, the new value is accepted, otherwise the previous value is maintained. The acceptance probability is given by:

$$\alpha = \min\left(1, \frac{P(X_{\text{new}})q(X_{\text{old}}|X_{\text{new}})}{P(X_{\text{old}})q(X_{\text{new}}|X_{\text{old}})}\right) \tag{2.1}$$

where $q(.|.)$ is the proposal distribution, and $P(.)$ the target.

The chain must be initialised at some value, and this is unlikely to be a sample from the target (if we could sample it, we would not need MCMC). Thus, there will be a period over which the sampler converges to the target distribution. This period is known as the "burn in" and the samples should be excluded from the particle approximation.

The choice of proposal does not affect the target distribution, although it will affect the rate at which the algorithm converges. If the proposal distribution has a low variance then successive samples will be close together, and it may take many moves to explore the whole distribution. If the variance is large, then the acceptance probability may be very low and again convergence may be slow.

A complete and excellent discussion of MCMC design considerations, convergence properties, etc. can be found in (Gilks, Richardson & Spiegelhalter 1996), and will not be repeated here.

Block sampling

When MCMC algorithms are applied to multivariate distributions it is not necessary to change every variable in every move. If the state is divided into two blocks, X_i and X_{-i}, and a move is proposed only on X_i, then the acceptance probability is now given by:

$$\alpha = \min\left(1, \frac{P(X_{i,\text{new}}|X_{-i})q(X_{i,\text{old}}|X_{i,\text{new}}, X_{-i})}{P(X_{i,\text{old}}|X_{-i})q(X_{i,\text{new}}|X_{i,\text{old}}, X_{-i})}\right) \tag{2.2}$$

A special case of this scheme occurs when $q(X_{i,\text{new}}|X_{i,\text{old}}, X_{-i}) = P(X_{i,\text{new}}|X_{-i})$. This is the Gibbs sampler devised by (Geman & Geman 1984). With this choice of proposal distribution, the acceptance probability cancels out to 1. Thus, all samples are accepted. The difficulty with this formulation is that the required conditional distribution may not be available, or may not be samplable.

MH moves which alter only a subset of the variables are commonly known as Metropolis-within-Gibbs (MwG) moves.

Reversible jump Markov chain Monte Carlo

MCMC methods are not restricted to distributions of fixed dimension. The Reversible Jump MCMC (RJMCMC) method devised by (Green 1995) allows MH moves to jump between state spaces with different dimensionality. This is especially useful when trying to choose between models while simultaneously estimating parameters, or for estimating the order or number of components in a model, for example in a Gaussian mixture or autoregressive model.

RJMCMC moves may have very low acceptance probabilities. When the sampler jump from one state space to another, there may be no obvious function for generating likely values in the new state. For such cases, (Al-Awadhi, Hurn & Jennison 2004) propose using a system of compound moves, where a model change move is followed by a number of normal MH moves designed to seek out a likely set of state values. A single acceptance probability is then calculated for the set of moves.

Sequential Markov chain Monte Carlo

The interest in MCMC for this project lies is in their application to sequential inference problems. Such 'MCMC particle filters' were first proposed by (Khan, Balch & Dellaert 2005), although this first implementation suffered from computation issues, and (Golightly & Wilkinson 2006), and have recently been applied to a range of problems including target tracking, (Septier, Pang, Carmi & Godsill 2009), group tracking, e.g. (Carmi, Godsill & Septier 2009), tracking of a

correlated stock prices, (Pang, Godsill, Li & Septier 2011) and music transcription, (Bunch & Godsill 2010).

The MCMC particle filter runs a Markov chain for each frame which targets the posterior state distribution at that time. The particles are thus unweighted and are generated in series rather than in parallel, with each particle depending on the last. The state space of the Markov chain consists of the entire history of the hidden variables as well as the latest values. Two types of MH move are thus required: History-moves involve proposing a new history from the particles of the previous particle distribution. Current-moves involve a proposal of the latest variables given the history and data. The two types of move are analogous to the resampling and importance sampling steps of the SISR algorithm respectively.

MCMC particle filters have a significant advantage over their SISR counterparts for complex, high dimensional problems. As noted in (Pang, Li & Godsill 2008), we can use MwG moves, changing only a few variables at once, to explore the posterior state distribution. In contrast, with an SISR particle filter, we must propose a new value for every variable at once. In some cases this enables the MCMC particle filter to give a better approximation of the state posterior, as demonstrated in (Pang et al. 2011).

2.1.3 *Particle smoothing*

Smoothing is the process of making a state estimate after some delay, using data from further ahead in time. Just as Gaussian models lead to the KF for filtering, so a number of Kalman-like smoother algorithms can also be derived, including the Rauch-Tung-Striebel (RTS) smoother of (Rauch, Tung & Striebel 1965). As noted in (Kitagawa 1996), an ordinary SISR particle filter generates a distribution over the entire state history at each step, thus providing smoothing estimates. However, because of the resampling procedure, these estimates will be increasingly degenerate as the lag increases. Improved strategies are proposed in (Clapp & Godsill 1999, Doucet et al. 2000, Godsill, Doucet & West 2004) in which the particles of each filtering distribution are reweighted based on the future results. These methods are limited by the fact that the support of the smoothing distributions are restricted to that of the filtering distribution. Furthermore, only the last of these provides an algorithm with linear complexity in the number of particles.

In contrast, (Pitt & Shephard 2001) propose a fixed lag smoother in which new states are sampled over the whole window in each processing frame. This idea is developed further by (Doucet, Briers & Sénécal 2006), in which a general framework for fixed-lag state estimation is set out, based on the principles of the SMC Sampler (Del Moral, Doucet & Jasra 2006). It is this

framework which is exploited in our work for target tracking. Along similar lines, the "practical filter" of (Polson, Stroud & Müller 2008) is an MCMC-based smoother which proposes states over a fixed-lag window. This is limited by an approximation for the state at the start of the window. A single particle is used to represent this state, which may result in poor performance in the case of multi-modality.

2.1.4 Coping with dimensionality

It is a commonly noted problem that the efficacy of particle filters falls as the number of state dimensions increases. This is a problem for both SISR and MCMC filters, and indeed for any Monte Carlo approximation. There is a strong intuition for this phenomenon - it will clearly take more particles to adequately sample a square than it would a line, and more still for a cube. An analysis of this effect, including numerical studies, was conducted in (Daum & Huang 2003).

The severity of the "curse of dimensionality" will depend on the correlation between the state variables, and the ability to exploit such correlations by the design of effective proposal distributions. Methods to improve performance of MCMC algorithms include Hybrid Monte Carlo (HMC) (Duane, Kennedy, Pendleton & Roweth 1987), which introduces "momentum" variables to the state in order to build better proposals and the Riemann manifold methods of (Girolami & Calderhead 2011).

2.1.5 Marginalised particle filters

Monte Carlo methods are effective for tackling nonlinear, non-Gaussian problems when analytic methods cannot be found. However, they are computationally expensive and so should be avoided when an analytic solution does exist. In some problems, the state can be partitioned into two parts, one which behaves in a linear-Gaussian way conditional on the other. In such a case a particle filter can be used for inference of the nonlinear part, and a Kalman filter for the linear-Gaussian part. Such a strategy reduces the variance of the state estimates, in accordance with the Rao-Blackwell theorem. See, for example, (Casella & Robert 1996). The only change required is that likelihood calculation for the particle filter will now require the linear-Gaussian part of the state to be marginalised. Such schemes may be used with both SISR and MCMC particle filters and are known as *marginalised* or *Rao-Blackwellised* particle filters.

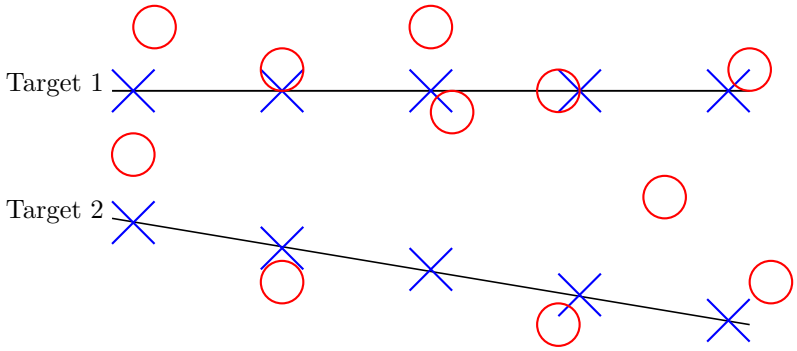


Figure 2.1: A simple two-target tracking problem. Hidden states shown as blue crosses. Observations shown as red circles.

2.2 Tracking

In a target tracking situation we aim is to trace the trajectory of an object over time from a set of discrete observations. For the simplest case, with a single target under observation, which is detected in every frame, and with no false alarms, this is a simple state-space inference problem. It can be approached with a Kalman filter or a basic particle filter. The phenomenon which makes tracking problems more challenging is association ambiguity. In a given frame, we may not be guaranteed to detect a target, and we may pick up false alarms or clutter measurements. In general there is no way to establish with certainty which observation arose from which target. Finally, the number of targets in the scene may also be unknown. We are thus faced with a three-fold problem:

- Detect how many targets are present
- Estimate which observation arose from each target
- Estimate the state of each target

Figure 2.1 shows a illustrates the problem. The requirement is to estimate the hidden states (blue crosses) from the observations (red circles).

In this section, we review a number of strategies for tackling such a target tracking problem. Note that although we divide up the algorithms into sections, there is significant overlap between them, which we endeavour to highlight.

2.2.1 Probabilistic data association

The earliest works on target tracking with data association used a combination of Kalman filters and heuristics. (Sea 1971) suggests using only the observation with the minimum Mahalanobis distance from the Kalman prediction, i.e. a maximum likelihood estimate of the association. Such nearest neighbour algorithms can easily be caused to lose track by a single unfortunate false alarm.

A probabilistic approach was introduced by (Bar-Shalom & Tse 1975), in the form of the probabilistic data association filter (PDAF). Rather than select a single observation for each target in each frame, for the PDAF a posterior probability of each possible association is calculated, given the previous target states. A Kalman update is then evaluated for each of the possible associations and the results added together, weighted by the corresponding association posterior. Thus, the final update takes into account multiple (indeed, all) observations. Thus the PDAF cannot be so easily upset by single clutter measurement, although additional error is introduced in situations where a nearest neighbour method would have picked the correct association, because of the influence of false alarms which now contribute to the estimate.

When more than one target is present in the scene, a separate PDAF can be run for each one. However, this can lead to errors. When two tracks pass close together, they may both have a high association probability with the same observation(s). The result is that the estimates converge onto the same path, and one of the tracks is lost.

The shortcomings of the PDAF are addressed by extending the filter to consider the joint association posteriors. This is the joint PDAF (JPDAF) of (Fortmann, Bar-Shalom & Scheffe 1983), excellently reviewed in (Bar-Shalom, Daum & Huang 2009). Rather than calculating association probabilities for each target in isolation, the joint association hypotheses are assessed. We can now include as prior information the fact that no observation can be associated with two targets. Thus the problem of tracks following the same observations is reduced. The price of this improved performance is the need to calculate an association probability for every joint hypothesis. The number of these is of combinatorial complexity in the number of targets and observations in a frame. Gating is used to address this issue: a target-observation association is only considered possible if the Mahalanobis distance between the two is below some threshold. The choice of threshold governs the probability of excluding the correct hypothesis, and thus trades off performance and complexity (Sea 1971).

Both the PDAF and the JPDAF suffer from a problem of bias and track coalescence when targets pass close to each other. This was observed and quantified by (Fitzgerald 1985). The measurements generated by the second target ‘pull’ the estimate of the first away from its correct

position. When two targets are travelling along parallel paths this can lead to "track coalescence". The estimates of both target states move together, midway between the two correct locations. Solutions to this problem often resort to reintroducing nearest-neighbour methods, such as the nearest-neighbour-JPDAF (Fitzgerald 1986) or the "set-JPDAF" (Svensson, Svensson & Willett 2009).

So far we have outlined how the JPDAF approaches the problems of data association and state estimation. It remains to consider how target detection may be incorporated. One of the first methods proposed was that of (Bar-Shalom, Chang & Blom 1989) which used the interacting multiple model (IMM) algorithm in parallel with the JPDAF. When a the possibility of a new target existed, two filters were run in parallel, one assuming the existence of the new target, and one assuming its non-existence. Decision logic was included to choose between the two possibilities. A less computationally demanding approach was formulated by (Musicki, Evans & Stankovic 1994, Musicki & Evans 2004), in which each target was labeled with a probability of exitence, allowing potential new targets to be assessed in parallel with tracking.

The JPDAF requires the dynamics of the targets and the observation process to be linear and Gaussian, or that an appropriate EKF-like linear approximation can be made. However, even given this requirement, the estimation process still involves a sub-optimal step. The final posterior state distribution is constructed by adding a weighted sum of distributions, each made with a different association hypothesis. The individual components are Gaussians, and so the posterior should be a sum of Gaussian. However, in the JPDAF, this is collapsed into a single Gaussian with a matched mean and variance. This collapsing step may throw away important information if there is more than one significant component, and is responsible for the track coallescence effect noted before. It is possible to construct a filter which maintains the complete or a partially complete Gaussian sum posterior, (Singer, Sea & Housewright 1974, Salmond 1990), at the expense of increased complexity. In fact, this produces an algorithm broadly equivalent to the multi-hypothesis tracker (Blackman 2004)!

There is an alternative to using a Kalman filter for the JPDAF, with its associated approximations. We can use a particle filter instead for the state estimation. Such an algorithm is developed by (Schulz, Burgard, Fox & Cremers 2001, Karlsson & Gustafsson 2001, Vermaak, Godsill & Perez 2005). A set of particles representing targets states are propagated forwards using IS and used to calculate the association priors using a Monte Carlo estimate. These association priors are then used in the particle weight calculations. The particles approximate the target posterior state distribution. Using a particle approximation allows us to trade off accuracy against computation by varying the number of particles, which may be an improvement

on Gaussian approximations for nonlinear models.

A final note about the JPDAF. The algorithm assumes that the only interaction between targets is through the associations - no two targets can be associated with the same observation. Thus, once the association probabilities have been calculated, each target may be updated independently using a Kalman filter to give a marginal state distribution. Interactions between targets are difficult to accomodate because we do not work in the joint state space of all targets.

2.2.2 Data association hypothesis methods

With a JPDAF, the probability of each feasible combination of associations is calculated, and used to weight a sum of the state estimates made with the respective combinations. The Multi-Hypothesis Tracker (MHT) of (Reid 1979) instead maintains each of these estimates separately, with an associated probabilistic score. The MHT requires the target dynamics to be exactly or approximately linear-Gaussian, so that KFs can be used for state updates. As we shall see, a PF version results in Monte Carlo Data Association.

For each frame of data, each possible combination of associations between objects and observations is formed. State estimates are calculated with KF updates and the hypothesis is scored with a posterior probability. Detection of new tracks is readily incorporated into MHT. New tracks can be added into hypotheses wherever an un-associated observation exists. MHT has a potentially enormous computational complexity which grows combinatorially as time proceeds. To render it practical, observations are gated, disallowing associations between targets and distant measurements, and 'pruning' is used to eliminate hypotheses with low probabilities. (Blackman 2004) contains an excellent introduction to MHT and details of its implementation.

MHT requires significantly more computational complexity than the JPDAF. In particular, high levels of clutter can lead to prohibitively large numbers of hypotheses. Furthermore, it is reliant on the use of Gaussian approximations so that KF methods can be used to obtain state estimates.

A modification of MHT is derived by relaxing the constraint that any observation can only be associated with one target. This allows the tracking of each target to be conducted independently, with a great saving in computation (Complete lists of hypotheses are no longer required). The EM algorithm may be used to select the maximum probability association hypothesis over a window of frames. This is known as Probabilistic MHT (PMHT) and was first proposed by (Streit & Luginbuhl 1994). (Willett, Ruan & Streit 2002) show that performance is similar to that of the PDAF (The PMHT assumption is equivalent to running an independent PDAF on each target instead of a JPDAF on the whole lot).

MHT suffers from high computational loads due to the large number of possible association hypotheses. A particle method was introduced to address this problem in (Oh, Russell & Sastry 2004), called MCMC data association (MCMCDA). This method still uses Gaussian approximations so that KFs may be used to estimate the state distributions. However, instead of enumerating a posterior probability for every possible hypothesis, a Markov chain is constructed to target the posterior association hypothesis probability. This is a batch method, allowing changes to the associations in previous frames within some fixed length window. MH moves allow transitions between valid hypotheses, for example by initiating new tracks, extending or shortening tracks, or swapping observations between nearby targets. By choosing sensible proposals, low probability hypotheses are never even considered. In (Oh et al. 2004, Oh, Russell & Sastry 2009), the authors report significant improvements over MHT.

2.2.3 *Full particle filters*

In the MCMCDA method, a particle distribution was developed over the associations. In the MC-JPDAF, a particle filter was used for the target posterior state distributions. The next step is to maintain a particle distribution over both states and associations of all targets. The first method proposed along these lines was that of (Hue, Le Cadre & Perez 2002), followed by the SISR-based schemes of (Doucet, Vo, Andrieu & Davy 2002, Vermaak et al. 2005). The targetted distribution of the particle filter is the joint posterior of the target states and associations. Both these components must be proposed for each target, and the probability of each used in the weight updates.

Full multi-target particle filters suffer from complexity issues. The dimensionality of the state space scales with the number of targets. Thus the variance of importance weights, or acceptance probabilities, increases rapidly. An SISR particle filter with more than a couple of targets may repeatedly have all the weight on a single particle, even with resampling every step. An intuitive interpretation of the effect is that a particle may be given a low weight because its estimate of one target state is poor, even if the others are all good. (Orton & Fitzgerald 2002) proposes a method to combat this effect based on swapping particle states between particles, but this broadly equivalent to making an independence assumption. (Maskell, Rollason, Gordon & Salmond 2003) develops particle approximations for the target marginal densities with some interaction between the independent filters.

In (Vermaak et al. 2005), two strategies are proposed to cope with the dimensionality problems. In the Sequential Sampling Particle Filter (SSPF), targets are sampled sequentially, with optional resampling steps between each. This can be used to maintain a higher level of particle

diversity, but there is still a worse than linear scaling in complexity to maintain a constant level of accuracy. For the Independent Partition Particle Filter, targets are assumed to be completely independent, by allowing multiple targets to associate with the same observation, as with PMHT. This allows an independent particle filter to be run for each target from which the joint distribution may be reconstructed.

Only limited attention has been paid to joint detection and estimation in multi-target particle filters, due to the high computational complexities which result. (Vermaak et al. 2005, Horridge & Maskell 2009) introduce an existence variable, an indicator of whether a target exists or not, but this is handled in the style of a MC-JPDAF with the association and existence variables marginalised.

Interestingly, a marginalised or Rao-Blackwellised version of the full multi-target particle filter is equivalent to the MCMCDA approach, in that we return to a system where associations are estimated using a particle distribution and the states are estimated analytically with a Kalman filter. Such an algorithm using SISR is presented in (Särkkä, Vehtari & Lampinen 2007).

2.2.4 Probability hypothesis density methods

All the methods outlined so far assume that each target is identified by some unique label. There exists another family of tracking algorithms based on the assumption that the multi-target state is an unordered set, i.e. that it does not matter which target is which, only where they occur. The finite set statisitics (FISST) required to handle states which are random finite sets (RFS) was presented by (Mahler 1994). Practical algorithms based on RFS methods are generally restricted to estimating the first moment of the multitarget probability distribution, known as the "probability hypothesis density" (PHD), (Mahler 2003), which can be approximated using Gaussian mixtures, (Vo & Ma 2006), or particle filters, (Vo, Singh & Doucet 2005, Whiteley, Singh & Godsill 2010). A more detailed introduction to PHD methods can be found in (Mahler 2004, Wood 2010).

2.2.5 Bringing it all together

The methods we have considered can broadly be divided into those which attempt to explicity estimate the associations between targets and observations, and those that marginalise this information and estimate only a combined state estimate. We can also divide the methods into those which use Gaussian approximations and Kalman filters, and those which use particle approximations for the target states.

	Estimate Associations	Marginalise Associations
Gaussian State Approximation	MHT MCMCDA RBPF	JPDAF GM-PHD
Particle State Approximation	Full PF	MC-JPDAF SMC-PHD

Table 2.1: Loose grouping of target tracking algorithms by the treatment of associations and state approximations

3

Mathematical foundations

3.1 Bayes of our lives - A short introduction to filtering

For the literature review, mathematical detail was kept to a minimum for clarity of presentation. In this chapter we revisit some basics of inference and particle filtering in order to establish notation and mathematical foundations for the later chapters.

Many tasks in signal processing, science in general, and indeed life, require us to make some estimate of an unknown quantity from indirect, incomplete, or inaccurate observations. By constructing a model to explain how these observations depend on the underlying state, we can infer something about that state. We will express this observation model in terms of a likelihood function:

$$P(Y|X) \tag{3.1}$$

where X is the state and Y the observations. This is not the whole story - in many cases we are not estimating our unknown state "from scratch". Previous experience, prejudice, and prior knowledge can also contribute to our estimates. The likelihood and prior terms can be combined through our friend, Bayes rule (Bayes & Price 1763, Laplace 1774), to calculate the posterior probability of the state, i.e. the probability of the state given the observations:

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)} \tag{3.2}$$

This is the basis of the process of inference. Mathematically, we can assign a probability distribution to the state space of X. By applying Bayes rule, we are updating our belief about

the values of X using the information in Y.

Often the quantity in which we are interested, X, is changing over time, and we would like to estimate its value at each point in time given only the observations received so far. In this case we will generally assume that the state is Markovian, i.e. that $X_t|X_{t-1}$ is independent of $X_{1:t-2}$ in discrete time, giving us a hidden Markov state-space model. This is the traditional filtering problem. This model is shown graphically in figure 3.1.

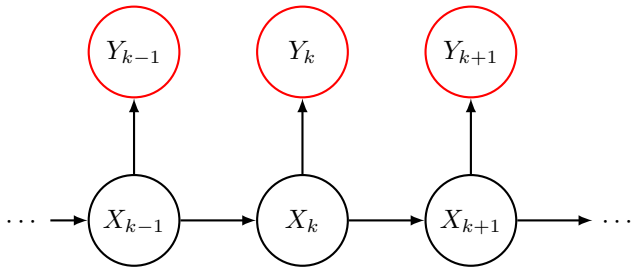


Figure 3.1: Graphical hidden Markov model.

By constucting a model for the evolution of the unknown state, we can now derive our prior information from our estimate at the previous time step. We now write:

$$P(X_t|Y_{1:t}) = \frac{\int P(Y_t|X_t)P(X_t|X_{t-1})P(X_{t-1}|Y_{1:t-1})dX_{t-1}}{P(Y_t|Y_{1:t-1})} \tag{3.3}$$

where the subscript indicates the time and ranges are notated by : in the MATLAB style. $P(X_t|Y_{1:t})$ is called the filtering distribution. Equation 3.3 describes the ideal Bayesian filter.

So far, we have expressed the problem entirely in terms of distributions. The same models may be expressed in terms of difference equations of random variables. In the most general form:

$$X_t = f_t(X_{t-1}, V_t) \tag{3.4}$$

$$Y_t = g_t(X_t, W_t) \tag{3.5}$$

where f_t and g_t are known deterministic functions and V_t and W_t and random variables, known as the process and observation noise respectively.

3.2 Keep Kalman carry on - the Kalman filter and its extensions

3.2.1 The basic filter

In a few simple cases the filtering set-up permits the derivation of closed form posterior distributions at each time instant. Most notably, the Kalman filter (KF) (Kalman 1960) is an analytic filter for models with continuous state and observation variables, in which both transition and observation models are linear transformations with Gaussian innovations.

$$x_t = Ax_t + v_t \tag{3.6}$$

$$y_t = Cx_t + w_t \tag{3.7}$$

where v_t and w_t are now Gaussian random variables with zero mean and covariance matrices Q and R. We use lower case variables here to emphasise that these are "nice", continuous vectors. (As we shall see, our state variable will later be sets or lists).

Kalman's solution for the linear-Gaussian case is given by:

$$P(x_t|y_{1:t}) = \mathcal{N}(x_t|\mu_t, \Sigma_t) \tag{3.8}$$

$$P(x_t|y_{1:t-1}) = \mathcal{N}(x_t|\hat{\mu}_t, \hat{\Sigma}_t) \tag{3.9}$$

where μ_t, Σ_t, etc. are given by the following recursions.

Time Updates:

$$\hat{\mu}_t = A\mu_{t-1} \tag{3.10}$$

$$\hat{\Sigma}_t = A\Sigma_{t-1}A^T + Q \tag{3.11}$$

Measurement Updates:

$$z_t = y_t - C\hat{\mu}_t \tag{3.12}$$

$$S_t = C\hat{\Sigma}_t C^T + R \tag{3.13}$$

$$K_t = \hat{\Sigma}_t C^T S_t^{-1} \tag{3.14}$$

$$\mu_t = \hat{\mu}_t + K_t z_t \tag{3.15}$$

$$\Sigma_t = (I - K_t C)\hat{\Sigma}_t \tag{3.16}$$

The KF is delightful because it not only provides us with a closed-form analytic solution, but

the complexity of that solution does not increase as we receive additional measurements. This is a consequence of the fact that the Gaussian distribution is its own conjugate prior. Unfortunately, no other such convenient cases have been discovered (Daum 2005). Analytic solutions to non-linear, non-Gaussian filtering problems generally require unacceptable conditions, such as zero process noise $Q = 0$ (Daum 2005).

3.2.2 The extended filter

Given the loveliness of the KF, the instinct when faced by an intractable non-linear filtering problem is to linearise it. This produces the Extended Kalman Filter (EKF), and is achieved by replacing the A and C matrices in equations 3.10 through 3.16 above with Jacobians:

$$A_t = \left.\frac{\partial f}{\partial x_t}\right|_{\mu_{t-1}} \tag{3.17}$$

$$C_t = \left.\frac{\partial g}{\partial x_t}\right|_{\hat{\mu}_t} \tag{3.18}$$

3.2.3 The Kalman smoother

The KF gives us an optimum estimate of $P(x_t|y_{1:t})$. However, once more data has arrived, we can improve this estimate. For a given set of data, $y_{1:T}$, we can estimate the optimum estimates for all previous state distributions, $P(x_{1:T}|y_{1:T})$ using a Rauch-Tung-Striebel (RTS) smoother, (Rauch et al. 1965). This begins with a normal KF, followed by a backward filtering pass which propagates information to earlier time instances. This backward pass is implement by the following recursions:

$$\tilde{\mu}_t = \mu_t + \Sigma_t A^T \hat{\Sigma}_{t+1}^{-1}(\tilde{\mu}_{t+1} - \hat{\mu}_{t+1}) \tag{3.19}$$

$$\tilde{\Sigma}_t = \Sigma_t + [\Sigma_t A^T \hat{\Sigma}_{t+1}^{-1}](\tilde{\Sigma}_{t+1} - \hat{\Sigma}_{t+1})[\Sigma_t A^T \hat{\Sigma}_{t+1}^{-1}]^T \tag{3.20}$$

giving us

$$P(x_t|Y_{1:T}) = \mathcal{N}(x_t|\tilde{\mu}_t, \tilde{\Sigma}_t) \tag{3.21}$$

For a full derivation, see (Rauch et al. 1965). There exist other ways to implement Kalman smoothing in a fixed-interval sense, such as the forward-backward smoother, and in a fixed-lag sense, but they will not be used in this work.

3.3 Tough as old bootstraps - the basic particle filter

In general we will not be so lucky as to have a problem with linear-Gaussian dynamics. In this case, a particle filter (PF) may be the best alternative. The PF is based on the idea that if we cannot represent a probability distribution analytically then we may approximate it as a collection of discrete samples or "particles" drawn from the distribution.

Often, it will not even be possible to sample from the desired distribution. We therefore use importance sampling (IS) or Markov chain Monte Carlo methods to generate these samples. The principle of a particle filter is that we recursively generate such a set of particles to approximate $P(X_{1:t}|Y_{1:t})$ given a previous set representing $P(X_{1:t-1}|Y_{1:t-1})$.

3.3.1 The basics

With a PF, we approximate a probability distribution with a set of (weighted) samples drawn from that distribution.

$$P(X) \approx \frac{1}{N} \sum_m W^{(m)} \delta_{X^{(m)}}(X) \tag{3.22}$$

where $\delta(x)$ represents a unit probability point mass at a point x, and $\sum_m W^{(m)} = 1$. Such a method allows us to represent any probability distribution of arbitrary complexity, including multidimensional, multimodal, mixed distributions. As the number of particles increases, the accuracy of the approximation improves at the expense of computational complexity. Thus we have the required tool for estimation in non-linear, non-Gaussian scenarios.

We still face the problem of how to generate these samples. The conventional method for this is Sequential Importance Sampling with Resampling (SISR). For a more complete and traditional introduction to this method, see (Cappé, Godsill & Moulines 2007) or (Doucet & Johansen 2009). Here we follow an outline similar to that used for the derivation of the auxiliary particle filter of (Pitt & Shephard 1999).

Suppose we have a particle approximation to the joint posterior distribution from the previous frame, $\hat{P}(X_{1:t-1}|Y_{1:t-1})$. Each particle represents a path through time, $X^{(m)}_{1:t-1}$, and has an associated weight, $W^{(m)}_{t-1}$. Let us suppose also that the unweighted particles may be considered samples from another distribution:

$$\mu(X_{1:t-1}|Y_{1:t-1}) \approx \frac{1}{N} \sum_m W^{(m)} \delta_{X^{(m)}_{1:t}}(X_{1:t}) \tag{3.23}$$

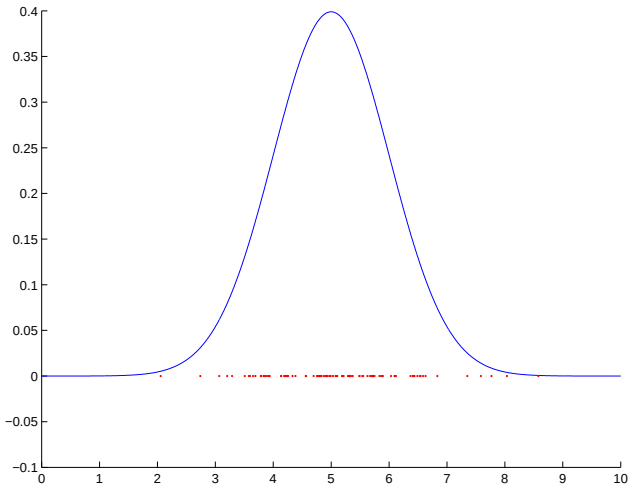


Figure 3.2: An analytic probability density (blue) may be approximated by a set of samples drawn from it (red)

Thus for a given particle,

$$\hat{P}(X_{1:t-1}^{(m)}|Y_{1:t-1}) = W_t^{(m)}\mu(X_{1:t-1}^{(m)}|Y_{1:t-1}) \tag{3.24}$$

We would like to generate a new particle set which includes the current time instance. We

propose a new set of extended tracks from a factored proposal distribution $X_{1:t} \sim q(X_{1:t}|Y_{1:t}) = q(X_t|Y_t, X_{1:t-1})q(X_{1:t-1}|Y_{1:t})$. The two factors are the proposal probabilities for the new state value, X_t and the history, $X_{1:t-1}$, respectively. Particles are then weighted to take account of the difference between the targeted posterior distribution and the importance distribution:

$$W_t^{(m)} = \frac{P(X_{1:t}^{(m)}|Y_{1:t})}{q(X_{1:t}^{(m)}|Y_{1:t})} \tag{3.25}$$

The simplest choice for the history proposal is $q(X_{1:t-1}|Y_{1:t}) = \mu(X_{1:t-1}|Y_{1:t-1})$, which can be implemented by simply keeping the same set of paths as the previous particle set. This is equivalent to an ordinary IS step with no resampling, and importance weights are given by:

$$W_t^{(m)} = \frac{P(X_{1:t}^{(m)}|Y_{1:t})}{q(X_{1:t}^{(m)}|Y_{1:t})} \approx \frac{W_{t-1}^{(m)} P(X_{1:t}^{(m)}|Y_{1:t})}{P(X_{1:t-1}^{(m)}|Y_{1:t-1})q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)} \propto \frac{W_{t-1}^{(m)} P(Y_t|X_t^{(m)})P(X_t^{(m)}|X_{t-1}^{(m)})}{q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)} \tag{3.26}$$

Alternatively, we could use $q(X_{1:t-1}|Y_{1:t}) = \hat{P}(X_{1:t-1}|Y_{1:t-1})$ as the history proposal, i.e. sample from the weighted particle distribution which approximates the previous posterior. This is equivalent to an IS step preceeded by resampling. Importance weights are now given by:

$$W_t^{(m)} = \frac{P(X_{1:t}^{(m)}|Y_{1:t})}{q(X_{1:t}^{(m)}|Y_{1:t})} \approx \frac{P(X_{1:t}^{(m)}|Y_{1:t})}{P(X_{1:t-1}^{(m)}|Y_{1:t-1})q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)} \propto \frac{P(Y_t|X_t^{(m)})P(X_t^{(m)}|X_{t-1}^{(m)})}{q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)} \tag{3.27}$$

3.3.2 Auxiliary sampling

We can generalise the form of our history proposal distribution by weighting the particles from the previous posterior distribution with any arbitrary set of weights.

$$q(X_{1:t-1}|Y_{1:t}) = \frac{1}{N}\sum_m V_t^{(m)} \delta_X(x_{1:t}^{(m)}) \tag{3.28}$$

Now we have

$$\hat{P}(X_{1:t-1}^{(m)}|Y_{1:t-1}) = \frac{W_{t-1}^{(m)}}{V_t^{(m)}} q(X_{1:t-1}^{(m)}|Y_{1:t}) \tag{3.29}$$

giving a general form for the importance weights

$$W_t^{(m)} = \frac{P(X_{1:t}^{(m)}|Y_{1:t})}{q(X_{1:t}^{(m)}|Y_{1:t})} \approx \frac{W_{t-1}^{(m)}}{V_t^{(m)}} \times \frac{P(X_{1:t}^{(m)}|Y_{1:t})}{P(X_{1:t-1}^{(m)}|Y_{1:t-1})q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)}$$
$$\propto \frac{W_{t-1}^{(m)}}{V_t^{(m)}} \times \frac{P(Y_t|X_t^{(m)})P(X_t^{(m)}|X_{t-1}^{(m)})}{q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)} \quad (3.30)$$

A suitable choice for the auxiliary proposal weights is the predictive likelihood of the next measurement, i.e.

$$V_t^{(m)} = P(Y_t|X_{t-1}^{(m)}) \quad (3.31)$$

This favours the selection of particles which better explain the new observation. This quantity is often not easily calculable, so approximations may be used, for example using a Gaussian approximation.

3.3.3 *Degeneracy and resampling*

The simplest history proposal is given by $q(X_{1:t-1}|Y_{1:t}) = \mu(X_{1:t-1}|Y_{1:t-1})$. This may be 'sampled' by simply keeping the N particles from the previous processing step, which is thus equivalent to importance sampling with no resampling. Multiple steps of this nature will lead to a fall in particle diversity, as many of the weights tend towards 0. This occurs because of the recursive nature of the importance weight calculation in equation **??**. The solution to this degeneracy is to use the other form of history proposal, $q(X_{1:t-1}|Y_{1:t}) = \hat{P}(X_{1:t-1}|Y_{1:t-1})$, equivalent to importance sampling with resampling. Such a proposal will require additional computation time, but the weight updates are no longer recursive, see equation **??**, so diversity is improved.

The resampling/history-sampling may be conducted in a variety of ways. The simplest is multinomial sampling, in which each particle is sampled independently with replacement. In residual sampling, particles are sampled from the particle distribution without replacement, i.e. after a particle is selected, the probability of selecting it again is reduced. Finally, for systematic resampling, a finite real line is divided into sections corresponding to the weight of each particle. Particles are sampled by choosing regular points along this line, with some constant, random offset. The final method produces the least variance in the number of child particles chosen from each parent given the weights, and thus introduces the least Monte Carlo error. For more details see (Doucet & Johansen 2009) and the references therein.

Degeneracy is measured using the Effective Sample Size of (Liu & Chen 1995), which is given

by:

$$ESS = \frac{1}{N}\sum_m W^{(m)2} \tag{3.32}$$

where $W^{(m)}$ are the particle weights, and there are N particles. If ESS falls below some chosen threshold, resampling this required to replenish particle diversity.

3.3.4 Importance distributions

It remains to choose the the importance distribution for the current state, $q(X_t|X_{t-1}, Y_t)$. In the original "bootstrap filter" of (Gordon et al. 1993), this was set equal to the transition density, $P(X_t|X_{t-1})$, which leads to cancellation in the expression for importance weights. This is simple but not necessarily optimal. For example if the process noise is high, the samples of X_t will be widely spread. If, however, the observation noise is comparitively low, many or most of the samples will be far from the observation and will have a low weight, giving us poor particle diversity.

An improved choice of proposal distribution is given by:

$$q(X_t|X_{t-1}, Y_t) = P(X_t|X_{t-1}, Y_t) = \frac{P(Y_t|X_t)P(X_t|X_{t-1})}{P(Y_t|X_{t-1})} \tag{3.33}$$

This is often known as the "optimal importance distribution". The importance weights for a step without resampling are now given by:

$$W_t^{(m)} \propto \frac{W_{t-1}^{(m)}}{V_t^{(m)}} \times \frac{P(Y_t|X_t^{(m)})P(X_t^{(m)}|X_{t-1}^{(m)})}{q(X_t^{(m)}|X_{t-1}^{(m)}, Y_t)} \propto \frac{W_{t-1}^{(m)}}{V_t^{(m)}} \times P(Y_t|X_{t-1}^{(m)}) \tag{3.34}$$

If the auxiliary weights are set to $V_t^{(m)} = P(Y_t|X_{t-1}^{(m)})$ then all weights are equal throughout, and resampling will never be required. In general, the optimal importance distribution will not be samplable, but proposals using Gaussian approximations of this form may be used.

3.4 MCMC in da house - Markov chains for particle filtering

The previous sections have focussed on the traditional, SISR particle filter. Here we examine the MCMC version. The mathematical extensions required are minimal.

MCMC methods allow us to generate an unweighted particle approximation to a target probability distribution. Particles are generated sequentially, each dependent on the last, giving us a Markov chain whose stationary distriubtion is equal to that which we wish to approximate.

For a basic MH step, a proposal distribution is sampled to acquire a new candidate state. With some probability this candidate is accepted as the new state. Otherwise, it is rejected and the old state is kept as the new state. The acceptance probability required to make $P(X)$ the stationary distribution of the Markov chain is given by:

$$\alpha = \min\left(1, \frac{P(X_{\text{new}})q(X_{\text{old}}|X_{\text{new}})}{P(X_{\text{old}})q(X_{\text{new}}|X_{\text{old}})}\right) \tag{3.35}$$

This will be appear familiar as the ratio of the importance weights the old and candidate states would receive in a SISR scheme. In a MCMC particle filtering scheme we target the usual joint posterior distribution:

$$P(X_{1:t}|Y_{1:t}) = \frac{P(Y_t|X_t)P(X_t|X_{t-1})P(X_{1:t-1}|Y_{1:t-1})}{P(Y_t|Y_{1:t-1})} \tag{3.36}$$

As for the SISR particle filter, the proposal distribution factorises as $X_{1:t} \sim q(X_{1:t}|Y_{1:t}) = q(X_t|Y_t, X_{1:t-1})q(X_{1:t-1}|Y_{1:t})$. We can use a biased history proposal in the flavour of the auxiliary particle filter, as described by equation 3.28.

We now adopt the following notation: X for the new state of the Markov chain, and Z for the old state, with $V_{X,t}$ and $V_{Z,t}$ respectively for the auxiliary proposal weights. Using these expansions, the acceptance probability is given by:

$$\begin{aligned}\alpha = \min\left(1, \frac{P(X_{1:t}|Y_{1:t})q(Z_{1:t}|Y_{1:t}}{P(Z_{1:t}|Y_{1:t})q(X_{1:t}|Y_{1:t}}\right) \\ = \min\left(1, \frac{P(Y_t|X_t)P(X_t|X_{t-1})q(Z_t|Y_t, Z_{1:t-1})V_{Z,t}}{P(Y_t|Z_t)P(Z_t|Z_{t-1})q(X_t|Y_t, X_{1:t-1})V_{X,t}}\right)\end{aligned} \tag{3.37}$$

The advantage of using a MCMC scheme is that we can vary only some subset of the state variables in each move if we choose. For example, we can have one sort of MH move which keeps the history the same, $X_{1:t} = Z_{1:t}$. This leads to a simplification of the equation for α and an improved probability of acceptance.

4

The tracking model

In this chapter we introduce the mathematical model for the object tracking problem which will be used throughout the rest of this report.

We first assume that our targets exist at singular points in space, and that targets move independently of each other. Each target will have be characterised by a state vector $x_{j,t}$. This state is composed of continuous-valued coordinates and evolves according to a hidden Markov model (HMM). At first, we will also assume that we know how many targets are in the scene and their starting locations, $x_{j,0}$.

$$x_{j,t} = f(x_{j,t-1}, v_{j,t}) \tag{4.1}$$

where $v_{j,t}$ is a random vector.

The targets are observed by a sensor which detects each with a probability P_D. If detected, the sensor returns a point observation at a location $y_t^{(i)}$ given by:

$$y_t^{(i)} = g(x_{j,t}, w_{j,t}) \tag{4.2}$$

where $w_{j,t}$ is another random vector, independent of $v_{j,t}$.

In addition to the target-originating observations, the sensor also detects a number of false alarms. We assume that these are generated by a Poisson process, with a uniform intensity over the observation area. The expected number of clutter observations in a frame is denoted μ_C.

We denote the set of targets present as $X_t = \{x_{1,t}, x_{2,t}, ..., x_{K_t,t}\}$, and the set of observations as $Y_t = \{y_t^{(1)}, y_t^{(2)}, ..., y_t^{(M_t)}\}$.

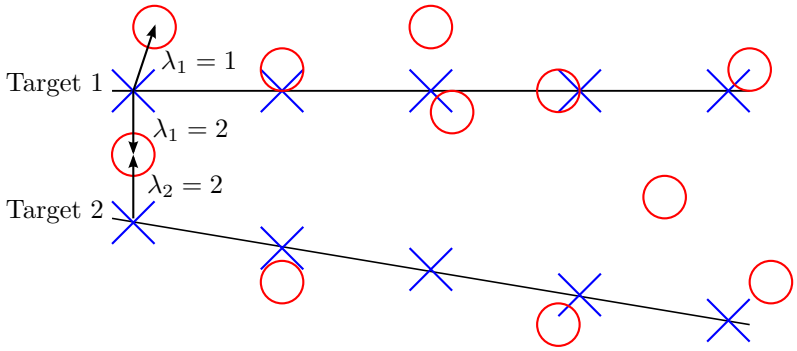


Figure 4.1: A simple two-target tracking problem illustrating the different possible associations at one time step. Hidden states shown as blue crosses. Observations shown as red circles.

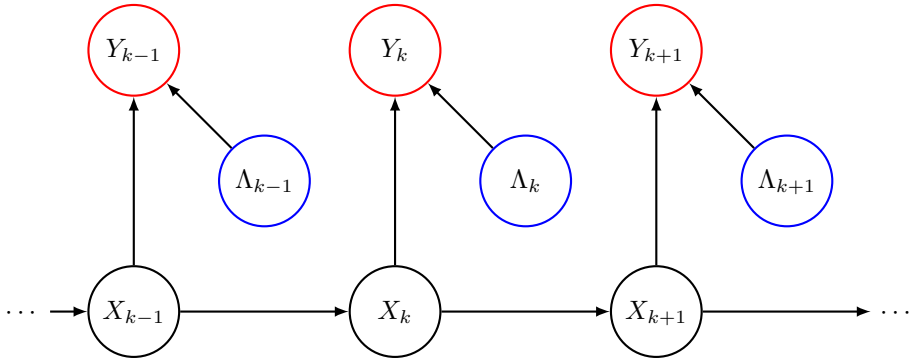


Figure 4.2: Graphical model for tracking including association variables.

We introduce an association variable for each target, $\lambda_{j,t}$, which indicates which of the observations in that frame was generated by this target, as shown in figure 5.1. If the target is not detected, then $\lambda_{j,t}$ is set to 0. We denote the set $\Lambda_t = \{\lambda_{1,t}, \lambda_{2,t}, ..., \lambda_{K_t,t}\}$. As each observation is generated by one target or clutter, no two elements of Λ_t may take the same value, unless 0. (This is the constraint that the PMHT and PDAF relax.)

The graphical model including the association variables is shown in figure 4.2.

If we wished to calculate the posterior distribution over the target states, X_t, we would need to marginalise the association variables in the calculation of the likelihood term:

$$P(Y_t|X_t) = \sum_{\Lambda_t} P(\Lambda_t) \prod_{j=1}^{K_t} P(y_t^{(\lambda_{j,t})}|x_{j,t}) \tag{4.3}$$

where the summation is over all feasible values of Λ_t. The number of terms in this summation combinatorially in the number of targets and the number of observations, and may be prohibitively large if there are many targets or observations in the scene. If it is necessary to calculate the likelihood very often, as in a particle filter, it may be preferable to estimate Λ_t instead of marginalising it. The target distribution is now:

$$P(X_{1:t}, \Lambda_{1:t}|Y_{1:t}) = \frac{P(Y_t|X_t, \Lambda_t)P(X_t|X_{t-1})P(\Lambda_t)P(X_{1:t-1}, \Lambda_{1:t-1}|Y_{1:t-1})}{P(Y_t|Y_{1:t-1})} \tag{4.4}$$

We now consider the terms of this equation one at a time.

4.1 Observation Likelihood

The likelihood given the associations is given by:

$$P(Y_t|X_t, \Lambda_t) = V^{-(M_t - M_{T,t})} \prod_{\lambda_{j,t} \neq 0} P(y_t^{(\lambda_{j,t})}|x_{j,t}) \tag{4.5}$$

where V is the volume of the observation region, and thus V^{-1} is the likelihood of a clutter obervation, and where $M_{T,t} = |\{\lambda_{j,t} \neq 0\}|$. This can be conveniently factorised as

$$P(Y_t|X_t, \Lambda_t) = V^{-(M_t - K_t)} \prod_{j=1}^{K_t} \begin{cases} P(y_t^{(\lambda_{j,t})}|x_{j,t}) & \lambda_{j,t} \neq 0 \\ V^{-1} & \lambda_{j,t} = 0 \end{cases} \tag{4.6}$$

This expression contains a factor for each target, and a coefficient independent of the state. This coefficient will either cancel out (MCMC) or be discarded in the weight normalisation (SISR).

4.2 Transition Dynamics

Similarly, because of our independence assumption on the targets, the transition dynamic term will factorise:

$$P(X_t|X_{t-1}) = \prod_{j=1}^{K_t} P(x_{j,t}|x_{j-1}) \tag{4.7}$$

4.3 Association Likelihood

The association variable, λ_t, tells us how many targets have been detected (and how many have not), and how many clutter observations are present. To construct a prior distribution on this, consider the $\lambda_{j,t}$s in turn. Each one can be either 0 with probability $(1 - P_D)$, or an observation index with probability P_D. If it is the latter, there are $M_t - M_{\text{taken}}$ indexes from which to choose, where M_{taken} is the number of observations already assigned to a previous target. Once we have considered all the targets, $M_{T,t}$ observations will have been assigned to targets, and all those remaining must be generated by the clutter process. The number of these is Poisson distributed. The association prior for feasible is Λ_t thus given by:

$$P(\Lambda_t) = \underbrace{\frac{P_D^{M_t}}{M_t(M_t-1)...(M_t-M_{T,t})}}_{\text{detected targets}} \underbrace{(1-P_D)^{(K-M_t)}}_{\text{undetected targets}} \underbrace{\frac{\exp(-\mu_C)\mu_C^{(M_t-M_{T,t})}}{(M_t-M_{T,t})!}}_{\text{clutter}}$$
$$= \frac{\exp(-\mu_C)\mu_C^{(M_t-M_{T,t})} P_D^{M_t}(1-P_D)^{(1-M_t)}}{M_t!} \quad (4.8)$$

while for invalid Λ_t ($\lambda_{i,t} = \lambda_{j,t}$ for $i \neq j$, $\lambda_{i,t}, \lambda_{j,t} \neq 0$), the probability is zero. Again, we can factorise this expression over the targets.

$$P(\lambda_t) = \frac{\exp(-\mu_C)\mu_C^{(M_t-K_t)}}{M_t!} \prod_{j=1}^{K_t} \begin{cases} P_D & \lambda_{j,t} \neq 0, \lambda_{j,t} \notin \{\lambda_{1:j-1,t}\} \\ 0 & \lambda_{j,t} \in \{\lambda_{1:j-1,t}\} \\ (1-P_D)\mu_C & \lambda_{j,t} = 0 \end{cases} \quad (4.9)$$

Just as for the observation likelihood, this expression has a factor for each target and a state-independent coefficient which can be ommitted for either a SISR or MCMC particle filter.

4.4 Assumptions

Here we summarise the assumptions made in the model formulations:

- Targets and observations occur at singular points.
- The number of targets is known.
- Target starting locations known.
- Target states evolve independently of each other according to a HMM.
- Targets are detected in each frame with an independent probability P_D. A detected target

generates one observation.

- A single target is associated with each observation.
- Clutter observations are generated by a homogeneous spatial Poisson process.

4.5 Specific models

4.5.1 State transitions

In this work we employ a Near-Constant Velocity (NCV) model. We omit the target index j in the following and consider only a single spatial dimension. The extension to multiple dimensions is trivial. A target state, x, is written as a vector:

$$x = \begin{pmatrix} s \\ \dot{s} \end{pmatrix} \tag{4.10}$$

We define the target motion by a simple stochastic differential equation, in which acceleration is a Wiener process.

$$d\dot{s}_\tau = \sigma dW_\tau \tag{4.11}$$

$$ds_\tau = \dot{s}_\tau \tag{4.12}$$

Integrating these gives us

$$\dot{s}_\tau = \dot{s}_0 + \sigma W_\tau \tag{4.13}$$

$$s_\tau = s_0 + \dot{s}_0\tau + \sigma \int_0^\tau W_s ds \tag{4.14}$$

which we can rewrite in matrix form

$$\begin{bmatrix} s_\tau \\ \dot{s}_\tau \end{bmatrix} = \begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_0 \\ \dot{s}_0 \end{bmatrix} + \sigma \begin{bmatrix} W_\tau \\ \int_0^\tau W_s ds \end{bmatrix} \tag{4.15}$$

It is thus clear that the $x_\tau | x_0$ is normally distributed with mean x_0 and covariance

$$\mathbb{V}[x_\tau] = \sigma^2 \mathbb{E} \begin{bmatrix} W_\tau^2 & W_\tau \int_0^\tau W_s ds \\ W_\tau \int_0^\tau W_s ds & \left(\int_0^\tau W_s ds\right)^2 \end{bmatrix} = \sigma^2 \begin{bmatrix} \tau & \frac{\tau^2}{2} \\ \frac{\tau^2}{2} & \frac{\tau^3}{3} \end{bmatrix} \tag{4.16}$$

The model can now be discretised onto a set of time point, indexed with t. The transition density is linear-Gaussian given by:

$$P(x_{t+1}|x_t) = \mathcal{N}(x_{t+1}|Ax_t, Q) \tag{4.17}$$

Throughout this report, we will consider models in two spatial dimensions. Thus the matrices A and Q are given by:

$$A = \begin{bmatrix} 1 & 0 & P & 0 \\ 0 & 1 & 0 & P \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.18}$$

$$Q = \sigma^2 \begin{bmatrix} P & 0 & \frac{P^2}{2} & 0 \\ 0 & P & 0 & \frac{P^2}{2} \\ \frac{P^2}{2} & 0 & \frac{P^3}{3} & 0 \\ 0 & \frac{P^2}{2} & 0 & \frac{P^3}{3} \end{bmatrix} \tag{4.19}$$

and P is the sampling period.

4.5.2 Observations

We will use two different observation models. Firstly, linear-Gaussian where the position is observed directly:

$$P(y_t^{\lambda_t}|x_t) = \mathcal{N}(y_t^{\lambda_t}|Cx_t, R) \tag{4.20}$$

where C is:

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

Secondly, we consider a polar observation model, of the form used for radar systems.

$$P(y_t^{\lambda_t}|x_t) = \mathcal{N}(y_t^{\lambda_t}|h(x_t), R) \tag{4.22}$$

where

$$h(x_t) = \begin{bmatrix} \arctan\left(\frac{x_{2,t}}{x_{1,t}}\right) \\ \sqrt{(x_{1,t}^2 + (x_{2,t}^2} \end{bmatrix} \tag{4.23}$$

We will need a linearised, EKF approximation of this model for use in proposal distributions. This will require the jacobian of the observation function:

$$C_t = \begin{bmatrix} \frac{\partial h_1}{\partial x_{1,t}} & \frac{\partial h_1}{\partial x_{2,t}} & \frac{\partial h_1}{\partial \dot{x}_{1,t}} & \frac{\partial h_1}{\partial \dot{x}_{2,t}} \\ \frac{\partial h_2}{\partial x_{1,t}} & \frac{\partial h_2}{\partial x_{2,t}} & \frac{\partial h_2}{\partial \dot{x}_{1,t}} & \frac{\partial h_2}{\partial \dot{x}_{2,t}} \end{bmatrix} = \begin{bmatrix} \frac{-x_{2,t}}{x_{1,t}^2+x_{2,t}^2} & \frac{x_{1,t}}{x_{1,t}^2+x_{2,t}^2} & 0 & 0 \\ \frac{x_{1,t}}{\sqrt{x_{1,t}^2+x_{2,t}^2}} & \frac{x_{2,t}}{\sqrt{x_{1,t}^2+x_{2,t}^2}} & 0 & 0 \end{bmatrix} \tag{4.24}$$

The effect of such an approximation is shown in figure 4.3

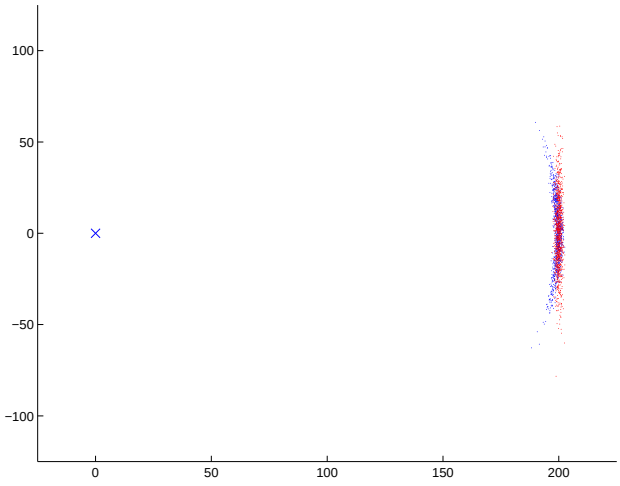


Figure 4.3: Samples from the nonlinear observation distribution (blue) and from the EKF approximation (red).

5

Fixed Lag Estimation for Tracking

5.1 The benefits of delaying

A basic filter, the Kalman filter included, allows us to estimate the latest value of a hidden state given all previous observations. However, if we wait until the next observation arrives, we can often make a much better estimate - we now know where the state is going as well as where it came from. This is the idea exploited by the Kalman smoother. In fact, as we have seen, the regular particle filter gives us an estimate of the entire state history, $X_{1:t}$. However, due to the resampling process, the particle diversity of the state distribution for previous frames falls as we move further back in time.

As well as giving poor estimates for state distributions, a further disadvantage of this falling diversity in a tracking context is a tendency to lose tracks. Suppose that in some frame the majority of particles associate with a particular likely-looking observation and head off in one direction, but that a few frames later it becomes apparent that this is an error, and that the correct route goes another way. With an ordinary SISR particle filter the majority of the particles are now eliminated by resampling. The particle diversity of the state estimate will be very poor, and in a challenging problem there is a good chance that the track will be lost.

Improved estimates for previous state distributions can be made using the resample-move (RM) method of (Gilks & Berzuini 2001) to rejeuvenate the particle diversity. However, this requires running at least one MH move for each particle on top of the usual importance sampling step, which will be computationally expensive. Furthermore, the RM steps will cause most rejeuvenation in the most recent frames, which will be discarded in the next frame.

An alternative framework for fixed lag estimation is presented by (Doucet et al. 2006). Now

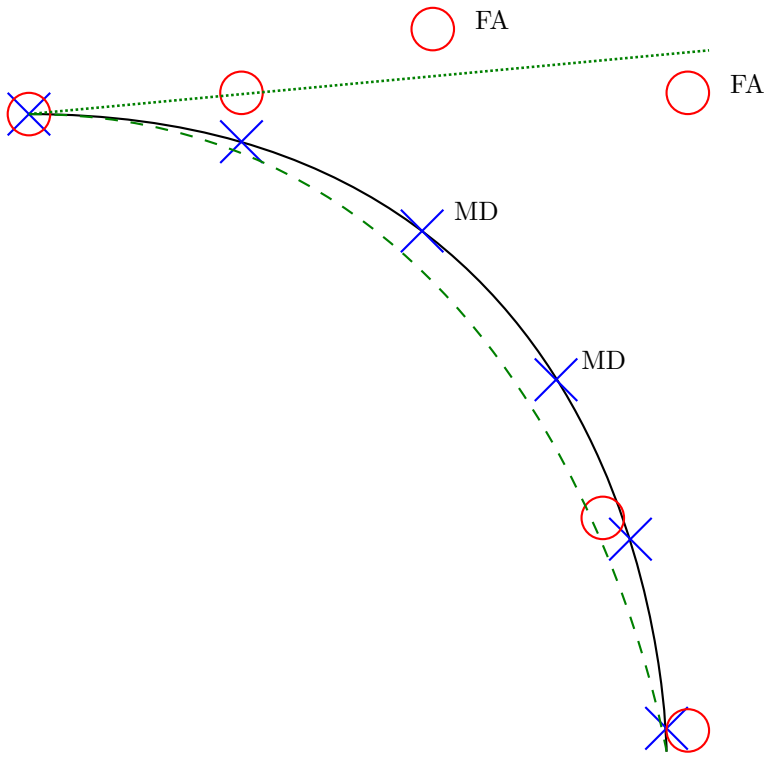


Figure 5.1: An example of particle redirecting. Hidden states shown as blue crosses. Observations shown as red circles. FA and MD indicate a false alarm and missed detection respectively. After four time steps, the path shown in green dots is most likely. After two further steps, the most likely path is correct, shown in green dashes.

we propose new values not only for the current state but also the previous states within some time window. The effect of this in our tracking example is that we can 'redirect' particles which have gone astray along the correct path, maintaining a better particle diversity and reducing the probability of track loss. (Doucet et al. 2006) report that RM does not perform as well as their fixed-lag estimation method, although the two shemes are related.

5.2 A mathematical framework for fixed lag estimation

Here we consider the mathematical framework for fixed-lag estimatation, as presented by (Doucet et al. 2006) and (Briers, Doucet, Maskell & Horridge 2006).

As before, the target posterior distribution in which we are interested is the familiar $P(X_{1:t}|Y_{1:t})$.

However, the proposal mechanism now becomes more complex, because we will be replacing states in an existing particle. We first propose a particle from which to take the state "history", that is $X_{1:t-L}$. However, if we take this from the particle distribution from the previous step we get more of the path than we need, because each particle is a set of states $X_{1:t-1}$. The final $L-1$ states will be replaced when by a new "present", $X'_{t-L+1:t}$, drawn from an importance distribution. The complete proposal is thus

$$\{X_{1:t-L}, X'_{t-L+1:t}\} \sim \int q(X_{1:t-1}|Y_{1:t-1})q(X'_{t-L+1}|X_{1:t-1}, Y_{t-L+1:t})dX_{t-L+1:t-1} \tag{5.1}$$

where $q(X_{1:t-1}|Y_{1:t-1})$ is a proposal distribution using the arbitrarily-weighted particles from $\hat{P}(X_{1:t-1}|Y_{1:t-1})$, in the auxiliary sampling sense. This integral is required for calculation of importance weights of acceptance probabilities, but in general it will be intractable. If we were to restrict our proposals to depend only on the history, i.e. use $q(X_{t-L+1:t}|X_{1:t-L}, Y_{t-L+1:t})$, then the proposal would become

$$\{X_{1:t-L}, X'_{t-L+1:t}\} \sim \hat{P}(X_{1:t-L}|Y_{1:t-1})q(X'_{t-L+1}|X_{1:t-1}, Y_{t-L+1:t}) \tag{5.2}$$

However, with this proposal, in order to evaluate importance weights or acceptance probabilities we still need to calculate

$$P(X_{1:t-L}|Y_{1:t-1}) = \frac{P(Y_{t-L+1:t-1}|X^{(m)}_{t-L})P(X_{1:t-L}|Y_{1:t-L}}{P(Y_{t-L+1:t-1}|Y_{1:t-L})} \tag{5.3}$$

The problematic term is:

$$P(Y_{t-L+1:t}|X^{(m)}_{t-L}) = \int P(Y_{t-L+1:t}|X^{(m)}_{t-L}, X_{t-L+1:t})P(X_{t-L+1:t}|X^{(m)}_{t-L})dX_{t-L+1:t} \tag{5.4}$$

This is intractable. The solution to this problem proposed in (Doucet et al. 2006) is to augment the dimension of the target distribution to include the discarded tracks. The new target is:

$$P(X_{1:t-L}, X'_{t-L+1:t}|Y_{1:t})\rho(X_{t-L+1:t-1}|X_{1:t-L}, X'_{t-L+1:t}) \tag{5.5}$$

Once a particle approximation has been generated for this target the required posterior distribution may be obtained by marginalisation. As we are simply going to discard the old

track sections $X_{t-L+1:t-1}$, the choice of $\rho(.)$ does not alter the distribution of the posterior. However, the quality of the particle approximation may be affected.

With an expanded space for the target distribution there is no need to marginalise the previous states. The proposal distribution now becomes:

$$\{X_{1:t-1}, X'_{t-L+1:t}\} \sim q(X_{1:t-1}|Y_{1:t-1})q(X'_{t-L+1:t}|X_{1:t-1}, Y_{t-L+1:t}) \tag{5.6}$$

Histories are proposed from the $t-1$ frame in the normal manner, see equation 3.28.

(Doucet et al. 2006) show that the optimum choice (in the minimal importance weight variance sense) of artifical conditional distribution is given by

$$\rho(X_{t-L+1:t-1}|X_{1:t-L}, X'_{t-L+1:t}) = P(X_{t-L+1:t-1}|X_{t-L}, Y_{t-L+1:t-1}) \tag{5.7}$$

with the current proposal distribution set to

$$q(X'_{t-L+1:t}|X_{1:t-s}, Y_{t-L+1:t}) = P(X'_{t-L+1:t}|X_{t-L}, Y_{t-L+1:t}) \tag{5.8}$$

Thus both artificial conditional and proposal distribution should equal to the conditional posterior. This is neither samplable nor calculable, and approximations will be required.

5.3 Application to tracking

5.3.1 *Importance distributions*

The dimensionality of a fixed lag particle filter will generally be very large. Consider a problem where each target has a 4-dimensional kinematic state (postion and velocity in x and y) and an observation-association index. This gives us five dimensions per target per time step. If we use a lag window with length $L = 5$ and 5 targets we will have a 125 dimensional state. If we used a basic, bootstrap approach to such a problem, the chances of even a single particle following the correct path are negligible. Instead we must exploit the strong correlations between states arising from the structure of the problem. This is achieved by designing better proposal distributions. We aim to approximate the "optimal" importance distribution of equation 3.33. The same approximation will also be useful as the artifical conditional distribution required for extending the state-space for the fixed lag particle filters. We first factorise the proposal over the targets:

$$q(X_{t-L+1:t}, \Lambda_{t-L+1:t}|X_{t-L}, Y_{t-L+1:t}) = \prod_{j=1}^{K_t} q(x_{j,t-L+1:t}, \lambda_{j,t-L+1:t}|\lambda_{1:j-1,t-L+1:t}, x_{j,t-L}, Y_{t-L+1:t}) \tag{5.9}$$

Thus, the targets may be proposed sequentially. The proposal for each target is then further decomposed.

$$\begin{aligned} q(x_{j,t-L+1:t}, \lambda_{j,t-L+1:t}|\lambda_{1:j-1,t-L+1:t}, x_{j,t-L}, Y_{t-L+1:t}) & \\ &= q(\lambda_{j,t-L+1:t}|\lambda_{1:j-1,t-L+1:t}, x_{j,t-L}, Y_{t-L+1:t}) \\ &= q(x_{j,t-L+1:t}|x_{j,t-L}, \lambda_{j,t-L+1:t}, Y_{t-L+1:t}) \end{aligned} \tag{5.10}$$

Thus we can sample first the association variables then the state variables. Each of these terms can be further factorised over time, as we shall see. If we were to replace each of the factors with its corresponding posterior distribution then this factorsiation would recreate the optimal importance distribution.

Association proposals

We first consider a proposal distribution for a single target. The proposal for the association variables is a discrete distribution over the possible observations with which the target could be associated. The "optimal" form of the proposal is:

$$\begin{aligned} q(\lambda_{t-L+1:t}|x_{t-L}, Y_{t-L+1:t}) & \\ &\propto P(Y_{t-L+1:t}|\lambda_{t-L+1:t}, x_{t-L})P(\lambda_{t-L+1:t}|x_{t-L}) \\ &= \prod_{\substack{k=1:L \\ tt=t-L+k}} P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L})P(\lambda_{tt}) \\ &= \prod_{\substack{k=1:L \\ tt=t-L+k}} \int P(Y_{tt}|x_{tt}, \lambda_{tt})P(x_{tt}|Y_{tt+1:t}, \lambda_{tt+1:t}, x_{t-L})dx_{tt}P(\lambda_{tt}) \end{aligned} \tag{5.11}$$

This suggests a convenient sequential sampling procedure, starting with λ_t and working backwards in time. The nomalisation constant is not known, but as this is discrete distribution we can enforce normalisation by dividing by the sum.

First we consider a factor from this expression with $\lambda_{tt} = 0$, i.e. a proposal that in a particular frame the target is not detected. In this case, the observation density is independent of the state of the target, and we have:

$$P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L})P(\lambda_{tt}) = V^{-1}P(\lambda_t = 0) \tag{5.12}$$

When the target is detected, the factors of the proposal ditribution of equation 5.11 may be calculated analytically for the linear-Gaussian case. For other cases, the EKF approximations may be used. As this is only a proposal distribution, such an approximation will not affect the distribution of the particles generated. The state distribution term over x_{tt} is problematic, requiring k integrals to calculate. Although this is will be analytic with Gaussian dynamics, its complexity may become unmanageable. This term represents the probability of the state given the set of observations associated with this target at later times. We can render this calculation more manageable by replacing the whole set of future observations with just one.

$$P(x_k|Y_{k+1:t}, \lambda_{k+1:t}, x_{t-L}) \approx P(x_k|Y_{k+d}, \lambda_{k+d}, x_{t-L}) \tag{5.13}$$

For cases with low observation noise, where an observation gives us significant information about the state of the target, this substitution will have little effect. Later observations cannot add much additional information. Again, as this is a proposal distribution, such a substitution will not affect the validity of the resulting particle distribution.

In general we will use $d = 1$, as the closest observation in time will give us the most information about x_{tt}. However, if the target is not detected at time $tt+1$, then we can increase d to pick out the next detection of the target.

Using this approximation, for $\lambda_{t-L+k} \neq 0$, we have

$$P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L})P(\lambda_{tt}) \propto \mathcal{N}(y_{tt}^{(\lambda_{tt})}|m_{tt}, S_{tt}) \tag{5.14}$$

where usually

$$S_{tt} = [I - R^{-1}C_{tt}\Sigma_{tt}C_{tt}R^{-1}]^{-1} \tag{5.15}$$

$$m_{tt} = SR^{-1}C_{tt}\Sigma_{tt}[(A^d)^T C_{tt+d}^T R_d^{-1} y_{tt+d}^{\lambda_{tt+d}} + Q_k^{-1}A^k x_{t-L}] \tag{5.16}$$

$$\Sigma_{tt} = [C_{tt}^T R^{-1} C_{tt} + (A^d)^T C_{tt+d}^T R_d^{-1} C_{tt+d} A^d + Q_k^{-1}]^{-1} \tag{5.17}$$

$$Q_d = \sum_{l=0}^{d-1} A^l Q (A^l)^T \tag{5.18}$$

$$R_d = R + C_{tt+d} Q_d (C_{tt+d})^T \tag{5.19}$$

We will need a different expression for the case when $tt = t$, because no future associations have yet been proposed. Similarly, if $tt < t$ but the future associations have all been proposed as missed detections, then there are no future observations to guide us, whatever choice of d we use. In these cases we have:

$$S_{tt} = R_d \tag{5.20}$$

$$m_{tt} = C_{tt} A^k x_{t-L} \tag{5.21}$$

For full derivations, see Appendix.

Finally, substituting for the association prior terms, we have:

$$q(\lambda_{t-L+1:t} | x_{t-L}, Y_{t-L+1:t}) \propto \prod_{\substack{k=1:L \\ tt=t-L+k}} \begin{cases} P_D \mathcal{N}(y_{tt}^{(\lambda_{tt})} | m_{tt}, S_{tt}) & \lambda_{tt} = 0 \\ (1 - P_D) \mu_C V^{-1} & \lambda_{tt} \neq 0 \end{cases} \tag{5.22}$$

This scheme can be extended to multiple targets by simply sampling targets sequentially. The only modification required is that the proposal probability is set to zero if an observation has already been associated with another target.

Realistically, we cannot calculate a proposal weight for all possible associations. If we have a large observation area and many false alarms this would require a vast number of calculations of weights which would mostly be negligible. Instead we use a gating procedure, considering only targets within a certain Mahalanobis squared distance of the proposal mean. i.e. those for which

$$(y_{tt}^{(\lambda_{tt})} - m_{tt})^T S_{tt}^{-1} (y_{tt}^{(\lambda_{tt})} - m_{tt}) < M_{\text{thresh}} \tag{5.23}$$

This Mahalanobis squared distance will be chi-square distributed with d degrees of freedom, where d is the dimension of y_{tt}. To demonstrate this, consider the definition of a chi-square variable as (time indices omitted)

$$\chi_d^2 = \sum_{i=1}^{d} z_i^2 = z^T z \tag{5.24}$$

where z_i are i.i.d standard normal variables. Now apply a transformation

$$y = m + Uz\Lambda^{\frac{1}{2}} \tag{5.25}$$

where U and Λ are derived from the eigenvalue decompostion

$$S = U\Lambda U^t \tag{5.26}$$

Thus

$$\chi^2_d = (y-m)\Sigma^{-1}(y-m) \tag{5.27}$$

Hence M_{thresh} can be chosen from the tabulated cumulative chi-square distribution to ensure a given probability of catching the correct observation in the gate. Choosing $M_{\text{thresh}} = 16$ gives a probability of just 0.03% of excluding the correct observation with 2 degrees of freedom. Note however that when a nonlinear model is in use this will rise significantly due to the linear approximation. A generous choice of $M_{\text{thresh}} = 25$ is thus used for nonlinear cases.

State proposals

Once the associations are fixed, the states can be proposed. When the state space model is linear-Gaussian, we can propose directly from the "optimal" importance distribution for the states using the forward-filtering-backward-sampling algorithm of (Chib 1996), as suggested in (Doucet et al. 2006). For nonlinear models, we can use EKF approximations, as for the associations. Once again, we factorise the proposal:

$$\begin{aligned} &q(x_{t-L+1:t}|x_{t-L}, \lambda_{t-L+1:t}, Y_{t-L+1:t}) \\ &\qquad = P(x_{t-L+1:t}|x_{t-L}, \lambda_{t-L+1:t}, Y_{t-L+1:t}) \\ &= P(x_t|\lambda_{t-L+1:t}, Y_{t-L+1:t}, x_{t-L}) \prod_{k=t-L+1}^{t-1} P(x_k|\lambda_{t-L+1:k}, Y_{t-L+1:k}, x_{t-L}, x_{k+1}) \end{aligned} \tag{5.28}$$

where

$$P(x_k|\lambda_{t-L+1:k}, Y_{t-L+1:k}, x_{t-L}, x_{k+1}) \propto P(x_{k+1}|x_k)P(x_k|\lambda_{t-L+1:k}, Y_{t-L+1:k}, x_{t-L}) \tag{5.29}$$

The distributions $P(x_k|\lambda_{t-L+1:k}, Y_{t-L+1:k}, x_{t-L})$ are given by a Kalman filter, and are Gaussian with mean μ_k and covariance Σ_k. Thus the complete state proposal is given by:

$$q(x_{t-L+1:t}|x_{t-L}, \lambda_{t-L+1:t}, Y_{t-L+1:t}) = \mathcal{N}(x_t|\mu_t, \Sigma_t) \prod_{k=t-L+1}^{t-1} \mathcal{N}(x_k|m_k, S_k) \tag{5.30}$$

where

$$S_k = [A^T Q^{-1} A + \Sigma^{-1}]^{-1} \tag{5.31}$$

$$m_k = S_k[A^T Q^{-1} x_{k+1} + \Sigma^{-1}\mu_k] \tag{5.32}$$

This method is equivalent to proposing states from a Kalman smoother estimate over the window.

6

Fixed Lag Particle Filters

6.1 Sequential Importance Sampling and Resampling Implementation

In this section we outline an SISR implementation of the fixed lag particle filter for target tracking. Using the augmented target distribution of equation 5.5 and an auxiliary form of history proposal as in equation 3.28, the importance weights can be written as

$$
\begin{aligned}
W_t^{(m)} &= \frac{P(X_{1:t-L}^{(m)}, X_{t-L+1:t}^{'(m)}|Y_{1:t})\rho(X_{t-L+1:t-1}^{(m)}|X_{1:t-L}^{(m)}, X_{t-L+1:t}^{'(m)})}{q(X_{1:t-1}^{(m)}|Y_{1:t-1})q(X_{t-L+1:t}^{'(m)}|X_{1:t-1}^{(m)}, Y_{t-L+1:t})} \\
&\approx \frac{W_{t-1}^{(m)}}{V_t^{(m)}} \times \frac{P(X_{1:t-L}^{(m)}, X_{t-L+1:t}^{'(m)}|Y_{1:t})\rho(X_{t-L+1:t-1}^{(m)}|X_{1:t-L}^{(m)}, X_{t-L+1:t}^{'(m)})}{P(X_{1:t-1}^{(m)}|Y_{1:t-1})q(X_{t-L+1:t}^{'(m)}|X_{1:t-1}^{(m)}, Y_{t-L+1:t})} \\
&\propto \frac{W_{t-1}^{(m)}}{V_t^{(m)}} \times \frac{P(Y_{t-L+1:t}|X_{t-L+1:t}^{'(m)})P(X_{t-L+1:t}^{'(m)}|X_{t-L}^{(m)})\rho(X_{t-L+1:t-1}^{(m)}|X_{1:t-L}^{(m)}, X_{t-L+1:t}^{'(m)})}{P(Y_{t-L+1:t-1}|X_{t-L+1:t-1}^{(m)})P(X_{t-L+1:t-1}^{(m)}|X_{t-L}^{(m)})q(X_{t-L+1:t}^{'(m)}|X_{1:t-1}^{(m)}, Y_{t-L+1:t})}
\end{aligned} \tag{6.1}
$$

Proposals are made using the method detailed in section 5.3.

The artifical conditional distribution, $\rho(.)$ should be an approximation for the conditional posterior, as discussed in section 5.2. This was the objective we had when constructing the proposal distribution, so we use the same approximation. i.e. $\rho(X_{t-L+1:t-1}|X_{1:t-L}, X'_{t-L+1:t}) = q(X_{t-L+1:t-1}|X_{t-L}, Y_{t-L+1:t-1})$.

6.1.1 Resampling strategies

In the basic particle filter, the resampling procedure is equivalent to proposing a set of histories, $X_{1:t-1}$, for the new particle distribution. Histories can only be taken from the previous step, otherwise there would be states missing. In a fixed lag particle filter, we could propose valid histories, X_{t-L} from any of the previous L frames. In certain difficult situations, where the correct path of the target is unclear, such a strategy could improve performance by increasing the diversity of histories which we can propose.

Directly proposing histories from old particle distributions would be difficult to implement in an SISR system (not so for MCMC - see section 2.1). Instead, we use a modified resampling scheme to achieve a similar effect. This scheme is based on that of (Godsill, Vermaak, Ng & Li 2007). Each particle is copied to form $n^{(m)}$ children:

$$n^{(m)} = \min(1, \left\lceil NW^{(m)} \right\rceil) \tag{6.2}$$

and and intermediate weight is assigned to each:

$$\tilde{W}^{(m)} = \frac{W^{(m)}}{n^{(m)}} \tag{6.3}$$

Finally, the particle set is systematically downsampled to reduced its size to N again. The effect of this procedure is to keep some low weight particles, improving the diversity of track histories. This "conservative resampling" scheme reduces track loss.

Such a resampling scheme is equivalent to auxiliary particle filtering with systmatic resampling and particle proposal weights given by

$$V_t^{(m)} = \frac{n^{(m)}}{\sum_m n^{(m)}} \tag{6.4}$$

6.1.2 Coping with dimensionality

A particle filter operating on the full joint posterior performs very poorly with more than a few targets. The reason for this is the very high dimensionality of the state space, which would require too many particles to adequately populate. In a particle with good estimates for one target, there many be poor estimates for another, resulting in a low weight. The probability of achieving a good estimate for all targets is very low if we propose them all at once.

The methods of (Vermaak et al. 2005) can be used to alleviate the dimensionality problem. In particular, if we relax the constraint that only one observation arises from each target in

each frame, as for the IPPF, then the PF can effectively be split into K_t separate filters, one for each target. The posterior for each individual target filter is given by taking the factor corresponding to that target from the observation likelihood, equation 4.6, the transition dynamics, equation 4.7, and the association likelihood, equation 4.9. In particular, the factorisation we use here for the association likelihood does not require a clutter weight to be applied accross all targets at the end in order to construct the joint posterior, as in (Vermaak et al. 2005).

Relaxing the unique association constraint is very mild assumption for well-spaced targets - they are highly likely to be actually independent in the posterior. However, for closely-spaced targets, an assumption of independence is poor, and is likely to lead to multiple estimated tracks following the same target, associated with the same observations, in a similar way to multiple PDAFs approximating a JPDAF. We can construct a hybrid between the full joint target system and the independent-association form. Initially we assume independence between all targets. After each processing frame, a test for "collisions" is carried out. A collision occurs when the sets of observations associated with two targets overlaps. When this occurs, we discard the estimates from the two independent particle filters and run a joint filter on the pair of targets. This clustering is maintained until the targets are well-spaced again, at which point we can revert to independent tracking.

Such a collision-detecting algorithm can only work well on problems when targets are reasonably sparse. If many targets pass close to each other in a short time, we will end up tracking them all jointly, and performance will revert to the that of the complete joint tracker. While two (or more) tracks are being jointly tracked the state estimation error is increased and the probability of losing a track increases.

6.2 Markov Chain Monte Carlo

The main failing of the SISR particle filter for tracking multiple targets is its inability to cope with high dimensionalities. This stems from the fact that a value for every state must be proposed for each particle at once, leading to high variance in the importance weights, poor particle diversity and ultimately poor tracking performance. An MCMC particle filter is able to cope with this problem more gracefully. Rather than propose all the states at once, we can break them down into groups and propose a change of only one group at a time, using the block sampling or "Metropolis-within-Gibbs" method.

Using the augmented target distribution of equation 5.5 and an auxiliary form of history proposal as in equation 3.28, the MH acceptance probability can be calculated. Again, we use $\{Z_{1:t-1}, Z'_{t-L+1:t}\}$ to represent the current state of the sampler and $\{X_{1:t-1}, X'_{t-L+1:t}\}$ for the

proposed state. Furthermore, we omit the association variables from the next expression for clarity. They should be considered as grouped with the corresponding state variables.

$$\alpha = \min\left(1, \frac{P(X_{1:t-L}, X'_{t-L+1:t}|Y_{1:t})\rho(X_{t-L+1:t-1}|X_{1:t-L}, X'_{t-L+1:t})q(Z_{1:t-1}, Z'_{t-L+1:t}|Y_{1:t}}{P(Z_{1:t-L}, Z'_{t-L+1:t}|Y_{1:t})\rho(Z_{t-L+1:t-1}|Z_{1:t-L}, Z'_{t-L+1:t})q(X_{1:t-1}, X'_{t-L+1:t}|Y_{1:t}}\right)$$
$$= \min\left(1, \frac{P(Y_{t-L+1:t}|X'_{t-L+1:t})P(X'_{t-L+1:t}|X_{t-L})}{P(Y_{t-L+1:t}|Z'_{t-L+1:t})P(Z'_{t-L+1:t}|Z_{t-L})} \times \frac{P(Y_{t-L+1:t-1}|Z_{t-L+1:t-1})P(Z_{t-L+1:t-1}|Z_{t-L})}{P(Y_{t-L+1:t-1}|X_{t-L+1:t-1})P(X_{t-L+1:t-1}|X_{t-L})}\right.$$
$$\left. \times \frac{\rho(X_{t-L+1:t-1}|X_{1:t-L}, X'_{t-L+1:t})}{\rho(Z_{t-L+1:t-1}|Z_{1:t-L}, Z'_{t-L+1:t})} \times \frac{q(Z_{t-L+1:t}|Y_{t-L+1:t}, Z_{t-L})}{q(X_{t-L+1:t}|Y_{t-L+1:t}, X_{t-L})} \times \frac{V_{Z,t}}{V_{X,t}}\right) \quad (6.5)$$

As for the SISR particle filter, we can use the appoximations developed in section 5.2 for the artificial conditional distribution. The same form may be used for the proposal distribution.

As suggested, we can have MH moves which change only some subset of the states. For example, we can have moves which change only the current states, $X'_{t-L+1:t}$, and others which change the history, $X_{1:t-1}$. For the current states, an obvious choice here is to change only one target at a time. For well-spaced targets, this is almost equivalent to running a separate particle filter on each, the moves concerning one target will have no effect on any other. When targets are closely spaced, there is a disadvantage to using only single-target moves. If two targets share two likely candidate observations, it will be hard for them to "swap". Thus we should also include a number of two-target moves in our MCMC scheme, using pairs of targets identified as being close together.

The difficulty for an MCMC scheme is implementing the history moves. A history move is the selection of a particle from the previous frame posterior approximation. Thus, the states of all targets will be changed. For large numbers of targets, this can lead to very low acceptance probabilities. At this point there is little choice but to assume the target histories are independent and approximate the history particle distribution as a product of marginal distributions. We can now propose changes in the target histories independently. Moreover, this assumption is really quite mild - it can lead to violations of the constraint on a single target per observation, but only in the historical states, $X_{1:t-L}$ (or the discarded states, $X_{t-L+1:t-1}$, but this is inconsequential). Merging tracks, the usual problem with relaxing this constraint, are prevented because the constraint is still enforced in the current states, $X_{t-L+1:t}$.

An additional problem with the history moves, even with an independence assumption, is that they can have low acceptance probabilities because of the tendency to introduce tracks which are disjointed between $t-L$ and $t-L+1$. This can be mitigated by simultaneously

proposing new states in some bridging region, $t-L+1:t-L+b$, where b is 1 or 2. These states give us much better acceptance rates than changing just histories alone. NUMBERS.

6.2.1 Proposing older histories

So far we have used the $t-1$ particle distribution from which to propose histories, $X_{1:t-L}$ for the Markov chain. However, a suitable history could be also selected from the $t-s$ distribution, for $1 \leq s \leq L$. In a situation where a target has not been detected for several consecutive frames, taking a history from further in the past may make the proposal of a good candidate more probable. The correct path will have been more probable before the frames in which the target was not detected. Such a scheme has a aim as the conservative resampling strategy employed for the SISR FLPF - to maintain the track on a target which temporarily follows a low probability path.

The history proposal lag variable s can most easily be handled by sampling it too.

$$\{X_{1:t-s}, X'_{t-L+1:t}, s\} \sim q(s)q(X_{1:t-s}|Y_{1:t-s})q(X'_{t-L+1:t}|X_{1:t-s}, Y_{t-L+1:t}) \tag{6.6}$$

We only use s in the proposal mechanism, so it can simply be discarded (marginalised) once the chain has moved on. In addition, if $q(s)$ is uniform over all possible values then it will cancel out in the acceptance probabilities. The only change to the proposal then is that we have an enlarged set of particles from which to select a history. The acceptance probability is now given by:

$$\begin{aligned}\alpha &= \min\left(1, \frac{P(X_{1:t-L}, X'_{t-L+1:t}|Y_{1:t})\rho(X_{t-L+1:t-s_X}|X_{1:t-L}, X'_{t-L+1:t})q(Z_{1:t-s_Z}, Z'_{t-L+1:t}|Y_{1:t}}{P(Z_{1:t-L}, Z'_{t-L+1:t}|Y_{1:t})\rho(Z_{t-L+1:t-s_Z}|Z_{1:t-L}, Z'_{t-L+1:t})q(X_{1:t-s_X}, X'_{t-L+1:t}|Y_{1:t}}\right) \\ &= \min\left(1, \frac{P(Y_{t-L+1:t}|X'_{t-L+1:t})P(X'_{t-L+1:t}|X_{t-L})}{P(Y_{t-L+1:t}|Z'_{t-L+1:t})P(Z'_{t-L+1:t}|Z_{t-L})} \times \frac{P(Y_{t-L+1:t-s_Z}|Z_{t-L+1:t-s_Z})P(Z_{t-L+1:t-s_Z}|Z_{t-L})}{P(Y_{t-L+1:t-s_X}|X_{t-L+1:t-s_X})P(X_{t-L+1:t-s_X}|X_{t-L})}\right. \\ &\quad \left. \times \frac{\rho(X_{t-L+1:t-s_X}|X_{1:t-L}, X'_{t-L+1:t})}{\rho(Z_{t-L+1:t-s_Z}|Z_{1:t-L}, Z'_{t-L+1:t})} \times \frac{q(Z_{t-L+1:t}|Y_{t-L+1:t}, Z_{t-L})}{q(X_{t-L+1:t}|Y_{t-L+1:t}, X_{t-L})} \times \frac{V_{Z,t}}{V_{X,t}}\right)\end{aligned} \tag{6.7}$$

This scheme will be shown to give improved tracking performance.

6.3 Marginalised Particle Filters

If the dynamics of our tracking problem are linear-Gaussian, then using a particle filter for the state estimation is like using a sledgehammer to crack a nut. Similarly, if part of the state is

linear-Gaussian *conditional* on the rest, then the Rao-Blackwellisation method may be used. A marginalised PF (MPF) is used for estimating the nonlinear state partition, and a Kalman filter is used for the rest. Target tracking problems are ideally suited to MPFs. The association variables are estimated with a PF, after which the kinematic state is estimated by a Kalman filter. If the target dynamics are only mildly nonlinear, then the combination of an EKF and a MPF may well outperform a simple PF, on account of the reduced dimensionality of the particle distribution.

The task of our MPF is to estimate the posterior of the associations, independent of the states, i.e. $P(\Lambda_{1:t}|Y_{1:t})$. A Kalman filter is then used to estimate the state distribution of each target conditional on the associations. The joint posterior is then constructed as

$$P(X_{1:t}, \Lambda_{1:t}|Y_{1:t}) = P(\Lambda_{1:t}|Y_{1:t}) \prod_{j=1}^{K_t} P(x_{j,1:t}|\Lambda_{1:t}, Y_{1:t}) \tag{6.8}$$

The only change to the particle filter is the use of the marginal posterior, which expands as:

$$P(\Lambda_{1:t}|Y_{1:t}) = \frac{P(Y_{t-L+1:t}|\Lambda_{1:t}, Y_{1:t-L})P(\lambda_{t-L+1:t})P(\Lambda_{1:t-L}|Y_{1:t-L})}{P(Y_{t-L+1:t}|Y_{1:t-L})} \tag{6.9}$$

The predictive likelihood terms is given by:

$$\begin{aligned} P(Y_{t-L+1:t}|\Lambda_{1:t}, Y_{1:t-L}) &= \prod_{k=t-L+1}^{t} P(Y_k|Y_{1:k-1}, \Lambda_{1:k}) \\ &\propto \prod_{k=t-L+1}^{t} \prod_{j=1}^{K_k} P(y_k^{(\lambda_{j,k})}|\Lambda_{1:k-1}, Y_{1:k-1}) \end{aligned} \tag{6.10}$$

The factors may be calculated using our assumption of Gaussian target dynamics. Provided $\lambda_{j,k} \neq 0$,

$$P(y_k^{(\lambda_{j,k})}|\Lambda_{1:k-1}, Y_{1:k-1}) = \int P(y_k^{(\lambda_{j,k})}|x_{j,k}, \Lambda_{1:k-1}, Y_{1:k-1})P(x_{j,k}|\Lambda_{1:k-1}, Y_{1:k-1})dx_{j,k} \tag{6.11}$$

The term $P(x_{j,k}|\Lambda_{1:k-1}, Y_{1:k-1})$ is familiar as the Kalman filter prediction of the target state. Hence we have:

$$P(y_k^{(\lambda_{j,k})}|\Lambda_{1:k-1}, Y_{1:k-1}) = \begin{cases} \mathcal{N}(y_k^{(\lambda_{j,k})}|C\hat{\mu}_k, R + C\hat{\Sigma}_k C^T) & \lambda_{j,k} \neq 0 \\ V^{-1} & \lambda_{j,k} = 0 \end{cases} \tag{6.12}$$

where $\hat{\mu}_k$ and $\hat{\Sigma}_k$ are the Kalman filter prediction mean and covariance respectively.

In all other ways, including association proposals, history proposals, resampling, etc., the MPF is unchanged. It may be implemented using either SISR or MCMC.

7

Particle Filtering for Target Detection

7.1 Changes to the model

In order to handle targets appearing and disappearing from the scene, we introduce an existence variable for each possible target, $e_{j,t}$, the set of which we denote E_t. These boolean variables indicate whether a target is present at a given time instant, taking values 1 (present) or 0 (absent). We limit the number of targets to $K_{\max}$.

We would like to infer the values of E_t, so this may be added to the posterior thus:

$$P(X_{1:t}, \Lambda_{1:t}, E_{1:t}|Y_{1:t}) = \frac{P(Y_t|X_t, \Lambda_t, E_t)P(X_t|X_{t-1}, E_t, E_{t-1})P(\Lambda_t|E_t)P(E_t|E_{t-1})P(X_{1:t-1}, \Lambda_{1:t-1}, E_{1:t-1}|Y_{1:t-1}}{P(Y_t|Y_{1:t-1})} \tag{7.1}$$

The modified graphical model is shown in figure 7.1.

Consider each of these terms. The form of the likelihood changes little.

$$\begin{aligned} P(Y_t|X_t, \Lambda_t, E_t) &= V^{-(M_t - M_{T,t})} \prod_{\lambda_{j,t} \neq 0} P(y_t^{(\lambda_{j,t})}|x_{j,t}) \\ &= V^{-(M_t - K_{\max})} \prod_{j=1}^{K_{\max}} \begin{cases} P(y_t^{(\lambda_{j,t})}|x_{j,t}) & e_{j,t} = 1, \lambda_{j,t} \neq 0 \\ V^{-1} & e_{j,t} = 0 \text{ or } \lambda_{j,t} = 0 \end{cases} \end{aligned} \tag{7.2}$$

The association likelihood becomes:

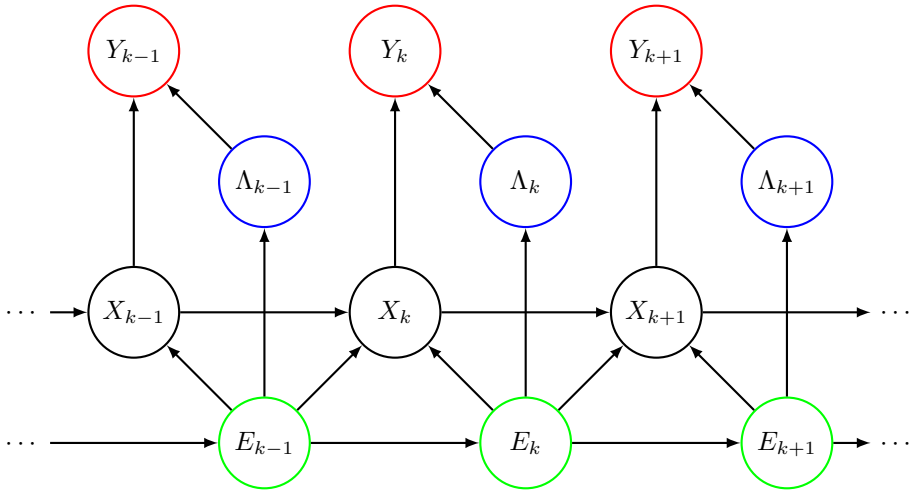


Figure 7.1: Graphical model for the modified target tracking model including target existence variables.

$$P(\lambda_t|E_t) = \frac{\exp(-\mu_C)\mu_C^{(M_t-K_{\max})}}{M_t!} \prod_{j=1}^{K_{\max}} \begin{cases} P_D & e_{j,t} = 1, \lambda_{j,t} \neq 0, \lambda_{j,t} \notin \{\lambda_{1:j-1,t}\} \\ 0 & e_{j,t} = 1, \lambda_{j,t} \in \{\lambda_{1:j-1,t}\} \\ (1-P_D)\mu_C & e_{j,t} = 1, \lambda_{j,t} = 0 \\ \mu_C & e_{j,t} = 0 \end{cases} \tag{7.3}$$

The transition density becomes more complex. When targets appear we assume that their kinematic states are distributed according to some birth distribution $P_b(x_{j,t}$, known a priori. This may be uniform. Before targets have appeared and after they have died we set their state to some fixed dead-state. Thus we have

$$P(X_t|X_{t-1}, E_t, E_{t-1}) = \prod_{j=1}^{K_{\max}} \begin{cases} P(x_{j,t}|x_{j-1}) & e_{j,t} = 1, e_{j,t-1} = 1 \\ P_b(x_{j,t}) & e_{j,t} = 1, e_{j,t-1} = 0 \\ \delta_{x_{\text{dead}}}(x_{j,t}) & e_{j,t} = 0 \end{cases} \tag{7.4}$$

Finally we need an expression for the existence transitions. We assume that targets have a fixed, independent probability of disappearing at any time, and that new targets are generated according to a Poisson process. Thus

$$P(E_t|E_{t-1}) = \frac{\exp(\mu_{\text{birth}})\mu_{\text{birth}}^{K_{\text{new},t}}}{K_{\text{new},t}} \prod_{j=1}^{K_{\max}} \begin{cases} 1 - P_{\text{death}} & e_{j,t} = 1, e_{j,t-1} = 1 \\ P_{\text{death}} & e_{j,t} = 0, e_{j,t-1} = 1 \end{cases} \tag{7.5}$$

where $K_{\text{new},t}$ is the number of targets present in frame t which were not present in frame $t-1$.

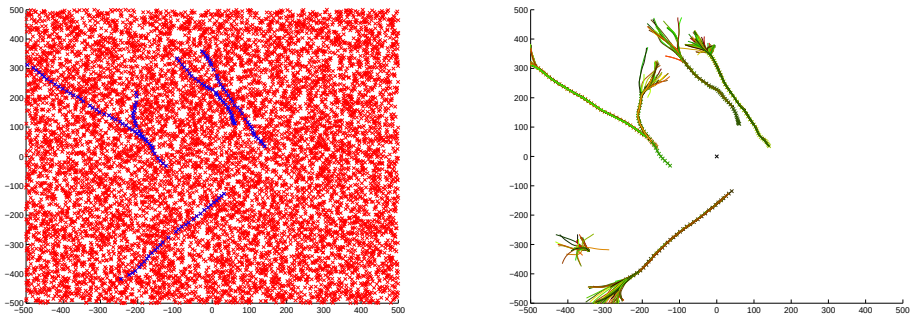
7.2 Particle filtering

A particle filter for joint target detection and state estimation was implemented using a fixed lag SISR particle filter. Target disappearance requires only a trivial change, allowing the proposal that a track should end with some small probability.

Target appearance is a greater challenge, as the dimensionality of the state space expands enormously (any number of targets can appear in any location). In order to devise a practical algorithm, we use the method of a 'search track' of (Horridge & Maskell 2009). The SISR particle filter already requires that we assume independence between targets. We assume that μ_{birth} is low enough that the probability of more than 1 track appearing at a given time is negligible. Then, we simply need one extra track whose particles represent the associations and states of a potential new target. When these search particles are observed to cluster in one location, we conclude that a new target has appeared and a new target filter is initiated.

A new mechanism is be required for proposing the observation associations for the search track particles, as we have no knowledge about the target position or velocity. Once the associations are proposed, the usual method for state proposals may be used. In even moderate clutter some scheme is needed to identify likely sites where a new target may have appeared, for example, by locating strings of observations in close proximity over time. These sites can be used to propose search particle associations. Without such a heuristic frontend, the probability of putting particles in the right place is likely to be very low.

An example of the SISR algorithm with lag 5 tracking 5 targets is shown in figure 7.2, using the linear-Gaussian observation model. Process and observation noises are set to 1, detection probability is 0.9 and expected number of clutter observations is 200 per frame.



(a) Observations (b) States and Particle Estimates

Figure 7.2: Target detection and tracking with fixed lag SISR algorithm with lag 5.

8

Results

8.1 Testing methods

A number of algorithms were compared on a set of standard tests to compare the performance:

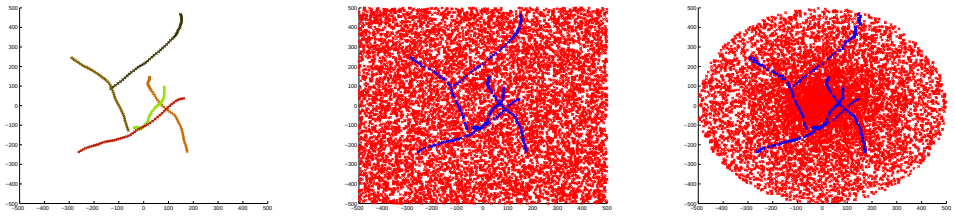
- SISR
- SISR FL(5)
- RB-SISR FL(5)
- MCMC
- MCMC FL(5)
- RB-MCMC FL(5)
- PDAF
- JPDAF

where FL(N) indicates fixed lag with a window length of N. The test set consisted of:

1. 5 widely-spaced targets
2. 5 widely-spaced targets, high clutter and missed detection rate
3. 5 widely-spaced targets, high observation noise covariance
4. 5 closely-spaced targets

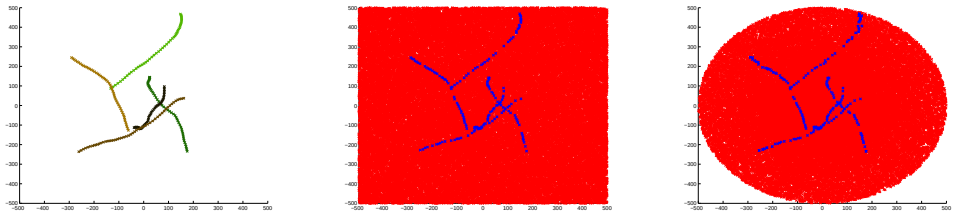
Each scenario lasted for 50 time steps. Each was tested using both the linear and bearing-range observation models. Each combination was run with 10 different random seeds. An example of each is shown in figures 8.1 to 8.4.

Each scene is 1000×1000. The velocity of the targets is limited to 10. The period between time steps is 1. The case-specific parameters used are shown in table 8.1



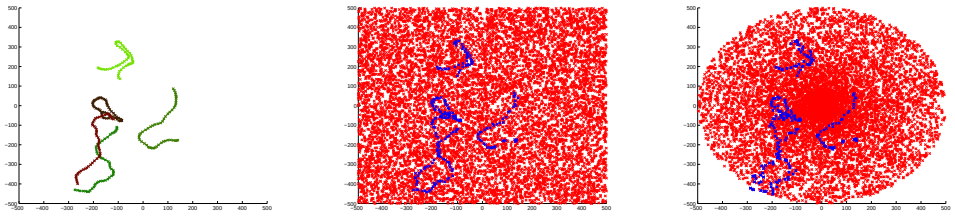
(a) state (b) linear observations (c) bearing-range observations

Figure 8.1: Case 1



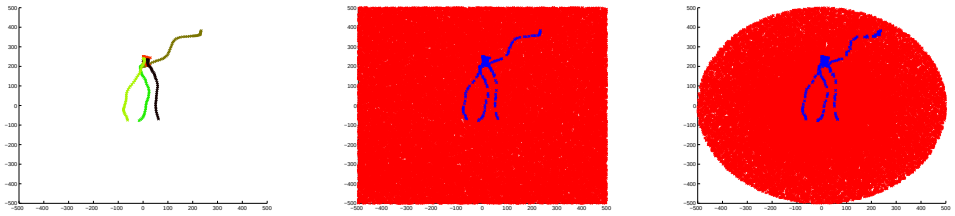
(a) state (b) linear observations (c) bearing-range observations

Figure 8.2: Case 2



(a) state (b) linear observations (c) bearing-range observations

Figure 8.3: Case 3



(a) state (b) linear observations (c) bearing-range observations

Figure 8.4: Case 4

	case 1	case 2	case 3	case 4
process variance	1	1	10	1
observation variance (linear)	1	1	10	1
bearing variance (nonlinear)	10^{-4}	10^{-4}	10^{-3}	10^{-4}
range variance (nonlinear)	1	1	10	1
expected no. clutter (μ_C)	200	1000	200	1000
detection probability (P_D)	0.9	0.75	0.9	0.75

Table 8.1:

The SISR algorithm was run with 500 particles per target. The MCMC algorithm used 5000 iterations and updated the targets in a deterministic order. All algorithms employed gating at 5 standard deviations using a normal approximation.

8.2 Performance assessment

Performance was assessed via a number of statistics. Firstly the number of lost tracks was counted. For the particle filter algorithms these were identified using the associations. When no particle was associated with the correct observation for five frames, the target was considered lost. For non-particle algorithms (PDAF and JPDAF), targets were considered lost when the mean state estimate was greater than a certain threshold distance from the correct position.

After excluding lost tracks, the RMS state error was measured. For the particle algorithms a state estimate was obtained by taking the mean of the state of all particles. For non-particle algorithms, the mean of the Gaussian representing the state distribution was used. For particle algorithms, the proportion of particles with the correct association was also calculated. These statistics were all measured at a fixed lag of five steps from the current time, for consistency of comparison.

Results are shown in table2.1. Blank cells indicate that the tracker failed to complete the test. With the JPDAF this occured when the number of association hypotheses grew too large.

Figure 2.1 to 2.1 show the comparative performance of the algorithms on a few example cases.

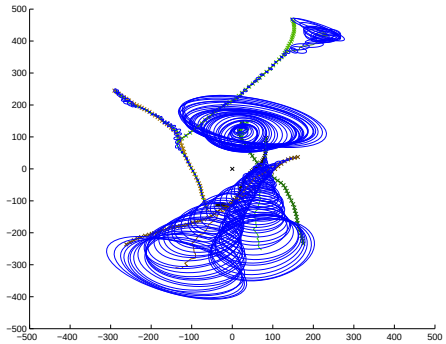
These results highlight the characteristics of the fixed lag particle filters. In almost all cases the particle algorithms outperform the basic PDAF tracker in terms of number of tracks lost and RMSE. In all cases using a longer window results in better tracking performance. Tracking performance is always better with the linear observation model rather than the bearing-

		Algorithm			
		Case 1	Case 2	Case 3	Case 4
PDAF	lost tracks	32%	4%	56%	50%
	RMSE	4.00	1.57	5.93	4.12
JPDAF	lost tracks		4%	48%	
	RMSE		1.57	5.89	
SISR(1)	lost tracks	32%	2%	100%	36%
	RMSE	5.53	1.39		7.46
	correct associations	72%	93%		63%
SISR(5)	lost tracks	2%	0%	42%	16%
	RMSE	2.02	0.95	50.12	2.01
	correct associations	85%	93%	71%	79%
RB-SISR(5)	lost tracks	4%	0%	12%	12%
	RMSE	1.95	1.36	9.01	2.90
	correct associations	86%	93%	86%	83%
MCMC(1)	lost tracks	30%	2%	100%	38%
	RMSE	2.24	1.04		1.74
	correct associations	87%	99%		81%
MCMC(5)	lost tracks	26%	0%	100%	26%
	RMSE	1.44	0.98		1.47
	correct associations	87%	99%		81%
RB-MCMC(5)	lost tracks	6%	0%	10%	12%
	RMSE	2.31	1.33	3.35	1.81
	correct associations	91%	99%	90%	90%

Table 8.2: Algorithm test performance measures for linear observation model

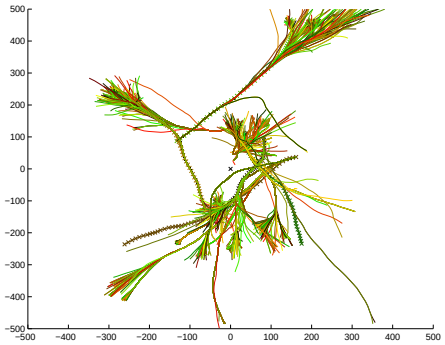
		Algorithm			
		Case 1	Case 2	Case 3	Case 4
PDAF	lost tracks	60%	8%	80%	72%
	RMSE	4.89	2.96	10.15	5.98
JPDAF	lost tracks		6%		
	RMSE		2.95		
SISR(1)	lost tracks	40%	10%	100%	50%
	RMSE	9.86	2.54		12.93
	correct associations	57%	86%		50%
SISR(5)	lost tracks	16%	4%	82%	22%
	RMSE	4.51	1.54	67.9	5.91
	correct associations	71%	90%	55%	69%
RB-SISR(5)	lost tracks	18%	6%	30%	26%
	RMSE	4.21	1.86	16.91	4.99
	correct associations	71%	89%	68%	72%
MCMC(1)	lost tracks	30%	2%	100%	52%
	RMSE	4.32	1.73		3.54
	correct associations	75%	98%		68%
MCMC(5)	lost tracks	20%	4%	100%	44%
	RMSE	3.21	1.52		3.11
	correct associations	79%	97%		71%
RB-MCMC(5)	lost tracks	22%	2%	48%	36%
	RMSE	3.36	1.78	5.96	3.18
	correct associations	80%	96%	65%	74%

Table 8.3: Algorithm test performance measures for bearing-range observation model

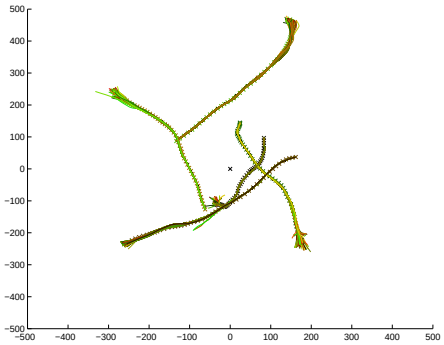


(a) PDAF

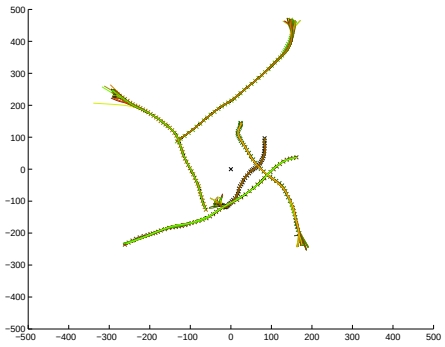
Figure 8.5: PDAF tracking results. Ellipses mark 1 standard deviation from the mean from the Gaussian covariance estimate.



(a) SISR(1)

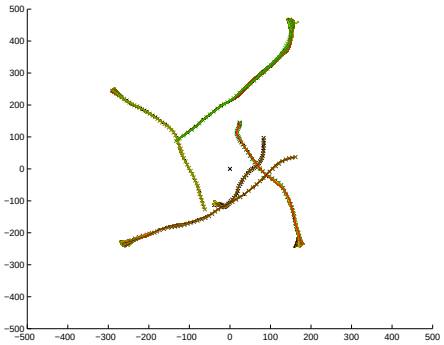


(b) SISR(5)

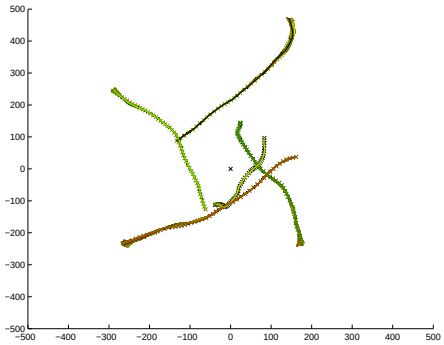


(c) RB-SISR(5)

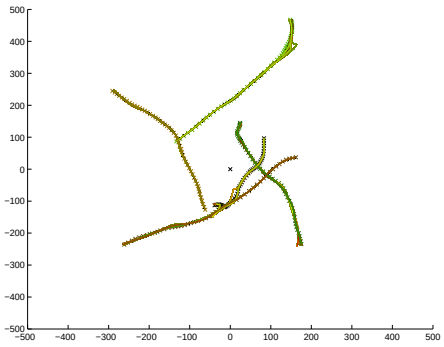
Figure 8.6: SISR tracking results. All particles plotted.



(a) MCMC(1)



(b) MCMC(5)



(c) RB-MCMC(5)

Figure 8.7: MCMC tracking results. All particles plotted.

range model. For the linear model, no Gaussian approximations are needed and the proposal distributions are optimal.

For the linear observation model, the use of Rao-Blackwellisation seems to give a reduction in the number of tracks lost, and makes little difference to the RMSE. This is the expected result. The Kalman filters used for state estimation are optimal for this model, and the reduction in dimensionality means that low probability paths are better characterised, resulting in fewer track losses. With the bearing-range observation model, the reduction in lost tracks from Rao-Blackwellisation is less. This could be because the Kalman filter estimates are no longer optimal, resulting in worse state estimates and a greater tendency to follow an incorrect path. However, this hypothesis is not born out in the RMSE results, which are not significantly worse than the full particle filter.

For case 3, with high process and observation noise, the full particle filters do not work at all, while the Rao-Blackwellised forms give good tracking performance. This is because of the number of particles used is insufficient to characterise the state distribution well. The probability of getting a 'good' particle becomes low, so tracks are lost very quickly. The marginalised particle filter does not need to characterise the state distribution, so the increased variance is not so significant.

On the widely spaced targets, the SISR-PFs suffer from fewer lost tracks than the MCMC-PFs. This appears to be caused by two related effects. The SISR particles are generated independently, and with the conservative resampling scheme employed a number of particles will be maintained with low weights on low probability paths. If these paths turn out to be correct, the tracker will not lose the target. The MCMC particles fail to maintain a particle contribution on such low probability paths. This may be because the particles are generated sequentially, each based upon the previous. Moves to low probability paths simply never get accepted. This problem can be viewed in an alternative light. After a few additional time steps, a low probability path which is actually correct will increase again in probability. We would like the MCMC sampler to then be able to jump across to this path. However, to do so will generally require a move in which all the states in the window (from the point at which the potential path bifurcated until the present) change significantly. Such a high dimensional move will have a very low acceptance probability, and the chains used may simply not be long enough.

Disappointingly, the MCMC-PFs do not seem to give an improvement in track loss over the SISR-PFs for the closely-spaced targets. Seeing as the MCMC-based tracker maintains a full joint distribution, it might be expected to suffer from fewer merged or crossing-over tracks. However, the difficulties outlined above seem to outweigh this potential advantage.

The particle based algorithms are of the order of 1000 times slower than the PDAF, i.e. the same order as the number of particles used. The computation time scales linearly with the length of the window. The scaling with the number of targets depends on how they interact with each other. Targets which are essentially independent (i.e. well-spaced), give rise to a linear scaling. For closely-spaced targets, the number of particles will have to increase more than linearly with the number of targets to maintain a constant level of tracking performance. The use of Rao-Blackwellisation gives the potential for a significant speed-up because many more calculations will be repeated between particles. However, this potential was not exploited in this work. The JPDAF is, while all targets remain in track, almost as fast as the PDAF. However, with a naive implementation, such as that used for these tests, the computation time explodes when a track is lost (due to the widening variance of the estimated state resulting in an increasing number of observations lying in each gate). More intelligent approaches can avoid such problems, e.g. (Horridge & Maskell 2006).

9

Cursor Intentionality Tracking

As an aside from the primary work on fixed lag particle filtering for tracking, a short study has been made on inferring the intended destimation of a computer mouse cursor, as a component for aiding disabled computer user. A number of algorithms have been implemented and tested on a set of logged data.

9.1 Algorithms

The input to the algorithm is a series of logged cursor positions. Measurements are received asynchronously and consist of a time stamp and (x,y) coordinates. In addition, markers indicate when a click occurs.

Datasets have been logged in which a sequence of buttons appear on the screen, and users are instructed to click on them as quickly as possible. The purpose of the algorithm is to infer which button the users are intending to click on at each point in time. In the tests, only one button is ever present at once (making the inference task rather easy!), so a grid of phantom buttons is introduced, tiled over the screen. Models are devised which will allow calculation of a likelihood for each of the buttons in this grid. Thus a maximum likelihood (ML) or maximum a posteriori (MAP) choice can be made. The performance of the algorithms can be assessed by comparing the proportion of time during which the correct button is chosen.

9.1.1 Nearest Neighbour

The simplest conceivable algorithm for selecting a button is to pick that which is closest to the cursor at the given time instant. This most basic of schemes will be used as a benchmark

against which to compare others. Such a scheme has a probabilistic interpretation, that the cursor position is normally distributed with mean at the target button and a fixed variance. This interpretation can be useful for calculating probabilities for each button, rather than just finding a ML or MAP choice.

$$P(x_k|b_i) = \mathcal{N}(x_k|b_i, \sigma^2 I) \tag{9.1}$$

where x_k is the k^{th} measurement of the cursor position, b_i is the i^{th} button, I the 2×2 identity matrix and σ the standard deviation, a design parameter.

If we consider that the cursor position at every measurement is independent (clearly not true, but we are aiming for simplicity), then we can find the posterior probability of each button given some set of the measurements.

$$P(b_i|x_{1:k}) = \frac{P(b_i)\prod_{j=1}^{k} P(x_j|b_i)}{P(x_{1:k})} \tag{9.2}$$

We assume a uniform prior over the buttons, although more sophisticated choices could be used if prior knowledge about the probaility of each button is known. The MAP choice of button is now that with the least mean square distance to the cursor over the measurements $x_{1:k}$, which accords with intuition. We need not necessarily use all the measurements for this calculation, but only those which lie in a window of fixed time length before the current time. The length of this window, L, is another design parameter.

Summary

Pick the button which is closest to the cursor (on average). Design parameters: Window length, L.

9.1.2 Mean-Reverting Diffusion

The cursor movement can be modeled as an Ornstein-Uhlenbeck process, a diffusion with a mean-reversion term. The cursor velocity is assumed to have two components, one random (Gaussian) and one deterministic which moves it towards the correct button.

$$dx_t = \lambda(b_i - x_t)dt + \sigma dw_t \tag{9.3}$$

where w_t is a Wiener process.

This may be solved analytically and discretised to give:

$$P(x_{k+1}|x_k, b_i) = \mathcal{N}(x_{k+1}|\mu_k, \gamma_k I) \tag{9.4}$$

where x_k is the k^{th}, measurement made at time t_k, and:

$$\mu_k = x_k + (b_i - x_k)(1 - \exp(-\lambda(t_{k+1} - t_k))) \tag{9.5}$$

$$\gamma_k = \sigma^2\left(\frac{1 - \exp(-2\lambda(t_{k+1} - t_k))}{2\lambda}\right) \tag{9.6}$$

σ and λ are design parameters.

As before, we can calculate the posterior probability given a set of measurements.

$$P(b_i|x_{1:k}) = \frac{P(b_i)P(x_1|b_i)\prod_{j=2}^{k} P(x_j|x_{j-1}, b_i)}{P(x_{1:k})} \tag{9.7}$$

We assume a uniform prior over the buttons, and the starting point, x_1. The posterior for each button is thus proportional to the product of the transition probabilities given that button. As before, the length of the window used must be selected.

This algorithm has the advantage that when the cursor stops moving, the ML choice of button is that closest to the cursor, i.e. it reverts to a nearest neighbour scheme.

Summary

Pick the button towards which the cursor is drifting. Design parameters: Window length, L. Diffusion noise variance, σ^2. Drift coefficient, λ.

9.1.3 Mean Bearing

We can assume that the bearing moved along by the cursor between each consecutive pair of measurements is a random variable with a mean equal to the bearing of the target button.

$$P(x_{k+1}|x_k) = \mathcal{N}(B_{x_k}(x_{k+1})|B_{x_k}(b_i), \sigma) \tag{9.8}$$

where $B_a(b)$ is the function which returns the bearing of b from a. σ is a design parameter.

As before, we can look at a set of measurements in a window as well as just one.

Summary

Pick the button at which the path of the cursor is pointing. Design parameters: Window length, L. Bearing noise variance, σ^2.

9.1.4 Composite

The bearing model works particularly poorly when the cursor is moving slowly, as it most often is, and thus does not have a clearly defined direction of travel. A composite algorithm has been implemented which uses the bearing model for high speeds and the mean-reverting diffusion model for low speeds.

Design parameters: Window length, L. Bearing noise variance, σ_B^2. Diffusion noise variance, σ_D^2. Drift coefficient, λ. Domain switching speed threshold, S_T.

9.2 Results

The algorithms were tested by running them on three data sets, those of subjects 1 (able-bodied), 14 (slightly impaired) and 21 (significantly impaired). As each measurement arrives, an estimate of the MAP button is made. We measure the proportion of time over which the correct button is chosen.

For three algorithms, the nearest neighbour (NN), mean-reverting diffusion (MRD) and mean bearing (MB), the design parameters were optimised by a simple grid method. The variance parameters do not affect the MAP choice of button, so the optimisation was only over 1 or 2 parameters, making is feasible.

For MRD, the best choice of diffusion coefficient was found to be $\lambda = 0.011 - 0.012$ for subjects 1 and 14 with a window length $L = 0s$ (i.e. using only the latest measurement), and $\lambda = 0.004$ with $L = 0.9s$ for subject 21.

For the NN algorithm, the optimum choice of window length was $L = 0s$ for subjects 1 and 14, $L = 0.5s$ for subject 21.

For the MB algorithm, the optimum choice of window length was $L = 0.4s$ for subject 1, $L = 0.5s$ for subject 14, $L = 1.2s$ for subject 21.

For the Composite (Comp) algorithm, the parameters were set to the optimum values for each of the constituent algorithms. The threshold used was $S_T = 500\text{pixel}/s$.

The best performance is obtained by the MRD algorithm, although the improvement over the NN algorithm is negligible for the impaired user, and is probably not worth the additional computation.

The MB algorithm performs very poorly according to the metric used. However, examining the posterior probabilities generally shows that the algorithm often identifies a likely cluster of targets in a wedge shape in front of the cursor. The correct target is often a member of the cluster, but is rarely picked out as the MAP choice. The videos of the algorithm in action

	Proportion Correct by Algorithm (%)			
Subject	NN	MRD	MB	Comp
1	44.4	49.4	14.9	42.4
14	41.7	43.4	11.5	40.7
21	55.4	55.9	7.4	54.9

Table 9.1: Table showing percentage of time for which each algorithm picks the correct button, with optimised parameters.

demonstrate this well. A bearing-based approach may therefore have some use in combination with some other strategy.

Figure 9.1 shows a screenshot of a tracking algorithm in action. The circles represent the grid of phantom buttons. The real button is shown with a red circle. The shading of the circles represents the posterior probability of the button. The blue line is a trace of the cursor movement. The red cross indicates the button selected by the algorithm.

9.3 Future work

The next stage required for a practical system is to introduce a method for deciding when the selected button is that which the user would like to click on. This could be decided by waiting for the MAP selection to settle on a particular choice for a given length of time, or by waiting for the MAP probability to exceed a certain threshold.

Improved tracking algorithms may be possible using jump-diffusion models, in which the cursor is assumed to move smoothly in between a number of discrete jump times, at which point a large change in velocity may occur. Such models may require particle filter implementations.

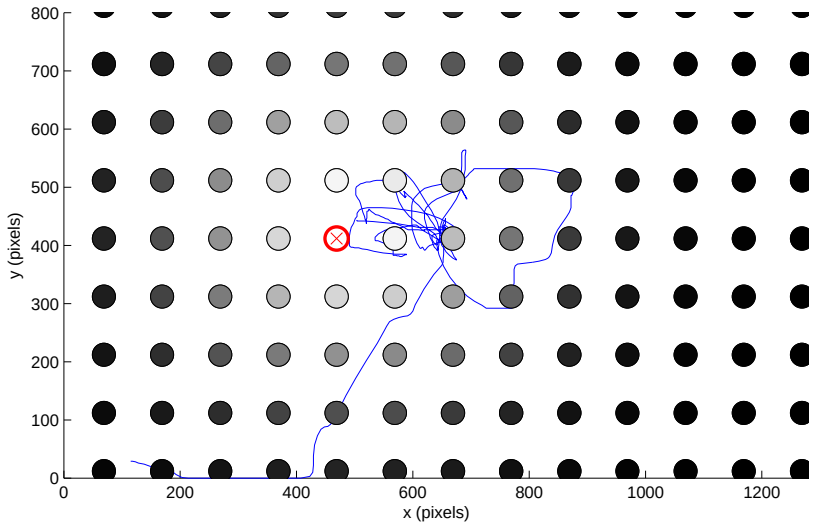


Figure 9.1: A screen shot showing the cursor trace, the grid of phantom buttons and their probabilities. See text for details.

10

Plan Of Future Research

A number of possible extensions to the current work are envisaged. Firstly, there are a number of variants of the particle filtering algorithms which would be interesting to investigate. Secondly, there are numerous additional problems in tracking which could be addressed. Finally, the fixed lag particle filter approach could be applied to different applications entirely.

10.1 Ideas for improved particle filters

10.1.1 Mixed exact-flow filter

In a traditional particle filter, particles are propagated from one time step to the next through the use of a proposal distribution and a weighting step, with resampling to maintain diversity. In (Daum, Huang & Noushin 2010), the authors introduce a new sort of particle filter, in which particles are propagated as if they were diffusing according a differential equation governed by the observation likelihood. Thus, it is possible to generate an exact, unweighted sample from the desired posterior distribution even for a nonlinear, non-Gaussian model. This method cannot be directly applied to a target tracking problem because such particle flow ideas can only be used for continuous state-spaces. The association variables will need to be handled differently.

It seems intuitive that we may be able to use an exact-flow filter for the kinematic target states, X_t, and an ordinary particle filter for the associations, Λ_t, in a similar manner to the Rao-Blackwellised method where a Kalman filter or filters is used for the kinematic states.

Consider a single target with state x_t and associations λ_t. From the previous processing step we have a particle approximation for the posterior, $\hat{P}(x_{1:t-1}, \lambda_{1:t-1}|Y_{1:t-1})$. The exact-flow

method allows us to generate a sample from $P(x_t|x_{t-1}, \lambda_t, Y_t)$. Thus, a new particle set can be generated from the following proposal

$$x_{1:t}^{(m)}, \lambda_{1:t}^{(m)} \sim \hat{P}(x_{1:t-1}, \lambda_{1:t-1}|Y_{1:t-1}) q(\lambda_t|X_{t-1}, Y_t) P(x_t|x_{t-1}, \lambda_t, Y_t) \tag{10.1}$$

The same association proposal may be used as for the standard particle filters described in chapter 2.1. Note that this proposal implicity includes a resampling step, which could be omited, as always. The importance weights for such a filter will be given by

$$W_t^{(m)} \approx \frac{P(x_{1:t}^{(m)}, \lambda_{1:t}^{(m)}|Y_{1:t})}{P(x_{1:t-1}, \lambda_{1:t-1}|Y_{1:t-1}) q(\lambda_t|X_{t-1}, Y_t) P(x_t|Y_{1:t}, \lambda_t)} \tag{10.2}$$

$$= \frac{P(\lambda_t^{(m)}) P(Y_t|x_{t-1}^{(m)}, \lambda_t^{(m)})}{P(Y_t|Y_{1:t-1}) q(\lambda_t|X_{t-1}, Y_t)} \tag{10.3}$$

Unfortuneately, the $P(Y_t|x_{t-1}^{(m)}, \lambda_t^{(m)})$ term will not be tractable in general. However, a Monte Carlo approximation could be used.

$$P(Y_t|x_{t-1}, \lambda_t) = \int P(Y_t|x_t, \lambda_t) P(x_t|x_{t-1}) dx_t \approx \sum_i P(Y_t|x_t^{(i)}, \lambda_t) \tag{10.4}$$

where the summation is over a set of particles sampled from $P(x_t|x_{t-1})$. This of course leads to an algorithm with $O(N^2)$ complexity in the number of particles as a Monte Carlo approximation is needed for each particle in the main filter. However, provided both the transition density and the likelihood have a simple, unimodal form, the inner approximation may require only a small number of particles, rendering the scheme tractable.

10.1.2 SISR proposals for MCMC

The main failure mode of the MCMC-PF tracker occurs when there are two possible convincing routes for a target to take - generally one correct but with some missed detections and one marked by an unfortunate chain of clutter observations. The sampler gets onto the wrong path (is often initialised in it) and fails to jump to the other.

The SISR-PF tracker does not have this problem, because the particles are generated in parallel. However, it only achieves acceptable performance if independence assumptions are introduced to reduce the dimensionality - effectively to run an independent PF on each target.

It may be possible to combine the two schemes. First, taking the particle distribution from the previous processing step, a set of independent SISR-PFs can be run. The resulting marginal

distributions may then be used as proposal distributions for moves in a joint-target MCMC-PF. Such a scheme may benefit from the advantages of the two different algorithms, maintaining a complete joint representation of the target space while minimising track loss. Furthermore, if the right set of values from the independent filters are stored in memory there is very little additional calculation required for the joint-target MCMC-PF stage.

Such an algorithm is likely to fail when many targets are close together. In such a case, many of the proposals will result in association contradictions, and the MH acceptance rate will fall. Such invalid proposals can be rejected very quickly as we only need to check target associations rather than calculate transition or likelihood probabilities. Thus, we can mitigate this problem by simply lengthening the chain.

10.1.3 MCMC importance sampling

The weakness of the MCMC-PF can be viewed in an another way. When the probability of the correct path is low, the SISR-PF is able to maintain it in the particle distribution with a number of low-weight particles until its probability rises again. In an MCMC scheme, all particles have equal weights, and such low-probability modes may get very few or no particles. An alternative way to combine the advantages of the SISR and MCMC methods would be to construct a Markov chain which generated a weighted particle distribution.

In an SISR scheme, a set of weighted particles $\{x_{1:t}^{(m)}, W_t^{(m)}\}$ are used to approximate the posterior distibution, $P(x_{1:t}|Y_{1:t})$. The unweighted particles may be considered to be samples from some underlying distribution, $\mu(x_{1:t}|Y_{1:t})$ (determined by the proposal distribution and previous resampling steps). Potentially, a Markov chain could be targetted on $\mu(x_{1:t}|Y_{1:t})$ instead of $P(x_{1:t}|Y_{1:t})$.

Suppose at the m^{th} step in the chain we have the state $x_{1:t}^{(m)}$, with unnormalised weight $W_t^{(m)}$. A new state, $x_{1:t}^*$ is sampled from a MH proposal distribution $q(.|x_{1:t}^{(m)}, Y_{1:t})$. The standard acceptance probability formula will be

$$\alpha = \min\left\{1, \frac{\mu(x_{1:t}^*|Y_{1:t})q(x_{1:t}^{(m)}|x_{1:t}^*, Y_{1:t})}{\mu(x_{1:t}^{(m)}|Y_{1:t})q(x_{1:t}^*|x_{1:t}^{(m)}, Y_{1:t})}\right\} \tag{10.5}$$

For any given particle, we know that $P(x_{1:t}^{(m)}|Y_{1:t}) = W_t^{(m)}\mu(x_{1:t}^{(m)}|Y_{1:t})$ by definition, so

$$\alpha = \min\left\{1, \frac{W_t^{(m)}}{W_t^*} \times \frac{P(x_{1:t}^*|Y_{1:t})q(x_{1:t}^{(m)}|x_{1:t}^*, Y_{1:t})}{P(x_{1:t}^{(m)}|Y_{1:t})q(x_{1:t}^*|x_{1:t}^{(m)}, Y_{1:t})}\right\} \tag{10.6}$$

But with this formulation W_t^* is unknown. In fact, it could be set to anything: $\mu(x_{1:t}|Y_{1:t})$

is defined by $P(x_{1:t}|Y_{1:t})$ and the choice of weights rather than vice versa. An obvious choice of weight is to select

$$W_t^* = W_t^{(m)} \times \frac{P(x_{1:t}^*|Y_{1:t})q(x_{1:t}^{(m)}|x_{1:t}^*, Y_{1:t})}{P(x_{1:t}^{(m)}|Y_{1:t})q(x_{1:t}^*|x_{1:t}^{(m)}, Y_{1:t})} \tag{10.7}$$

which results in an acceptance probability of 1 for all moves.

One possible choice of proposal would be to select a state history, $x_{1:t-1}$ from the previous particle distribution, $\hat{P}(x_{1:t-1}|Y_{1:t-1})$, and a current state dependent on this. This would result in a recursive cancellation in the particle weights, leading to (assuming $W_t^{(0)} = 0$):

$$W_t^* = \frac{P(Y_t|x_t^*)P(x_t^*|x_{t-1}^*)}{q(x_t^*|x_{1:t-1}^*, Y_t)} \tag{10.8}$$

which is just the normal weight update formula for a simple SISR particle filter! However, we need not necessarily propose an entire new state for every particle, but can change one component (target) at a time. The result will be an SISR-like algorithm in which particles and weights are not generated independently but dependent on the previous particle and weight. In addition, we can choose different forms for the weight update equation. At the other end of the spectrum, we could set $W_t^* = 1$ which gives us a standard MCMC-PF. Other choices could give some sort of intermediate filter which would allow some variation in weight but not as much as the SISR version.

The hope is that such a scheme should allow us to maintain a complete joint-target posterior distribution and yet not lose track of targets whose path is temporarily of low probability. However, it may simply suffer from all the problems of a joint-target SISR-PF. Specifically, it may discard particles with good estimates of one target because their estimates of another target are poor. In any case, it seems like an interesting new way to view the equivalence of SISR and MCMC.

10.2 Additional tracking problems

10.2.1 Additions to the algorithms

There are various possible extensions to the fixed-lag particle filter trackers. Firstly, the target detection mechanism currently only works with the SISR-PF and requires an independence assumption. It would be desirable to implement target detection for an MCMC-PF. It would also be interesting to try the algorithms on some more nonlinear models with non-Gaussian noise. In such cases the full particle filters would be expected to perform better than their

Rao-Blackwellised equivalents. Models for manoeuvring targets would also be of interest, such as those of (Li & Jilkov 2003) or the intrinsic coordinate models of (Godsill 2007).

Furthermore, it would be interesting to compare the algorithms with an MC-JPDAF and a proper MHT, to see how they compare in terms of tracking performance and computation time.

10.2.2 Target correspondence

In all the studies conducted so far there has been only one sensor observing the scene. Simple situations with multiple sensors can be handled by introducing additional association variables, as in (Vermaak et al. 2005). However, another problem of interest is that of matching the targets viewed by one sensor with those of another when the time periods of observation may not be overlapping, the characteristics of the sensors may be very different, including different biases, and the fields of view may not overlap. In particular, if one object in a group is of particular interest and has been identified by one sensor, we want to be able to identify it with the other sensor, based on the its location in the group. Such a problem will require the detection and tracking of objects as well as simultaneous estimation of correspondence.

10.2.3 Other applications

So far the work conducted has been entirely theoretical, using simulated data with no specific application in mind. It would be interesting to test the fixed-lag particle filter trackers on some real data. Suggested potential applications include tracking groups of people or animals in video sequences, or cells in microscopy data. Extensions for additional latent variables would also be interesting, such as the group structure variables used in (Pang et al. 2011).

10.3 Fixed lag particle filtering

There are numerous other situations in which fixed-lag particle filters may be of use. For example, the music transcription algorithms of (Bunch & Godsill 2010) could well be improved by revising particle estimates once further data has been received. Note onsets and offsets would certainly be more easily detected.

Another area in which the fixed-lag particle filters could be applied is in change-point detection, for example in the jump-diffusion models of (Godsill 2007) and (Christensen, Murphy & Godsill 2012). Here a particle filter is used to estimate a latent series of jump times in a signal. Intuitively, it should be easier to see whether a jump has occured after further data has

Task	Time estimate (months)
Application to more nonlinear models	1/2
Mixed exact-flow PF for tracking	3
Independent SISR proposals for an MCMC-PF tracker	1
MCMC importance sampling	2
Cursor intentionality tracking	1/2
Fixed-lag PFs for jump-diffusion models	2
Target correspondence and other high-level modelling	6
Application of trackers to real data	3
Fixed-lag PFs for music transcription	3

Table 10.1:

been received, so a fixed-lag particle filter may improve the estimate of jump times and thus the predictive power of the algorithm.

10.4 A plan for future research

As anyone working in signal processing will be aware, prediction is a difficult task, prone to exponential growth in uncertainty. Practically speaking, any attempt to plan out a PhD with a horizon longer than a few months is unlikely to bare much resemblence to the course actually followed. So in table 10.1 we simply list a number of the aforementioned interesting possibilities in a rough order of priority with an estimate of the time required.

The time estimates in table 10.1 sum to 22 months, leaving 4 months before the 3 year deadline (or 7 months before funding ends) for writing up. Realistically, it is far more likely that one of the tasks provides numerous interesting opportunities and takes over the entire course of research.

A

Association proposals

We begin with some useful identities. A Gaussian transition density model for a single target is given by

$$P(x_{t+1}|x_t) = \mathcal{N}(x_{t+1}|Ax_t, Q) \tag{A.1}$$

If the multiple step transition density is written as

$$P(x_{t+d}|x_t) = \mathcal{N}(x_{t+1}|A_d x_t, Q_d) \tag{A.2}$$

then we can write the recursion

$$\begin{aligned} P(x_{t+d+1}|x_t) &= \int P(x_{t+d+1}|x_{t+d})P(x_{t+d}|x_t)dx_{t+d} \\ &= \int \mathcal{N}(x_{t+d+1}|Ax_{t+d}, Q)\mathcal{N}(x_{t+d}|A_d x_{t+d}, Q_d)dx_{t+d} \\ &= \mathcal{N}(x_{t+d+1}|A \times A_d x_t, AQ_d A^T + Q) \end{aligned} \tag{A.3}$$

Hence, by induction, we have

$$\begin{aligned} A_d &= A^d \\ Q_d &= \sum_{k=0}^{d-1} (A^k)Q(A^k)^T \end{aligned} \tag{A.4}$$

Also, for $\lambda_t \neq 0$, and using the factorisation of equation 4.6

$$\begin{aligned} P(y_{t+d}^{(\lambda_{t+d})}|x_t) &\propto \int \mathcal{N}(y_{t+d}^{(\lambda_{t+d})}|C_{t+d}x_{t+d}, R)\mathcal{N}(x_{t+d}|A_d x_t, Q_d)dx_{t+d} \\ &= \mathcal{N}(y_{t+d}^{(\lambda_{t+d})}|C_{t+d}A_d x_t, R_d) \end{aligned} \tag{A.5}$$

where

$$R_d = R + C_{t+d}Q_d C_{t+d}^T \tag{A.6}$$

Here we derive an approximation for the "optimal" association proposal for a linear-Gaussian model. For nonlinear models, linear apprximations may be used in the style of the EKF. The optimum form for the proposal distribution is given by

$$\begin{aligned} &q(\lambda_{t-L+1:t}|x_{t-L}, Y_{t-L+1:t}) \\ &\qquad = P(\lambda_{t-L+1:t}|x_{t-L}, Y_{t-L+1:t}) \\ &\qquad \propto P(Y_{t-L+1:t}|\lambda_{t-L+1:t}, x_{t-L})P(\lambda_{t-L+1:t}|x_{t-L}) \\ &\qquad = \prod_{\substack{k=1:L \\ tt=t-L+k}} P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L})P(\lambda_{tt}) \end{aligned} \tag{A.7}$$

where the backward-predictive likelihood factors may be expanded as

$$P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L}) = \int P(Y_{tt}|x_{tt}, \lambda_{tt})P(x_{tt}|Y_{tt+1:t}, \lambda_{tt+1:t}, x_{t-L})dx_{tt} \tag{A.8}$$

We use the approximation

$$P(x_{tt}|Y_{tt+1:t}, \lambda_{tt+1:t}, x_{t-L}) \approx P(x_{tt}|Y_{tt+d}, \lambda_{tt+d}, x_{t-L}) \tag{A.9}$$

where d is the minimum positive value such that $\lambda_{tt+d} \neq 0$. This can be expanded as

$$
\begin{aligned}
&P(x_{tt}|Y_{tt+d}, \lambda_{tt+d}, x_{t-L}) \\
&\qquad \propto \int P(Y_{tt+d}|x_{tt+d}, \lambda_{tt+d}) P(x_{tt+d}|x_{tt}) dx_{tt+d} P(x_{tt}|x_{t-L}) \\
&\qquad \propto \int \mathcal{N}(y_{tt+d}^{(\lambda_{tt+d})}|Cx_{tt+d}, R)\mathcal{N}(x_{tt+d}|A_d x_{tt}, Q_d) dx_{tt+d} \mathcal{N}(x_{tt}|A_k x_{t-L}, Q_k) \\
&\qquad \propto \mathcal{N}(y_{tt+d}^{(\lambda_{tt+d})}|C_{tt+d}A_d x_{tt}, R + C_{tt+d}Q_d C_{tt+d}^T)\mathcal{N}(x_{tt}|A_k x_{t-L}, Q_k)
\end{aligned} \tag{A.10}
$$

Substituting equation A.10 into equation A.8, we have

$$
\begin{aligned}
&P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L}) \\
&\qquad \tilde{\propto} \int \mathcal{N}(y_{tt}^{(\lambda_{tt})}|C_{tt}x_{tt}, R)\mathcal{N}(y_{tt+d}^{(\lambda_{tt+d})}|C_{tt+d}A_d x_{tt}, R + C_{tt+d}Q_d C_{tt+d}^T)\mathcal{N}(x_{tt}|A_k x_{t-L}, Q_k) dx_{t} \\
&\qquad \propto \int \exp(-\frac{1}{2}\xi) dx_{tt} \\
&\qquad \propto \exp(-\frac{1}{2}\zeta)
\end{aligned} \tag{A.11}
$$

The challenge now is to rearrange ξ and hence find ζ. Superscript λs are omitted for clarity.

$$
\begin{aligned}
\xi &= (y_{tt} - C_{tt}x_{tt})^T R^{-1}(y_{tt} - C_{tt}x_{tt}) \\
&\quad + (y_{tt+d} - C_{tt+d}x_{tt+d})^T R_d^{-1}(y_{tt+d} - C_{tt+d}x_{tt+d}) \\
&\quad + (x_{tt} - A_k x_{t-L})^T Q_k^{-1}(x_{tt} - A_k x_{t-L}) \\
&= x_{tt}^T \underbrace{[C_{tt}^T R^{-1} C_{tt} + A_d^T C_{tt+d}^T R_d^{-1} C_{tt+d} A_d + Q_k^{-1}]}_{\Sigma_{tt}^{-1}} x_{tt} \\
&\quad - 2 \underbrace{[y_{tt}^T R^{-1} C_{tt} + y_{tt+d}^T R_d^{-1} C_{tt+d} A_d + x_{t-L}^T A_k^T Q_k^{-1}]}_{\mu_{tt}^T \Sigma_{tt}^{-1}} x_{tt} \\
&\quad + \underbrace{[y_{tt}^T R^{-1} y_{tt} + y_{tt+d}^T R_d^{-1} y_{tt+d} + x_{t-L}^T A_k^T Q_k^{-1} A_k]}_{C}
\end{aligned} \tag{A.12}
$$

Carrying out the integral, an expression for ζ is now obtainable. Terms independent of y_{tt} are absorbed into the constant of proportionality, which is unimportant as it will cancel out during normalisation.

$$\begin{aligned}\zeta &= C - \mu_{tt}^T \Sigma_{tt} \mu_{tt} \\ &= y_{tt}^T \underbrace{[I - R^{-1} C_{tt} \Sigma C_{tt} R^{-1}]}_{S_{tt}^{-1}} y_{tt} \\ &\quad - 2 \underbrace{[y_{tt+d}^T R_d^{-1} C_{tt+d} A_d \Sigma_{tt} C_{tt}^T R^{-1} + x_{t-L}^T A_k^T Q_k^{-1} \Sigma_{tt} C_{tt}^T R^{-1}]}_{m^T S^{-1}} y_{tt} \\ &\quad + \text{terms independent of } \zeta \end{aligned} \tag{A.13}$$

Hence we arrive at the final expression

$$P(Y_{tt} | Y_{tt+1:t} \lambda_{tt:t}, x_{t-L}) \propto \mathcal{N}(y_{tt}^{(\lambda_{tt})} | m_t, S_t) \tag{A.14}$$

where

$$\begin{aligned} m_t &= S_t R^{-1} C_{tt} \Sigma_{tt} [A_d^T C_{tt}^T R_d^{-1} y_{tt}^{(\lambda_{tt})} + Q_k^{-1} A_k x_{t-L}] \\ S_t &= [I - R^{-1} C_{tt} \Sigma_{tt} C_{tt} R^{-1}] \\ \Sigma_t &= [C_{tt}^T R^{-1} C_{tt} + A_d^T C_{tt+d}^T R_d^{-1} C_{tt+d} A_d + Q_k^{-1}]^{-1} \end{aligned} \tag{A.15}$$

and as before

$$\begin{aligned} A_d &= A^d \\ Q_d &= \sum_{k=0}^{d-1} (A^k) Q (A^k)^T \\ R_d &= R + C_{tt+d} Q_d C_{tt+d}^T \end{aligned} \tag{A.16}$$

We need a different expression for the case when $\lambda_{tt+d} = 0$ for all $d = 1 : t - tt$. In this case, the future observations tell us nothing about the current state. Now the expression for the backward state prediction becomes

$$P(x_{tt} | Y_{tt+1:t}, \lambda_{tt+1:t}, x_{t-L}) = P(x_{tt} | x_{t-L}) \tag{A.17}$$

Proceeding as before, we have

$$
\begin{aligned}
&P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L}) \\
&\qquad \propto \int \mathcal{N}(y_{tt}^{(\lambda_{tt})}|C_{tt}x_{tt}, R)\mathcal{N}(x_{tt}|A_k x_{t-L}, Q_k)dx_{tt} \\
&\qquad = \mathcal{N}(y_{tt}^{(\lambda_{tt})}|C_{tt}A_k x_{t-L}, R_k) \qquad \text{(A.18)}
\end{aligned}
$$

Finally, in the special case where $\lambda_{tt} = 0$,

$$
P(Y_{tt}|Y_{tt+1:t}\lambda_{tt:t}, x_{t-L}) \propto V^{-1} \qquad \text{(A.19)}
$$

Bibliography

Al-Awadhi, F., Hurn, M. & Jennison, C. (2004). Improving the acceptance rate of reversible jump mcmc proposals, *Statistics & Probability Letters* **69**(2): 189 – 198.
URL: `http://www.sciencedirect.com/science/article/B6V1D-4CT4DWS-2/2/cab144a2f70e81f189a0e819c4517adc`

Bar-Shalom, Y., Chang, K. & Blom, H. (1989). Automatic track formation in clutter with a recursive algorithm, *Decision and Control, 1989., Proceedings of the 28th IEEE Conference on*, IEEE, pp. 1402–1408.

Bar-Shalom, Y., Daum, F. & Huang, J. (2009). The probabilistic data association filter, *Control Systems Magazine, IEEE* **29**(6): 82 –100.

Bar-Shalom, Y. & Tse, E. (1975). Tracking in a cluttered environment with probabilistic data association, *Automatica* **11**(5): 451–460.

Bayes, M. & Price, M. (1763). An essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfrs, *Philosophical Transactions* **53**: 370.

Blackman, S. (2004). Multiple hypothesis tracking for multiple target tracking, *IEEE Aerospace and Electronic Systems Magazine* **19**(1): 5–18.

Blake, A., Isard, M. et al. (1998). *Active contours*, Vol. 8, Springer London.

Briers, M., Doucet, A., Maskell, S. & Horridge, P. (2006). Fixed-lag sequential monte carlo data association, *Proceedings of SPIE*, Vol. 6236, p. 62360S. Paper copy from British Library.

Bunch, P. & Godsill, S. (2010). Transcription of musical audio using poisson point processes and sequential mcmc, *Proc. of 7th International Symposium on Computer Music Modeling and Retrieval*.

Cappé, O., Godsill, S. & Moulines, E. (2007). An overview of existing methods and recent advances in sequential monte carlo, *Proceedings of the IEEE* **95**(5): 899–924.

Carmi, A., Godsill, S. & Septier, F. (2009). Evolutionary mcmc particle filtering for target cluster tracking, *IEEE 13th DSP Workshop and the 5th SPE Workshop*, Marco Island, Florida.

Casella, G. & Robert, C. P. (1996). Rao-blackwellisation of sampling schemes, *Biometrika* **83**(1): 81–94.
URL: `http://biomet.oxfordjournals.org/content/83/1/81.abstract`

Chib, S. (1996). Calculating posterior distributions and modal estimates in markov mixture models, *Journal of Econometrics* **75**(1): 79 – 97. Details forward-filtering-backward-sampling algorithm.
URL: `http://www.sciencedirect.com/science/article/B6VC0-3VWT1TW-7/2/b66dae744497c95ff3eddb744f301a6c`

Christensen, H. L., Murphy, J. & Godsill, S. J. (2012). Forecasting high-frequency futures returns using online langevin dynamics, *submitted to IEEE journal of selected topics in signal processing* .

Clapp, T. & Godsill, S. (1999). Fixed-lag smoothing using sequential importance sampling, *Bayesian Statistics 6: the Sixth Valencia International Meeting, June 1998, Valencia, Spain*, Oxford University Press.

Daum, F. (2005). Nonlinear filters: beyond the kalman filter, *Aerospace and Electronic Systems Magazine, IEEE* **20**(8): 57 –69.

Daum, F. & Huang, J. (2003). Curse of dimensionality and particle filters, *Aerospace Conference, 2003. Proceedings. 2003 IEEE*, Vol. 4, pp. 1979–1993.

Daum, F., Huang, J. & Noushin, A. (2010). Exact particle flow for nonlinear filters, *Proceedings of SPIE*, Vol. 7697, p. 769704.

Del Moral, P., Doucet, A. & Jasra, A. (2006). Sequential monte carlo samplers, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **68**(3): 411–436.

Doucet, A., Briers, M. & Sénécal, S. (2006). Efficient block sampling strategies for sequential monte carlo, *Journal of Computational and Graphical Statistics* **15**(3): 693–711.

Doucet, A., Godsill, S. & Andrieu, C. (2000). On sequential monte carlo sampling methods for bayesian filtering, *Statistics and Computing* **10**: 197–208. 10.1023/A:1008935410038.
URL: `http://dx.doi.org/10.1023/A:1008935410038`

Doucet, A. & Johansen, A. M. (2009). A tutorial on particle filtering and smoothing: Fifteen years later, *in* D. Crisan & B. Rozovsky (eds), *The Oxford Handbook of Nonlinear Filtering*, Oxford University Press.
URL: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.157.772&rep=rep1&type=pdf`

Doucet, A., Vo, B.-N., Andrieu, C. & Davy, M. (2002). Particle filtering for multi-target tracking and sensor management, *Information Fusion, 2002. Proceedings of the Fifth International Conference on*, Vol. 1, pp. 474 – 481 vol.1.

Duane, S., Kennedy, A. D., Pendleton, B. J. & Roweth, D. (1987). Hybrid monte carlo, *Physics Letters B* **195**(2): 216 – 222.
URL: `http://www.sciencedirect.com/science/article/pii/037026938791197X`

Fitzgerald, R. (1985). Track biases and coalescence with probabilistic data association, *Aerospace and Electronic Systems, IEEE Transactions on* **AES-21**(6): 822 –825.

Fitzgerald, R. J. (1986). Development of practical pda logic for multitarget tracking by microprocessor, *American Control Conference, 1986*, pp. 889 –898.

Fortmann, T., Bar-Shalom, Y. & Scheffe, M. (1983). Sonar tracking of multiple targets using joint probabilistic data association, *Oceanic Engineering, IEEE Journal of* **8**(3): 173 – 184.

Geman, S. & Geman, D. (1984). Gibbs distributions, and the bayesian restoration of images, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**(2): 721–741.

Gilks, W. R. & Berzuini, C. (2001). Following a moving targetmonte carlo inference for dynamic bayesian models, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **63**(1): 127–146.
URL: `http://dx.doi.org/10.1111/1467-9868.00280`

Gilks, W., Richardson, S. & Spiegelhalter, D. (1996). *Markov chain Monte Carlo in practice*, Chapman and Hall.

Girolami, M. & Calderhead, B. (2011). Riemann manifold langevin and hamiltonian monte carlo methods, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **73**(2): 123–214.
URL: `http://dx.doi.org/10.1111/j.1467-9868.2010.00765.x`

Godsill, S. (2007). Particle filters for continuous-time jump models in tracking applications, *ESAIM: Proc.* **19**: 39–52.
URL: `http://dx.doi.org/10.1051/proc:071907`

Godsill, S. J., Doucet, A. & West, M. (2004). Monte carlo smoothing for nonlinear time series, *Journal of the American Statistical Association* **99**(465): 156–168.
URL: `http://pubs.amstat.org/doi/abs/10.1198/016214504000000151`

Godsill, S., Vermaak, J., Ng, W. & Li, J. (2007). Models and algorithms for tracking of maneuvering objects using variable rate particle filters, *Proceedings of the IEEE* **95**(5): 925 –952.

Golightly, A. & Wilkinson, D. (2006). Bayesian sequential inference for nonlinear multivariate diffusions, *Statistics and Computing* **16**: 323–338. 10.1007/s11222-006-9392-x.
URL: `http://dx.doi.org/10.1007/s11222-006-9392-x`

Gordon, N., Salmond, D. & Smith, A. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation, *Radar and Signal Processing, IEE Proceedings F* **140**(2): 107 –113.

Green, P. (1995). Reversible jump markov chain monte carlo computation and bayesian model determination, *Biometrika* **82**(4): 711.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications, *Biometrika* **57**(1): 97–109.
URL: `http://biomet.oxfordjournals.org/content/57/1/97.abstract`

Horridge, P. & Maskell, S. (2006). Real-time tracking of hundreds of targets with efficient exact jpdaf implementation, *Information Fusion, 2006 9th International Conference on*, pp. 1 –8.

Horridge, P. & Maskell, S. (2009). Searching for, initiating and tracking multiple targets using existence probabilities, *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pp. 611 –617.

Hue, C., Le Cadre, J.-P. & Perez, P. (2002). Tracking multiple objects with particle filtering, *Aerospace and Electronic Systems, IEEE Transactions on* **38**(3): 791 – 812.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems, *Journal Of Basic Engineering* **82**(Series D): 35–45.
URL: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.6247&rep=rep1&type=pdf`

Karlsson, R. & Gustafsson, F. (2001). Monte carlo data association for multiple target tracking, *Target Tracking: Algorithms and Applications (Ref. No. 2001/174), IEE*, Vol. 1, pp. 13/1 – 13/5 vol.1.

Khan, Z., Balch, T. & Dellaert, F. (2005). Mcmc-based particle filtering for tracking a variable number of interacting targets, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **27**(11): 1805 –1819.

Kitagawa, G. (1996). Monte carlo filter and smoother for non-gaussian nonlinear state space models, *Journal of Computational and Graphical Statistics* **5**(1): pp. 1–25.
URL: `http://www.jstor.org/stable/1390750`

Laplace, P. S. (1774). Memoir on the probability of the causes of events, *Statistical Science* **1**(3): pp. 364–378.
URL: `http://www.jstor.org/stable/2245476`

Li, X. & Jilkov, V. (2003). Survey of maneuvering target tracking. part i: Dynamic models, *IEEE Transactions on Aerospace and Electronic Systems* **39**(4): 1333–1364.

Liu, J. S. & Chen, R. (1995). Blind deconvolution via sequential imputations, *Journal of the American Statistical Association* **90**(430): pp. 567–576.
URL: `http://www.jstor.org/stable/2291068`

Mahler, R. (1994). Random-set approach to data fusion, *Proceedings of SPIE*, Vol. 2234, p. 287.

Mahler, R. (2003). Multitarget bayes filtering via first-order multitarget moments, *Aerospace and Electronic Systems, IEEE Transactions on* **39**(4): 1152 – 1178.

Mahler, R. (2004). "statistics 101" for multisensor, multitarget data fusion, *Aerospace and Electronic Systems Magazine, IEEE* **19**(1): 53 –64.

Maskell, S., Rollason, M., Gordon, N. & Salmond, D. (2003). Efficient particle filtering for multiple target tracking with application to tracking in structured images, *Image and Vision Computing* **21**(10): 931 – 939.
URL: `http://www.sciencedirect.com/science/article/B6V09-4938V06-1/2/131145f5f06d8b8234e719b32a12fd74`

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E. et al. (1953). Equation of state calculations by fast computing machines, *The journal of chemical physics* **21**(6): 1087.

Musicki, D. & Evans, R. (2004). Joint integrated probabilistic data association: Jipda, *Aerospace and Electronic Systems, IEEE Transactions on* **40**(3): 1093 – 1099.

Musicki, D., Evans, R. & Stankovic, S. (1994). Integrated probabilistic data association, *Automatic Control, IEEE Transactions on* **39**(6): 1237 –1241.

Oh, S., Russell, S. & Sastry, S. (2004). Markov chain monte carlo data association for general multiple-target tracking problems, *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, Vol. 1, pp. 735 –742 Vol.1.

Oh, S., Russell, S. & Sastry, S. (2009). Markov chain monte carlo data association for multi-target tracking, *Automatic Control, IEEE Transactions on* **54**(3): 481 –497.

Orton, M. & Fitzgerald, W. (2002). A bayesian approach to tracking multiple targets using sensor arrays and particle filters, *Signal Processing, IEEE Transactions on* **50**(2): 216 – 223.

Pang, S., Godsill, S., Li, J. & Septier, F. (2011). Sequential inference for dynamically evolving groups of objects, *in* D. Barber, A. Cemgil & S. Chiappa (eds), *Inference and Learning in Dynamic Models*, CUP. To Appear.

Pang, S. K., Li, J. & Godsill, S. (2008). Models and algorithms for detection and tracking of coordinated groups, *Aerospace Conference, 2008 IEEE*, pp. 1 –17.

Pitt, M. K. & Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters, *Journal of the American Statistical Association* **94**(446): pp. 590–599.
URL: `http://www.jstor.org/stable/2670179`

Pitt, M. & Shephard, N. (2001). Auxiliary variable based particle filters, *in* A. Doucet, N. de Freitas & N. Gordon (eds), *Sequential Monte Carlo in Practice*, Springer, pp. 273–293.

Polson, N. G., Stroud, J. R. & Müller, P. (2008). Practical filtering with sequential parameter learning, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **70**(2): 413–428.
URL: `http://dx.doi.org/10.1111/j.1467-9868.2007.00642.x`

Rauch, H., Tung, F. & Striebel, C. (1965). Maximum likelihood estimates of linear dynamic systems, *AIAA journal* **3**(8): 1445–1450.

Reid, D. (1979). An algorithm for tracking multiple targets, *Automatic Control, IEEE Transactions on* **24**(6): 843 – 854.

Salmond, D. (1990). Mixture reduction algorithms for target tracking in clutter, *Proceedings of SPIE*, Vol. 1305, p. 434.

Särkkä, S., Vehtari, A. & Lampinen, J. (2007). Rao-blackwellized particle filter for multiple target tracking, *Information Fusion* **8**(1): 2 – 15. Special Issue on the Seventh International Conference on Information Fusion-Part II, Seventh International Conference on Information Fusion.
URL: `http://www.sciencedirect.com/science/article/B6W76-4HJRRX5-1/2/c4d5356937753f21dc812b7943ffd6cd`

Schulz, D., Burgard, W., Fox, D. & Cremers, A. (2001). Tracking multiple moving targets with a mobile robot using particle filters and statistical data association, *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, Vol. 2, pp. 1665 – 1670 vol.2.

Sea, R. G. (1971). An efficient suboptimal decision procedure for associating sensor data with stored tracks in real-time surveillance systems, *Decision and Control, 1971 IEEE Conference on*, Vol. 10, pp. 33 –37.

Septier, F., Pang, S. K., Carmi, A. & Godsill, S. (2009). On mcmc-based particle methods for bayesian filtering: Application to multitarget tracking, *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2009 3rd IEEE International Workshop on*, pp. 360 –363.

Singer, R., Sea, R. & Housewright, K. (1974). Derivation and evaluation of improved tracking filter for use in dense multitarget environments, *Information Theory, IEEE Transactions on* **20**(4): 423 – 432.

Streit, R. & Luginbuhl, T. (1994). Maximum likelihood method for probabilistic multihypothesis tracking, *Proceedings of SPIE*, Vol. 2235, p. 394.

Svensson, L., Svensson, D. & Willett, P. (2009). Set jpda algorithm for tracking unordered sets of targets, *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pp. 1187 –1194.

Vermaak, J., Godsill, S. & Perez, P. (2005). Monte carlo filtering for multi target tracking and data association, *Aerospace and Electronic Systems, IEEE Transactions on* **41**(1): 309 – 332.

Vo, B.-N. & Ma, W.-K. (2006). The gaussian mixture probability hypothesis density filter, *Signal Processing, IEEE Transactions on* **54**(11): 4091 –4104.

Vo, B.-N., Singh, S. & Doucet, A. (2005). Sequential monte carlo methods for multitarget filtering with random finite sets, *Aerospace and Electronic Systems, IEEE Transactions on* **41**(4): 1224 – 1245.

Whiteley, N., Singh, S. & Godsill, S. (2010). Auxiliary particle implementation of probability hypothesis density filter, *Aerospace and Electronic Systems, IEEE Transactions on* **46**(3): 1437 –1454.

Willett, P., Ruan, Y. & Streit, R. (2002). Pmht: problems and some solutions, *Aerospace and Electronic Systems, IEEE Transactions on* **38**(3): 738 – 754.

Wood, T. (2010). Random finite set theory for tracking.
URL: `http://people.maths.ox.ac.uk/~woodtm/rfs_overview.pdf`