INCOMPLETE DRAFT

An introduction to particle filters

David Salmond and Neil Gordon

Sept 2005

1 Introduction

Aims The aim of this tutorial is to introduce particle filters to those with a background in "classical" recursive estimation based on variants of the Kalman filter. We describe the principles behind the basic particle filter algorithm and provide a detailed worked example. We also show that the basic algorithm is a special case of a more general particle filter that greatly extends the filter design options. The paper concludes with a discussion of computational issues and application areas.

The emphasis of this paper is on principles and applications at an introductory level. It is not a rigorous treatise on the subject nor is it by any means an exhaustive survey. For a more detailed introduction (especially from a target tracking perspective) the reader is referred to the textbook [1] (which uses the same notation as this paper). For a collection of papers on theoretical foundations and applications see [2] and a special issue of the IEEE Transactions on Signal Processing (Monte Carlo Methods for Statistical Signal Processing) [3].

Recursive estimation There is an enormous range of applications that require on-line estimates and predictions of an evolving set of parameters given uncertain data and dynamics - examples include: object tracking, forecasting of financial indices, vehicle navigation and control, and environmental prediction. There is, therefore, a huge "market" for effective recursive estimation algorithms. Furthermore, if these problems can be posed in a common framework, it may be possible to apply general techniques over these varied domains. An obvious common framework consists of a dynamics model (describing the evolution of the system) and a measurement model that describes how available data is related to the system. If

there models can be expressed in a probabilistic form, a Bayesian approach may be adopted.

Bayesian estimation The aim of a Bayesian estimator is to construct the posterior probability density function (pdf) of the required state vector using all available information. The posterior pdf is a complete description of our state of knowledge about (or uncertainty in) the required vector. As such, it is key to optimal estimation - in the sense of minimizing a cost function - and to decision and control problems. The recursive Bayesian filter provides a formal mechanism for propagating and updating the posterior pdf as new information (measurements) is received. If the dynamics and measurement models can be written in a linear form with Gaussian disturbances, the general Bayesian filter reduces to the Kalman filter that has become so wide spread over the last forty years. (All Kalman-like estimators are founded on the genius of Gauss.) Mildly nonlinear problems can be linearized for Kalman filtering, but grossly nonlinear or non-Gaussian cases cannot be handled in this way.

A particle filter is an implementation of the formal recursive Bayesian filter using (sequential) Monte Carlo methods. Instead of describing the required pdf as a functional form, in this scheme it is represented approximately as a set of random samples of the pdf. The approximation may be made as good as necessary by choosing the number of samples to be sufficiently large: as the number of samples tends to infinity, this becomes an exact equivalent to the functional form. For multidimensional pdfs, the samples are random vectors. These random samples are the particles of the filter which are propagated and updated according to the dynamics and measurement models. Unlike the Kalman filter, this approach is not restricted by linear-Gaussian assumptions, so much extending the range of problems that can be tackled. The basic form of the particle filter is also very simple, but may be computationally expensive: the advent of cheap, powerful computers over the last ten years has been key to the introduction of particle filters.

2 The basic particle filter

2.1 Problem definition: dynamic estimation

The dynamic estimation problem assumes two fundamental mathematical models: the state dynamics and the measurement equation.

The dynamics model describes how the state vector evolves with time and is

assumed to be of the form

$$\mathbf{x}_k = \mathbf{f}_{k-1}\left(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}\right) \ , \quad \text{for } k > 0 \ . \tag{1}$$

Here $\mathbf{x}_k$ is the state vector to be estimated, k denotes the time step and $\mathbf{f}_{k-1}$ is a known possibly non-linear function. $\mathbf{v}_{k-1}$ is a white noise sequence, usually referred to as the process, system or driving noise. The pdf of $\mathbf{v}_{k-1}$ is assumed known. Note that (1) defines a first order Markov process, and an equivalent probabilistic description of the state evolution is $p\left(\mathbf{x}_k|\mathbf{x}_{k-1}\right)$, which is sometimes called the transition density. For the special case when $\mathbf{f}$ is linear and $\mathbf{v}$ is Gaussian, the transition density $p\left(\mathbf{x}_k|\mathbf{x}_{k-1}\right)$ is also Gaussian.

The measurement equation relates the received measurements to the state vector:

$$\mathbf{z}_k = \mathbf{h}_k\left(\mathbf{x}_k, \mathbf{w}_k\right) \ , \quad \text{for } k > 0 \ , \tag{2}$$

where $\mathbf{z}_k$ is the vector of received measurements at time step k, $\mathbf{h}_k$ is the known measurement function and $\mathbf{w}_k$ is a white noise sequence (the measurement noise or error). Again, the pdf of $\mathbf{w}_k$ is assumed known and $\mathbf{v}_{k-1}$ and $\mathbf{w}_k$ are mutually independent. Thus, an equivalent probabilistic model for (2) is the conditional pdf $p\left(\mathbf{z}_k|\mathbf{x}_k\right)$. For the special case when $\mathbf{h}_k$ is linear and $\mathbf{w}_k$ is Gaussian, $p\left(\mathbf{z}_k|\mathbf{x}_k\right)$ is also Gaussian.

The final piece of information to complete the specification of the estimation problem is the initial conditions. This is the prior pdf $p\left(\mathbf{x}_0\right)$ of the state vector at time $k = 0$, before any measurements have been received. So, in summary, the probabilistic description of the problem is: $p\left(\mathbf{x}_0\right)$, $p\left(\mathbf{x}_k|\mathbf{x}_{k-1}\right)$ and $p\left(\mathbf{z}_k|\mathbf{x}_k\right)$.

2.2 Formal Bayesian filter

As already indicated, in the Bayesian approach one attempts to construct the posterior pdf of the state vector $\mathbf{x}_k$ given all the available information. This posterior pdf at time step k is written $p\left(\mathbf{x}_k|\mathbf{Z}_k\right)$, where $\mathbf{Z}_k$ denotes the set of all measurements received up to and including $\mathbf{z}_k$: $\mathbf{Z}_k = \{\mathbf{z}_i, i = 1, \ldots, k\}$. The formal Bayesian recursive filter consists of a prediction and an update operation. The prediction operation propagates the posterior pdf of the state vector from time step $k-1$ forwards to time step k. Suppose that $p\left(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}\right)$ is available, then $p(\mathbf{x}_k|\mathbf{Z}_{k-1})$, the prior pdf of the state vector at time step $k > 0$ may be obtained via the dynamics model (the transition density):

$$\underbrace{p(\mathbf{x}_k|\mathbf{Z}_{k-1})}_{\text{Prior at } k} = \int \underbrace{p\left(\mathbf{x}_k|\mathbf{x}_{k-1}\right)}_{\text{Dynamics}} \quad \underbrace{p\left(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}\right)}_{\text{Posterior from } k-1} \quad d\mathbf{x}_{k-1} \ . \tag{3}$$

This is known as the Chapman-Kolmogorov equation.

The prior pdf may be updated to incorporate the new measurements $\mathbf{z}_k$ to give the required posterior pdf at time step $k > 0$:

$$\underbrace{p\left(\mathbf{x}_k|\mathbf{Z}_k\right)}_{\text{Posterior}} = \underbrace{p\left(\mathbf{z}_k|\mathbf{x}_k\right)}_{\text{Likelihood}} \underbrace{p\left(\mathbf{x}_k|\mathbf{Z}_{k-1}\right)}_{\text{Prior}} / \underbrace{p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right)}_{\substack{\text{Normalising} \\ \text{denominator}}} . \tag{4}$$

This is Bayes rule, where the normalising denominator is given by $p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right) = \int p\left(\mathbf{z}_k|\mathbf{x}_k\right) p\left(\mathbf{x}_k|\mathbf{Z}_{k-1}\right) d\mathbf{x}_k$. The measurement model regarded as a function of $\mathbf{x}_k$ with $\mathbf{z}_k$ given is the measurement likelihood. The relations (3) and (4) define the formal Bayesian recursive filter with initial condition given by the specified prior pdf $p\left(\mathbf{x}_0|\mathbf{Z}_0\right) = p\left(\mathbf{x}_0\right)$ (where $\mathbf{Z}_0$ is interpreted as the empty set). If (3) is substituted into (4), the prediction and update may be written concisely as a single expression.

The relations (3) and (4) define a very general but formal (or conceptual) solution to the recursive estimation problem. Only in special cases can an exact, closed form algorithm be obtained from this general result. (In other words, only in special cases can the posterior density be exactly characterized by a sufficient statistic of fixed and finite dimension.) By far the most important of these special cases is the linear-Gaussian (L-G) model: if $p\left(\mathbf{x}_0\right)$, $p\left(\mathbf{x}_k|\mathbf{x}_{k-1}\right)$ and $p\left(\mathbf{z}_k|\mathbf{x}_k\right)$ are all Gaussian, then the posterior density remains Gaussian [4] and (3) and (4) reduce to the standard Kalman filter (which recursively specifies the mean and covariance of the posterior Gaussian). Furthermore, for non-linear / non-Gaussian problems, the first recourse is usually to attempt to force the problem into an L-G framework by linearisation. This leads to the extended Kalman filter (EKF) and its many variants. For mildly non-linear problems, this is often a successful strategy and many real systems operate entirely satisfactorily using EKFs. However, with increasingly severe departures from the L-G situation, this type of approximation becomes stressed to the point of filter divergence (exhibited by estimation errors substantially larger than indicated by the filter's internal covariance). For such grossly non-linear problems, the particle filter may be an attractive option.

2.3 Algorithm of the basic particle filter

The most basic particle filter may be viewed as a direct mechanisation of the formal Bayesian filter.

Suppose that a set of N random samples from the posterior pdf $p\left(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}\right)$ $(k > 0)$ is available. We denote these samples or particles by $\{\mathbf{x}_{k-1}^{i\,*}\}_{i=1}^{N}$.

The **prediction** phase of the basic algorithm consists of passing each of these samples from time step $k-1$ through the system model (1) to generate a set of prior samples at time step k. These prior samples are written $\{\mathbf{x}_k^i\}_{i=1}^N$, where

$$\mathbf{x}_k^i = \mathbf{f}_{k-1}\left(\mathbf{x}_{k-1}^{i\,*}, \mathbf{v}_{k-1}^i\right)$$

and $\mathbf{v}_{k-1}^i$ is a (independent) sample drawn from the pdf of the system noise. This straightforward and intuitively reasonable procedure produces a set of samples or particles from the prior pdf $p\left(\mathbf{x}_k|\mathbf{Z}_{k-1}\right)$.

To **update** the prior samples in the light of measurement $\mathbf{z}_k$, a weight $\tilde{w}_k^i$ is calculated for each particle. This weight is the measurement likelihood evaluated at the value of the prior sample: $\tilde{w}_k^i = p\left(\mathbf{z}_k|\mathbf{x}_k^i\right)$. The weights are then normalized so they sum to unity: $w_k^i = \tilde{w}_k^i / \sum_{j=1}^N \tilde{w}_k^j$ and the prior particles are resampled (with replacement) according to these normalized weights to produce a new set of particles:

$$\{\mathbf{x}_k^{i\,*}\}_{i=1}^N \text{ such that } \Pr\{\mathbf{x}_k^{i\,*} = \mathbf{x}_k^j\} = w_k^j \text{ for all } i, j\ .$$

In other words, a member of the set of prior samples is chosen with a probability equal to its normalised weight, and this procedure is repeated N times to build up the new set $\{\mathbf{x}_k^{i\,*}\}_{i=1}^N$. We contend that the new set of particles are samples of the required pdf $p\left(\mathbf{x}_k|\mathbf{Z}_k\right)$ and so a cycle of the algorithm is complete.

Note that the measurement likelihood effectively indicates those regions of the state space that are plausible "explanations" of the observed measurement value. Where the value of the likelihood function is high, these state values are well supported by the measurement, and where the likelihood is low, these state values are unlikely. (And where the likelihood is zero, these state values are incompatible with the measurement model - i.e. they cannot exist.) So the update procedure effectively weights each prior sample of the state vector by its plausibility with respect to the latest measurement. The re-sampling operation is therefore biased towards the more plausible prior samples, and the more heavily weighted samples may well be chosen repeatedly (see discussion of sample impoverishment below). The algorithm is shown schematically in fig ?? and some Matlab code for an example application is given in Section 3.

This simple algorithm is often known as the Sampling Importance Resampling (SIR) filter and it was introduced in 1993 [5] where it called the bootstrap filter. It was independently proposed by a number of other research groups including Kitagawa [6] as a Monte Carlo filter and Isard and Blake [7] as the CONDENSATION Algorithm.

2.4 Empirical distributions

The sample sets described above may also be viewed as empirical distributions for the required state pdfs, i.e. the prior:

$$p\left(\mathbf{x}_k|\mathbf{Z}_{k-1}\right) \approx \frac{1}{N}\sum_{i=1}^{N}\delta(\mathbf{x}_k - \mathbf{x}_k^i) \tag{5}$$

and the posterior in weighted or resampled form:

$$p\left(\mathbf{x}_k|\mathbf{Z}_k\right) \approx \sum_{i=1}^{N} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) \approx \frac{1}{N}\sum_{i=1}^{N}\delta(\mathbf{x}_k - \mathbf{x}_k^{i\,*}) \ .$$

This representation also facilitates a simple justification of the update phase of the basic filter using the "plug-in principle" [8]. Substituting the approximate form of the prior (5) into Bayes rule (4), we obtain:

$$\begin{aligned}
p\left(\mathbf{x}_k|\mathbf{Z}_k\right) &= p\left(\mathbf{z}_k|\mathbf{x}_k\right) p\left(\mathbf{x}_k|\mathbf{Z}_{k-1}\right) / p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right) \\
&\approx p\left(\mathbf{z}_k|\mathbf{x}_k\right) \frac{1}{N}\sum_{i=1}^{N}\delta(\mathbf{x}_k - \mathbf{x}_k^i) / p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right) \\
&= \frac{1}{N}\sum_{i=1}^{N} p\left(\mathbf{z}_k|\mathbf{x}_k^i\right) \delta(\mathbf{x}_k - \mathbf{x}_k^i) / p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right) \\
&= \frac{1}{N}\sum_{i=1}^{N} \tilde{w}_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) / p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right) \\
&= \sum_{i=1}^{N} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) \ ,
\end{aligned}$$

where, by comparison with (4), $p\left(\mathbf{z}_k|\mathbf{Z}_{k-1}\right) \approx \frac{1}{N}\sum_{i=1}^{N}\tilde{w}_k^i$. For a more rigorous discussion of the theory behind the particle filter see [2, 9, 10].

2.5 Alternative resampling scheme

A direct implementation of the resampling step in the update phase of the algorithm would consist of generating N independent uniform samples, sorting them into ascending order and comparing them with the cumulative sum of the normalised weights. This scheme has a complexity of $\mathrm{O}(N \log N)$. There are several alternative approaches including systematic resampling which has complexity of $\mathrm{O}(N)$. In systematic resampling [6], the normalised weights w^i are incrementally summed to form a cumulative sum of $w_C^i = \sum_{j=1}^{i} w^j$. A "comb" of N points spaced at regular

intervals of $1/N$ is defined and the complete comb is translated by an offset chosen randomly from a uniform distribution over $[0, 1/N]$. The comb is then compared with the cumulative sum of weights w_C^i as illustrated in fig 1 for $N = 7$. For this example, the resamped set would consist of labels $2, 3, 3, 5, 6, 6$ and 7 of the original set. This scheme has the advantage of only requiring the generation of a single random sample, irrespective of the number of particles, and it minimises the Monte Carlo variation - see Section 2.7 below. This method is used in the example of Section 3 and so Matlab code for it is given in the example listing.

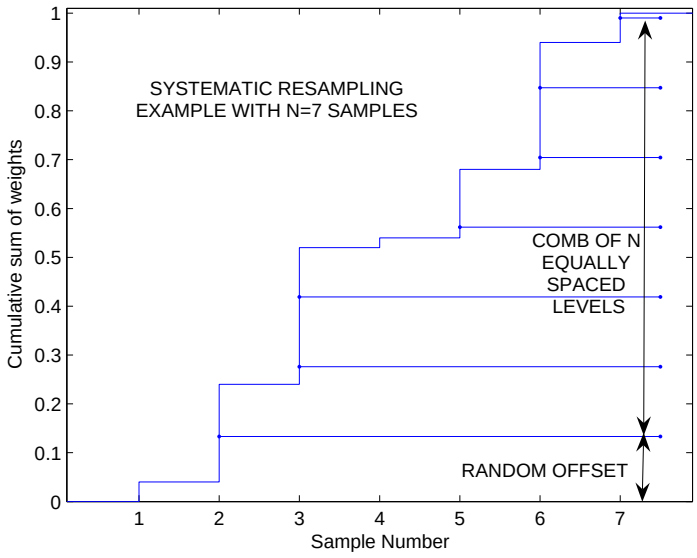

Figure 1: Systematic resampling scheme

2.6 Impoverishment of the sample set

As already noted, in the resampling stage, particles with large weights may be selected many times so that the new set of samples may contain multiple copies of just a few distinct values. This impoverishment of the particle set is the result of sampling from from a discrete rather than a continuous distribution. If the variance of the system driving noise is sufficiently large, these copies will be redistributed in the prediction phase of the filter and adequate diversity in the sample set may be maintained. However, if the system noise is small, or in extreme cases zero (i.e. parameter estimation), the particle set will rapidly collapse and some artificial means of introducing diversity must be introduced. An obvious way of doing this is to perturb or jitter each of the particles after resampling (termed roughening in [5]).

This rather ad hoc procedure can be formalized as regularization - where a kernel is placed over each particle to effectively provide a continuous mixture approximation to the discrete (empirical) distribution (akin to kernel density estimation). Optimal kernels for regularization are discussed in [11]. Another scheme for maintaining diversity is to perform a Monte Carlo move - see [12].

2.7 Degeneracy and effective sample size

In the basic version of the filter described in Section 2.3 above, resampling is performed at every measurement update. The function of this resampling process is to avoid wasting the majority of the computational effort in propagating particles with very low weights. Without resampling, as measurement data is integrated, for most interesting problems the procedure would rapidly collapse to a very small number of highly weighted particles amongst a hoard of almost useless particles carrying a tiny proportion of the probability mass. This results in failure due to an inadequate representation of the required pdf - i.e degeneracy. Although resampling counters this problem, as noted above, it tends to increase impoverishment and so there are good arguments for only carrying out resampling if the particle set begins to degenerate [10, 1].

A convenient measure of degeneracy is the effective sample size [13] defined by $\hat{N}_{eff} = 1/\sum_{j=1}^{N}(w_k^j)^2$ which varies between 1 and N. A value close to 1 indicates that almost all the probability mass is assigned to one particle and there is only one useful sample in the set - i.e. severe degeneracy. Conversely, if the weights are uniformly spread amongst the particles the effective sample size approaches N. It is often suggested that the resampling process should only be performed if $\hat{N}_{eff}$ falls below some threshold (chosen empirically). If resampling is not carried out, the particle weights from the previous time step are updated via the likelihood: $\tilde{w}_k^i = w_{k-1}^i p\left(\mathbf{z}_k|\mathbf{x}_k^i\right)$ and then normalised. In this case the required posterior pdf of the state is given by the random measure $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^N$, and the these particles are passed through the system model in the prediction phase to generate the $\mathbf{x}_{k+1}^i$ for the next measurement update (so the prior distribution at $k+1$ would be approximated by $p\left(\mathbf{x}_{k+1}|\mathbf{Z}_k\right) \approx \sum_{i=1}^{N} w_k^i \delta(\mathbf{x}_{k+1} - \mathbf{x}_{k+1}^i)$).

2.8 Sample representation of the posterior pdf

An important feature of the particle filter is that it provides (an approximation of) the full posterior of the required state. Moreover, the representation of the posterior pdf in the form of a set of samples is very convenient. As well as being straightforward to produce summary statistics, many useful parameters for command, control

and guidance purposes can be easily estimated.

Kalman-like estimators produce estimates of the mean and covariance of the posterior (which completely specify the Gaussian pdf from this type of filter). These statistics are easily estimated from the particle filter sample set (using the plug-in principle) as

$$\hat{\mathbf{x}}_k = E\left[\mathbf{x}_k|\mathbf{Z}_k\right] = \int \mathbf{x}_k p(\mathbf{x}_k|\mathbf{Z}_k) d\mathbf{x}_k \approx \sum_{i=1}^{N} w_k^i \mathbf{x}_k^i \text{ or } \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_k^{i\,*} \text{ and}$$

$$\begin{aligned} \text{cov}(\mathbf{x}_k) &= E\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T|\mathbf{Z}_k\right] = \int (\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T p(\mathbf{x}_k|\mathbf{Z}_k) d\mathbf{x}_k \\ &\approx \sum_{i=1}^{N} w_k^i (\mathbf{x}_k^i - \hat{\mathbf{x}}_k)(\mathbf{x}_k^i - \hat{\mathbf{x}}_k)^T \text{ or } \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_k^{i\,*} - \hat{\mathbf{x}}_k)(\mathbf{x}_k^{i\,*} - \hat{\mathbf{x}}_k)^T . \end{aligned}$$

However, the mean and covariance may be a poor summary of the posterior, particularly if it is multimodal or skewed. A scatter plot of the samples, a histogram or a kernel density estimate [14] are more informative for a 1 or 2-D state vector (or for marginals of the full state vector). Another useful descriptor is the highest probability density (HPD) region. The $(1-\alpha)$HPD region is the set of values of the state vector which contain $1-\alpha$ of the total probability mass, such that the pdfs of all points within the region are greater than or equal to the pdfs of all those outside the region - i.e. if H is the $(1-\alpha)$HPD region, then $\int_H p(\mathbf{x})d\mathbf{x} = 1-\alpha$ and $p(\mathbf{x}') \geq p(\mathbf{x}'')$ for all $\mathbf{x}' \in H$ and $\mathbf{x}'' \notin H$. The HPD region is usually only considered for scalars and it may be difficult to find for multmodal pdfs. A simpler option is to find the percentile points on scalar marginals of the distribution. For example the $(1-\alpha)100$ percentile point is given (roughly) by finding the largest $N\alpha$ samples and choosing the smallest of these.

In many cases, the requirement is find some particular function of the posterior, and the sample representation is often ideal for this. For example, for threat analysis, one may be interested in the probability that a target is within some particular region - this can be estimated by counting the number of particles falling within that region. Also for decision and control problems, an estimate of the expected value of any form of cost or utility function $C(\mathbf{x}_k)$ is simply given by

$$E[C(\mathbf{x}_k)|\mathbf{Z}_k] = \int C(\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{Z}_k) d\mathbf{x}_k \approx \sum_{i=1}^{N} w_k^i C(\mathbf{x}_k^i) \text{ or } \frac{1}{N}\sum_{i=1}^{N} C(\mathbf{x}_k^{i\,*}) .$$

This is the starting point for Monte Carlo approaches to the difficult problem of stochastic control - especially with non-quadratic cost functions [15, 16].

2.9 Discussion

Convenience The basic particle filter is a very simple algorithm and it is quite straightforward to obtain good results for many highly non-linear recursive estimation problems. So problems that would be difficult to handle using an extended Kalman filter, state space gridding or a Gaussian mixture approach are quite accessible to the "novice" via a blind application of the basic algorithm. Although this is hugely liberating, it is something of a mixed blessing : there is a danger that such challenging cases are not treated with proper respect and that subtleties and implications of the problem are not appreciated [17].

Generality The particle approach is very general. It is not restricted to a particular class of distribution or to a form of dynamics model (although the filters discussed in this paper do rely on the Markov property). So for example, the dynamics may include discrete jumps and densities may be multi-modal with disconnected regions. Furthermore, the measurement likelihood and transition density do not have to be analytical functions - some form of look-up table is quite acceptable. Also support regions with hard edges can easily be included (see Section 6 below).

3 Example

To demonstrate the operation of the particle filter we present an application to a pendulum estimation problem. A weightless rigid rod of length L is freely pivoted at one end and carries a mass at its other end. The rod makes an angle θ with the horizontal and its instantaneous angular acceleration is given by

$$\ddot{\theta} = (1/L)(-g + v)\cos\theta$$

where g is the acceleration due to gravity and v is a random disturbance. This differential equation is the motivation for the following simple discrete dynamics model:

$$\left.\begin{array}{rcl} \theta_k &=& \mod\ [\theta_{k-1} + \Delta t\theta_{k-1} + (\Delta t^2/2L)(-g + v_{k-1})\cos\theta_{k-1}\ ,\ 2\pi] \\ \dot{\theta}_k &=& \dot{\theta}_{k-1} + (\Delta t/L)(-g + v_{k-1})\cos\theta_{k-1} \end{array}\right\} \quad (6)$$

where θ has been restricted to the range $[0, 2\pi)$, Δt is the fixed time step and the acceleration disturbance v_k is a zero mean, white, Gaussian random sequence of variance q. So this example has a two element state vector $\mathbf{x}_k = \left(\theta_k, \dot{\theta}_k\right)^T$ for $k > 0$. Measurements are obtained from the length of the rod projected onto a vertical axis,

i.e. $L|\sin\theta_k|$. These measurements are quantized at intervals of δ, but are otherwise error-free, so

$$z_k = \mathcal{Q}_\delta\left(L|\sin\theta_k|\right) \ ,$$

where the quantization operator $\mathcal{Q}_\delta(x) = (n-1)\delta$ for the integer n such that $(n-1)\delta < x \leq n\delta$. Thus the likelihood of the state vector is

$$p\left(\mathbf{z}_k|\mathbf{x}_k\right) = \begin{cases} 1 & \text{if } z_k < L|\sin\theta_k| \leq z_k + \delta \\ \\ 0 & \text{otherwise .} \end{cases} \tag{7}$$

In other words, given a measurement z, the projected length of the pendulum is equally likely to be anywhere in the interval $(z, z+\delta]$ but cannot be anywhere else. The problem is to construct the posterior pdf of the state vector $\left(\theta_k, \dot{\theta}_k\right)^T$ given the set of measurements $\mathbf{Z}_k$ and the initial conditions that θ_0 is uniformly distributed over $[0, 2\pi)$ and $\dot{\theta}_0$ is Gaussian distributed with known mean and variance. The dynamics recursion (6), the likelihood (7) and the above initial conditions completely specify the problem for application of a particle filter. This system is illustrated in fig 2 for $\delta = L/3$.

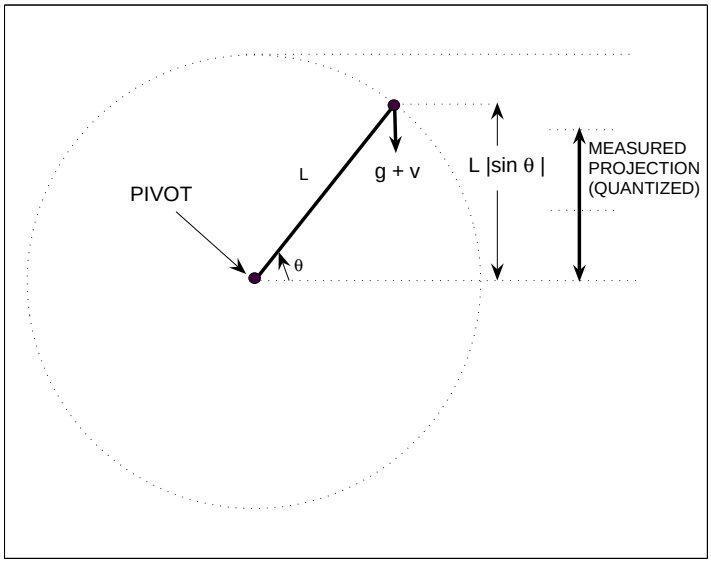

Figure 2: Pendulum with quantized projection measurement

The basic version of the particle filter has been applied to this example. Here each particle is a two element vector $(\theta, \dot{\theta})$. As already indicated, the prediction phase of the filter consists of passing each particle through the dynamics model (6). A Matlab code for this example is shown below. In this code, the posterior particles $\{\mathbf{x}^{i\,*}\}_{i=1}^{N}$ are contained in the $2 \times N$ array `x_post`, where the two rows correspond to θ and $\dot{\theta}$, and each column is an individual particle. Similarly, the prior particles $\{\mathbf{x}^{i}\}_{i=1}^{N}$ are contained in the $2 \times N$ array `x_prior`, and `nsamples` is the number of particles N. The un-normalised weights for each particle are stored in the N element array `likelihood`, while the normalised weights and their cumulative sum are held in `weight`. It is easy to recognise the dynamics equations (6) in the prediction phase and the likelihood (7) in the update phase.

Hopefully, this code listing will clarify the specification of the filter given in Section 2.3. Note that the complete filter can be expressed in a few lines of Matlab: the basic algorithm is (embarrassingly) simple. Furthermore, there are no "hidden extras": the code does not call any sophisticated numerical algorithms (numerical integration packages, eigenvector solvers, etc) or symbolic manipulation packages - except perhaps for the random number generator and the Matlab array handling routines.

```
%************************************************************************
%  Generate initial samples for k=0:
 x_post(1,:)     = 2*pi*rand(1,nsamples);
 x_post(2,:)     = theta_dot_init + sig_vel_init*randn(1,nsamples);

for k=1:nsteps

    % PREDICT
    F1           = dt*dt/(2*pend_len);      F2=dt/pend_len;
    drive1       = randn(1,nsamples);   % random samples for system noise
    accn_in      = (-gee+drive1*sig_a).*cos(x_post(1,:));
    x_prior(1,:) = mod( x_post(1,:) + dt*x_post(2,:) + F1*accn_in , 2*pi );
    x_prior(2,:) = x_post(2,:)      + F2*accn_in;

    % UPDATE
    % EVALUATE WEIGHTS resulting form meas(k):
    project      = pend_len.*abs(sin(x_prior(1,:)));  % rod projection for each sample
    likelihood   = zeros(nsamples,1);
    likelihood( find( project>=meas(k) & project<meas(k)+delta ) )=1;
    weight       = likelihood/sum(likelihood);    % normalise weights
    weight       = cumsum(weight);                % form cumulative distribution

    %  RE-SAMPLING PROCEDURE (SYSTEMATIC)
    addit=1/nsamples; stt=addit*rand(1);
    selection_points=[ stt : addit : stt+(nsamples-1)*addit ];  j=1; %set up comb
    for i=1:nsamples
      while selection_points(i) >= weight(j); j=j+1; end;
      x_post(:,i) = x_prior(:,j);
    end;

    % OUTPUT:  store posterior particles (for analysis only)
    samp_store(:,:,k) = x_post;
end
%************************************************************************
```

This program has been applied to the data set shown in fig 3. Here the quantization interval is $\delta = L/2$, so the only information available form the measurements is whether the projected rod length is greater or less than $L/2$ (i.e. one bit of information). The other parameters of this simulation are $\theta_0 = 0.3$ rads, $\dot{\theta}_0 = 2$ rads/s, $\Delta t = 0.05$ s, $L = 3$ m, $g = 10$ m/s^2 and the standard deviation of the driving noise v is 7 m/s^2. In the 10 second period shown, the pendulum changes its direction of rotation twice (after about 1 s and 8 s), between which it makes a complete rotation. The initial conditions supplied to the particle filter are that the angular velocity $\dot{\theta}_0$ is from a Gaussian distribution with a mean of 2.4 rads/s and a standard deviation of 0.4 rads/s. As already stated, the initial angle θ_0 is uniformly distributed over

$[0, 2\pi)$ rads. The initial particle set is drawn from these distributions as shown in the above listing.

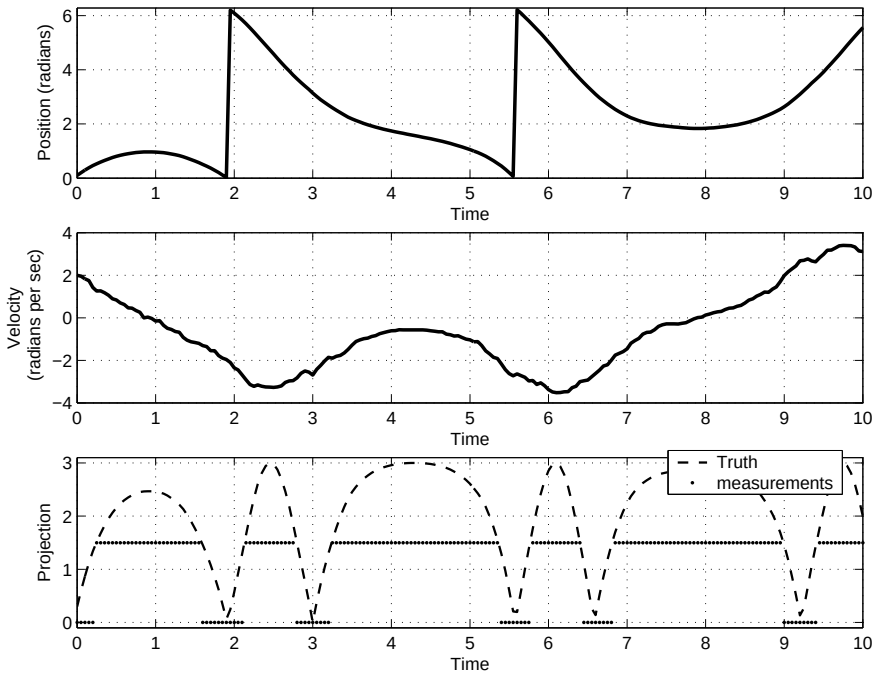

Figure 3: Truth and measurements

The result of running the filter with $N = 1000$ particles is shown in fig 4. This figure shows the evolution of the posterior pdf of the angle θ obtained directly from the posterior particles. The pdf for each of the 200 time steps is a simple histogram of the posterior angle particles. The evolving distribution consists of streams or paths of modes that cross and pass through regions of bifurcation. For this case of $\delta = L/2$, the measurements switch between 0 and $L/2$ whenever $|\sin\theta| = 0.5$, i.e when $\theta = \pi/6, 5\pi/6, 7\pi/6$ or $11\pi/6$. As is evident from fig 4, at these transition points, the pdf modes sharpen. Occasionally a path is terminated if it is incompatible with a measurement transition (for example, for $\theta = 11\pi/6$ at about $k = 15$). The actual angle of the pendulum is shown as a string of dots.

Note that $N = 1000$ is adequate to give a fairly convincing estimate of the posterior density, although it appears a little ragged in the region $\theta = \pi/6$ to $5\pi/6$ about the vertically up position (where the angular velocity tends to be low and the pendulum may swing back or continue over the top). The ragged structure can be smoothed by increasing the number of particles - fig 5 shows the evolving pdf for the extravagantly large value of $N = 50000$. This produces a pleasingly smooth result

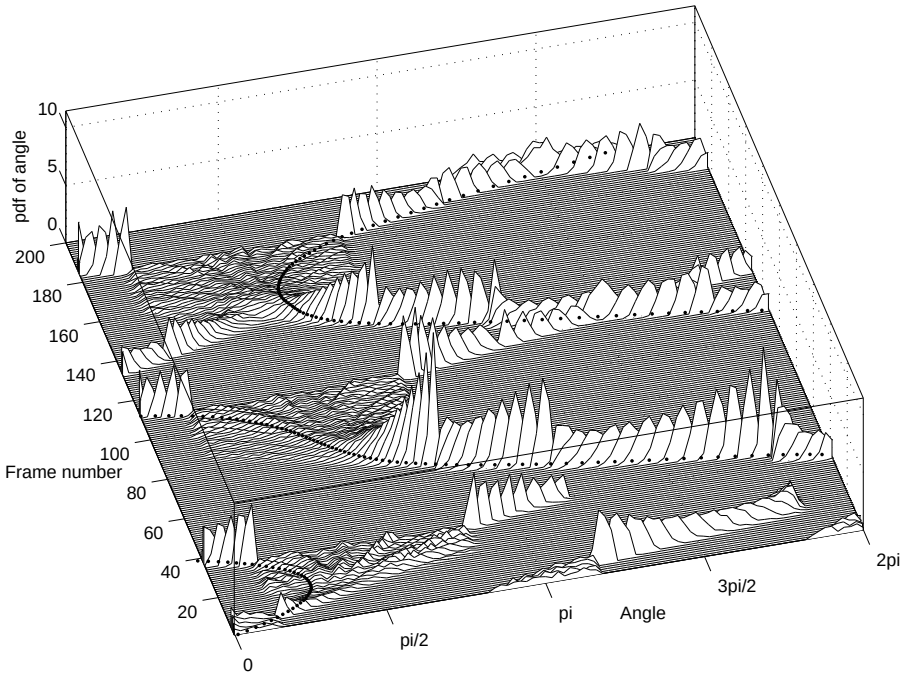

Figure 4: Posterior pdf from 1000 samples

but is otherwise very similar to the 1000 particle result. The $N = 1000$ case took about 5 ms per time step to run on a ??MHz Xeon processor - a quite acceptable rate for the quality of the result. For $N = 50000$, the time taken increased almost linearly to 260 ms per time step. Note that apart from the obvious time penalty, it is trivial to improve filter performance to approach the exact posterior pdf.

Discussion This filtering example was chosen to demonstrate the particle filter because it is simple to specify and would be difficult to tackle using an extended Kalman filter (or an Unscented Kalman filter). It is also a low dimensional case and so is an easy example for a particle filter. With some effort, it would be possible to develop a multiple hypothesis Kalman filter to capture the multi-modal nature of the posterior pdf and to include the 2π wrap around in angle. Also it might be possible to to represent the quantization function as a form of Gaussian mixture. However, this would all be quite awkward and definitely approximate (and would probably be more computationally expensive). The particle approach avoids all such difficulties in this example. Also, the traditional summary descriptors of recursive estimation - the mean and covariance - would be quite inappropriate for this example where the posterior pdf is often multi-modal and sometimes unimodal but highly skewed.

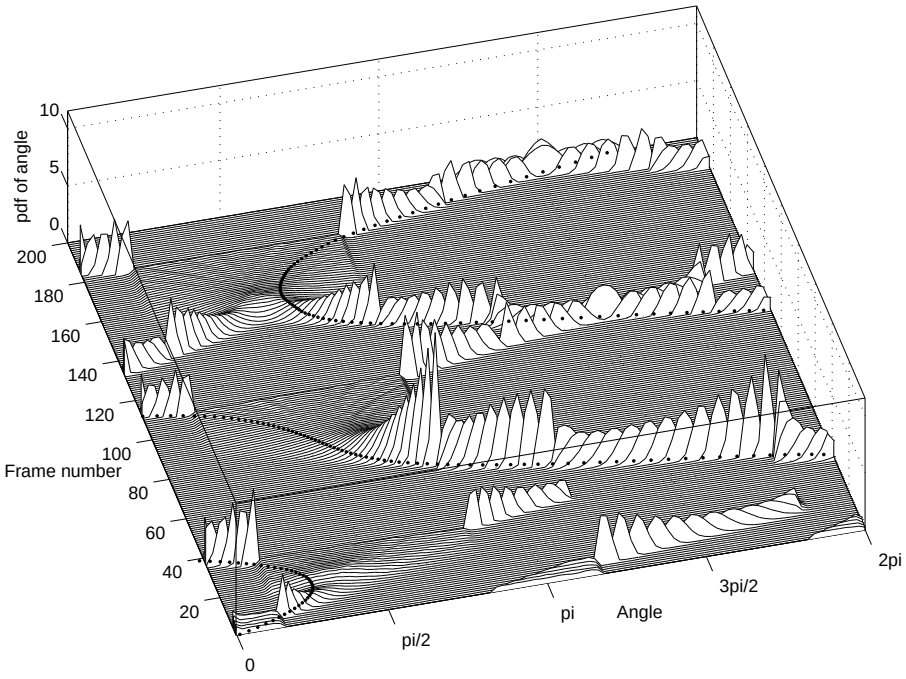

Figure 5: Posterior pdf from 50000 samples

4 More general particle filters

In the basic version of the particle filter, the particles $\{\mathbf{x}_k^i\}_{i=1}^N$ used to construct the empirical posterior pdf $\sum_{i=1}^N w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i)$ are assumed to be samples from the prior $p(\mathbf{x}_k|\mathbf{Z}_{k-1})$. Furthermore, these samples $\{\mathbf{x}_k^i\}_{i=1}^N$ are obtained from the posterior samples $\{\mathbf{x}_{k-1}^{i\,*}\}_{i=1}^N$ of the previous time step by passing them through the dynamics model. In other words, each support point $\mathbf{x}_k^i$ is a sample of the transition pdf $p\left(\mathbf{x}_k|\mathbf{x}_{k-1}^{i\,*}\right)$ conditional on $\mathbf{x}_{k-1}^{i\,*}$. However, it is not necessary to generate the $\{\mathbf{x}_k^i\}_{i=1}^N$ in this way, they may be obtained from any pdf (known as an importance or proposal density) whose support includes that of the required posterior $p(\mathbf{x}_k|\mathbf{Z}_k)$. In particular, the importance pdf may depend on $\mathbf{z}_k$, the value of the measurement at time step k. This more general approach considerably broadens the scope for filter design.

The more general formulation is a two stage process similar to the basic filter of Section 2.3, but these stages do not correspond directly to prediction and update phases. As before, we assume that N random samples $\{\mathbf{x}_{k-1}^{i\,*}\}_{i=1}^N$ of the posterior pdf $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ at time step $k-1$ are available.

Sampling : For each particle $\mathbf{x}_{k-1}^{i\,*}$, draw a sample $\mathbf{x}_k^i$ from an importance density $q\left(\mathbf{x}_k|\mathbf{x}_{k-1}^{i\,*}, \mathbf{z}_k\right)$.

Weight evaluation : The unnormalised weight corresponding to sample $\mathbf{x}_k^i$ is given by:

$$\tilde{w}_k^i = \frac{p\left(\mathbf{z}_k|\mathbf{x}_k^i\right) p\left(\mathbf{x}_k^i|\mathbf{x}_{k-1}^{i\,*}\right)}{q\left(\mathbf{x}_k^i|\mathbf{x}_{k-1}^{i\,*}, \mathbf{z}_k\right)} \,. \tag{8}$$

As before, the weights are normalised $w_k^i = \tilde{w}_k^i / \sum_{j=1}^N \tilde{w}_k^j$ and the empirical pdf of the posterior is given by $p\left(\mathbf{x}_k|\mathbf{Z}_k\right) \approx \sum_{i=1}^N w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i)$. Resampling with replacement according to the normalised weights produces a set of samples $\{\mathbf{x}_k^{i\,*}\}_{i=1}^N$ of the posterior pdf $p\left(\mathbf{x}_k|\mathbf{Z}_k\right)$. Note that if the importance density is chosen to be the transition pdf, i.e. $q\left(\mathbf{x}_k^i|\mathbf{x}_{k-1}^{i\,*}, \mathbf{z}_k\right) = p\left(\mathbf{x}_k^i|\mathbf{x}_{k-1}^{i\,*}\right)$, equation (8) reverts to the basic particle filter. The general from of the weight equation (8) is essentially a modification of the basic form to compensate for the different importance density.

The advantage of this formulation is that the filter designer can choose any importance density $q\left(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k\right)$ provided its support includes that of $p\left(\mathbf{x}_k|\mathbf{Z}_k\right)$. If this condition is met, as $N \to \infty$, the resulting sample set $\{\mathbf{x}_k^{i\,*}\}_{i=1}^N$will be distributed as $p\left(\mathbf{x}_k|\mathbf{Z}_k\right)$. This flexibility allows one to place samples where they are needed to provide a good representation of the posterior - i.e. in areas of high probability density rather than in sparse regions. In particular, since the importance density may depend on the value of the received measurement $\mathbf{z}_k$, if the measurement is very accurate (or if it strongly localizes the state vector in some sense), the importance samples can be placed in the locality defined by $\mathbf{z}_k$ [18]. This is especially important if the "overlap" between the prior and the likelihood is low - adjusting the importance density could avoid wasting a high percentage of the particles (i.e. impoverishment). There is considerable scope for ingenuity in designing the importance density and a number of particle filter versions have been suggested for particular choices of this density. An optimal importance density may be defined as one that minimises the variance of the importance weights. For the special case of non-linear dynamics with additive Gaussian noise, a closed form expression for the optimal importance density can be obtained [10]. In general such an analytical solution is not possible, but sub-optimal results based on local linearisation (via an EKF or unscented Kalman filter) may be employed [1].

As in the basic version of the filter, it is not necessary to carry out resampling at every time step. If resampling is omitted, the particle weights from the previous time step are updated according to:

$$\tilde{w}_k^i = w_{k-1}^i \frac{p\left(\mathbf{z}_k|\mathbf{x}_k^i\right) p\left(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i\right)}{q\left(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{z}_k\right)} \,. \tag{9}$$

This general result is known as Sequential Importance Sampling and it is most easily derived by (formally) considering the full time history or trajectory of each particle and marginalizing out past time steps [1, 2, 9, 10, 18]. This result is also the starting point for for most expositions on particle filter theory (although, unusually, in this paper the development has been from specific to general).

The Rao-Blackwellized or marginalized particle filter In many cases, it may be possible to divide the problem into linear-Gaussian and non-linear parts. Suppose that the state vector may be partitioned as $\mathbf{x}_k = \begin{pmatrix} \mathbf{x}_k^L \\ \mathbf{x}_k^N \end{pmatrix}$ so that the required posterior may be factorized into Gaussian and non-Gaussian terms:

$$p\left(\mathbf{x}_k|\mathbf{Z}_k\right) = p\left(\mathbf{x}_k^L, \mathbf{x}_k^N|\mathbf{Z}_k\right) = p\left(\mathbf{x}_k^L|\mathbf{x}_k^N, \mathbf{Z}_k\right) p\left(\mathbf{x}_k^N|\mathbf{Z}_k\right) \ ,$$

where $p\left(\mathbf{x}_k^L|\mathbf{x}_k^N, \mathbf{Z}_k\right)$ is Gaussian (conditional on $\mathbf{x}_k^N$) and $p\left(\mathbf{x}_k^N|\mathbf{Z}_k\right)$ is non-Gaussian. In other words, the linear component of the state vector $\mathbf{x}_k^L$ can be "marginalized out". Essentially, the term $p\left(\mathbf{x}_k^L|\mathbf{x}_k^N, \mathbf{Z}_k\right)$ may be obtained from a Kalman filter while the non-Gaussian part $p\left(\mathbf{x}_k^N|\mathbf{Z}_k\right)$ is given by a particle filter. The scheme requires that a Kalman filter update be performed for each $\mathbf{x}_k^N$ particle - see [19] for a full specification of the algorithm. This procedure is generally known as Rao-Blackwellization [20, 10, 21]. The main advantage of this approach is that the dimension of the particle filter state $\mathbf{x}_k^N$ is less than that of the full state vector, so that less particles are required for satisfactory filter performance - see below. This comes at the cost of a more complex algorithm, although the operation count of the marginallized filter for a given number of particles may actually be less than that of the standard algorithm (see [22]).

5 Computational issues

5.1 Computational cost for the basic filter

The computational cost of the basic particle filter (with systematic resampling) is almost proportional to the number N of particles employed, both in terms of operation count and memory requirements. The computational effort associated with each particle clearly depends directly on the complexity of the system dynamics and the measurement process. For example, problems involving measurement association uncertainty may require a substantial measurement likelihood calculation (i.e. a summation over hypotheses). For such cases there is a strong motivation to find efficient ways of evaluating the likelihood - including approximate gating and the use of likelihood ratios (see examples in Chapters 11 and 12 of [1]).

A notable advantage of the particle filter is that the available computational resources can be fully exploited by simply adjusting the number of particles - so it is easy to take advantage of the ever increasing capability cheap computers. Similarly, if the measurement data rate is variable, the filter can match the number of particles to the available time interval to optimize performance. (However, if the number of particles falls below a critical level, the filter performance may degrade to a point from which it cannot recover.) Also note that the filter is amenable to parallelization - until a resampling event occurs, all particle operations are independent. (Some recent developments on a parallel particle filter in the context of multiple target tracking are reported in [23].)

5.2 How many samples?

This is the most common question about particle filters and there is no simple answer. Classical analysis of Monte Carlo sampling does not apply as the underlying assumption - that the samples are independent - is violated. In the basic particle filter, immediately after the resampling stage, many of the particles are almost certainly identical - definitely not independent. So unfortunately, particle filters are not immune to the curse of dimensionality, although with careful filter design the curse can be moderated - see the informative and detailed discussion by Daum [17, 24]. Generally, based on simple arguments of populating a multidimensional space, one must expect the required number of particles to increase with the dimension of the state vector - hence the attraction of the Rao-Blackwellized or marginalized form of the filter.

The required sample size depends strongly on the design of the particle filter and the problem being addressed (dimension of state vector, volume of support, etc). For certain problems, especially high dimensional ones, an enormous, infeasible, number of samples is required to obtain satisfactory results with the basic filter. To obtain a practical algorithm in these circumstances, the designer has to be inventive. The theory outlined in Section 4 provides a rigorous framework for exploring options, and, with a careful choice of proposal distribution and/or exploiting Rao-Blackwellization, it may be possible to design a filter that gives quite satisfactory performance with a modest number of particles (a few hundred or even tens in some cases). However, the basic algorithm has the advantage of simplicity, so that the operation count for each particle may be much lower than for a more subtle filter. Practical particle filter design is therefore a compromise between these approaches with the aim of minimizing the overall computational load. Also note that heuristic tricks may well be helpful.

The usual way of determining when enough samples are being deployed is via

trial and error: the sample size is increased until the observed error in the parameter of interest (from a set of representative simulation examples) falls to a steady level. If the required sample size is too large for the available processing resources, one may have to settle for sub-optimal filter performance or attempt to improve the design of the filter. This empirical approach is not entirely satisfactory, and more work in this area is required to obtain, at least, guidelines that are of use to practising (as opposed to academic) engineers.

Finally, note that filter initialization is often the most challenging aspect of a recursive estimation problem. In particular, if the prior information (i.e. before measurements are received) is vague, so that the initial uncertainty spans a large volume of state space, the direct (obvious) approach of populating the prior pdf with particles may be very wasteful. Semi-batch schemes using the first few measurement frames may be useful.

6 Applications

Particle filters have been employed in a wide range of domains: essentially, wherever there is a requirement to estimate the state of a stochastic evolving system using uncertain measurement data. Below, we briefly indicate some of the more successful or popular applications (with a bias towards tracking problems).

Tracking and navigation with a bounded support Particle filters are ideal for problems where the state space has a restricted or bounded support. Examples include, targets moving on a road network (the Ground Moving Target Indicator (GMTI) problem - see [25] and chapter 10 of [1]), inside a building [26] or in restricted waters [27, 28]. Hard edges and boundaries, which cannot be easily accommodated by Kalman-type filters, do not pose any difficulty for the particle approach. Essentially, the bounded support is simply flooded with particles.

Tracking with non-standard sensors The classical non-linear tracking test case is the bearings-only problem with passive sensors (acoustic, electro-optical or electronic support measures (ESM)), and particle filters have certainly been applied to examples of this type (see [5, 28] and chapter 6 of [1]). However, particle tracking filters have also been successfully implemented with range-doppler sensors that provide measurements of only observer-target range and range rate (see chapter 7 of [1]). An interesting application to a network of binary sensors (i.e. each sensor provides one bit of information) is reported in [29]. Also particle filtering of raw sensor outputs (such as pixel grey levels) have been examined by a number of workers in the context of track-before-detect (see chapter 11 of [1] and [30, 31]).

Multiple object tracking and association uncertainty The obvious way of approaching multiple target tracking problems is to concatenate the state vectors of individual targets and attempt to estimate the combined state. This approach is appropriate if the targets' dynamics are interdependent (for example, formation or group dynamics - see chapter 12 of [1]) or if there is measurement association uncertainty (or unresolved targets) due to object proximity [32]. Particle filters have been successfully applied in these cases for small numbers of objects, although the evaluation of the likelihood function (for every particle) can be expensive as it involves summing over feasible assignment hypotheses. An alternative more efficient route suggested in [33] is to employ a Probabilistic Multiple Hypothesis Tracker (PMHT) likelihood which effectively imposes independence between object-measurement assignments. This approach may also be viewed as a superposition of Poisson target models (possibly including extended objects) [34]. Particle filtering is also the implementation mechanism for the finite-set statistics Probability Hypothesis Density (PHD) filter [35, 36, 37].

Computer vision and robotics Particle filtering was introduced to the computer vision community as the CONDENSATION algorithm [7]. In this application, the state vector includes shape descriptors as well as dynamics parameters. This has been a successful domain for particle filters and there is now a substantial literature especially in IEEE Computer Society Conferences on Computer Vision and Pattern Recognition (CVPR) and IEEE International Conferences on Pattern Recognition (ICPR). Applications include tracking of facial features (especially using active contours or "snakes"), gait recognition and people tracking (some recent publications include [38, 39, 40, 41]). Particle filters are also well represented in the robotics literature: they have been successfully applied to localization, mapping and fault diagnosis problems [26, 42, 43, 44].

Econometrics Progress in this field has tended to parallel, but remain largely independent of, engineering developments. However, in the case of particle filtering and Monte Carlo methods, there has been perhaps more "cross-over" than usual. Econometric applications include stochastic volatility modelling for stock indices and commodity prices [45, 46, 47, 48].

Numerical weather prediction The requirement here is to update model states with observational data from, for example, weather satellites. This is known as data assimilation and can be viewed as a (very large) nonlinear dynamic estimation and prediction problem. A range of techniques are employed including EKFs and "ensemble Kalman filters" which use samples for non-linear state propagation but fit a Gaussian for the Kalman update operation. Recently, the use of full particle

filters for data assimilation has been considered [49].

7 Concluding remarks

Over the past few years, particle filters have become a popular topic. There have been a large number of papers (arguably too many) demonstrating new applications and algorithm developments This popularity may be due to the simplicity and generality of the basic algorithm - it is easy to get started. Furthermore, the particle filter is not another variant of the EKF: it does not stem from linear-Gaussian or least-squares theory. It also appeals to both the "hands on engineer" (there is plenty of scope for algorithm tweaking) and to the more theoretical community (with substantial challenges to develop performance bounds and guidelines for finite sample sizes). Undoubtedly a key enabler for this activity has been the massive increase in the capability of cheap computers - as Daum [17] has recently pointed out "computers are now eight orders of magnitude faster (per unit cost) compared with 1960, when Kalman published his famous paper" .

The basic or naive version of the version of the particle filter may be regarded as a black box algorithm with a single tuning parameter - the number of samples. This filter is very effective for many low dimensional problems, and, perhaps fortuitously, reasonable results have been obtained for state vectors with about ten elements without resorting to an enormous number of particles. For more challenging high dimensional problems, a more subtle approach (exploiting Rao-Blackwellization and carefully chosen proposal distributions) is generally beneficial - there is a design trade-off between many simple or fewer smart particles. This (problem dependent) compromise would benefit from further study.

To date, most particle filter applications have been in simulation studies or off-line with recorded data. However, particle filters are beginning to appear as on-line elements of real systems - mainly in navigation and robotics applications. The technology (and necessary processing capability) is now sufficiently mature to support the leap to such real-time system implementation - we expect to see a significant increase here in coming years.

References

[1] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman filter: particle filters for tracking applications*. Artech House, 2004.

[2] A. Doucet, N. de Freitas, and N. J. Gordon, eds., *Sequential Monte Carlo Methods in Practice*. New York: Springer, 2001.

[3] IEEE, "Special issue on Monte Carlo methods for statistical signal processing," *IEEE Trans. Signal Processing*, vol. 50, February 2002.

[4] Y. C. Ho and R. C. K. Lee, "A Bayesian approach to problems in stochastic estimation and control," *IEEE Trans. Automatic Control*, vol. 9, pp. 333–339, 1964.

[5] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc.-F*, vol. 140, no. 2, pp. 107–113, 1993.

[6] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian non-linear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[7] M. Isard and A. Blake, "CONDENSATION - connditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[8] B. Efron and R. Tibshirani, *An introduction to the Bootstrap*. Chapman and Hall, 1998.

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

[10] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[11] C. Musso, N. Oudjane, and F. LeGland, "Improving regularised particle filters," in *Sequential Monte Carlo Methods in Practice* (A. Doucet, N. de Freitas, and N. J. Gordon, eds.), New York: Springer, 2001.

[12] W. R. Gilks and C. Berzuini, "Following a moving target – Monte Carlo inference for dynamic Baysian models," *Journal of the Royal Statistical Society, B*, vol. 63, pp. 127–146, 2001.

[13] A.Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278–288, 1994.

[14] B. Silverman, *Density estimation for statistics and applied data analysis*. Chapman and Hall, 1986.

[15] C. Andrieu, A. Doucet, S. Singh, and V. Tadic, "Particle methods for change detection, system identification, and control," *Proceedings of the IEEE*, vol. 92, pp. 423–438, March 2004.

[16] D. Salmond, N. Everett, and N. Gordon, "Target tracking and guidance using particles," *American Control Conference, Arlington, Virginia*, pp. 4387–4392, June 2001.

[17] F. Daum, "Nonlinear filters: beyond the Kalman filter," *IEEE A and E Systems Magazine - Part 2: Tutorials II*, vol. 28, pp. 57–69, August 2005.

[18] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for non-linear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, pp. 174–188, February 2002.

[19] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Transactions on Signal Processing*, vol. 53, pp. 2279–2289, July 2005.

[20] G. Casella and C. Robert, "Rao-Blackwellization of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.

[21] A. Doucet, N. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Processing*, vol. 49, pp. 613–624, March 2001.

[22] R. Karlsson, T. Schon, and F. Gustafsson, "Complexity analysis of the marginallized particle filter," *IEEE Transactions on Signal Processing*, to appear.

[23] S. Sutharsan, A. Kirubarajan, and A. Sinha, "An optimization-based parallel particle filter for multitarget tracking," *SPIE: Signal and Data Processing of Small Targets*, vol. 5913, August 2005.

[24] F. Daum and J. Huang, "Curse of dimensionality and particle filters," in *Proceedings of IEEE Aerospace Conference*, (Big Sky), March 2003.

[25] S. Arulampalam, N. Gordon, M. Orten, and B. Ristic, "A variable structure multiple model particle filter for GMTI tracking," in *Fusion 2002: Proceedings of the 5th international conference on Information Fusion*, pp. 927–934, 2002.

[26] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, "Particle filters for mobile robot localization," in *Sequential Monte Carlo Methods in Practice* (A. Doucet, N. de Freitas, and N. J. Gordon, eds.), New York: Springer, 2001.

[27] M. Mallick, S. Maskell, T. Kirubarajan, and N. Gordon, "Littoral tracking using a particle filter," in *Fusion 2002: Proceedings of the 5th international conference on Information Fusion*, pp. 935–942, 2002.

[28] R. Karlsson and F. Gustafsson, "Recursive Bayesian estimation - bearings-only applications," *IEE Proceedings on Radar, Sonar and Navigation*, vol. 152, no. 5, 2005.

[29] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 150–161, ACM Press, 2003.

[30] Y. Boers and J. Driessen, "Multitarget particle filter track before detect application," *IEE Proceedings on Radar, Sonar and Navigation*, vol. 151, no. 6, pp. 351–357, 2004.

[31] M. Rutten, N. Gordon, and S. Maskell, "Efficient particle-based track-before-detect in Rayleigh noise," in *Fusion 2004: Proceedings of the 7th international conference on Information Fusion*, 2004.

[32] Z. Khan, T. Balch, and F. Dellaert, "Multitarget tracking with split and merged measurements," in *CVPR 2005: Proceedings of the 2005 Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[33] C. Hue, J. L. Cadre, and P. Pérez, "Sequential Monte Carlo methods for multiple target tracking and data fusion," *IEEE Trans. Signal Processing*, vol. 50, pp. 309–325, February 2002.

[34] K. Gilholm, S. Godsill, S. Maskell, and D. Salmond, "Poisson models for extended target and group tracking," *SPIE: Signal and Data Processing of Small Targets*, vol. 5913, August 2005.

[35] B.-N. Vo, S. Singh, and A. Doucet, "Random finite sets and sequential monte carlo methods in multi-target tracking," in *Proceedings of International Radar Conference*, pp. 486–491, September 2003.

[36] R. Mahler, "Statistics 101 for multisensor, multitarget data fusion," *IEEE A and E Systems Magazine - Part 2: Tutorials*, vol. 19, pp. 53–64, January 2004.

[37] R. Mahler, "Random sets: unification and computation for information fusion - a retrospective assessment," in *Fusion 2004: Proceedings of the 7th international conference on Information Fusion*, pp. 1–20, 2004.

[38] R. Green and L. Guan, "Quantifying and recognizing human movement patterns from monocular images: Parts I and II," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 2, pp. 179–198, 2004.

[39] L. Wang, H. Ning, T. Tan, and W. Hu, "Fusion of static and dynamic body biometrics for gait recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 2, pp. 149–158, 2004.

[40] J. Tu, Z. Zhang, Zeng, and T. Huang, "Face localization via hierarchial CONDENSATION with Fisher boosting fearture selection," in *CVPR 2004: Proceedings of the 2004 Computer Society Conference on Computer Vision and Pattern Recognition*, pp. II–719–II–724, 2004.

[41] M. de Bruijne and M. Nielsen, "Image segmentation by shape particle filtering," in *ICPR 2004: Proceedings of the 17th International Conference on Pattern Recognition*, 2004.

[42] S. Thrun, "Particle filters in robotics," in *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)*, 2002.

[43] M. Rosencrantz, G. Gordon, and S. Thrun, "Locating moving entities in indoor environments with teams of mobile robots," in *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, (New York, NY, USA), pp. 233–240, ACM Press, 2003.

[44] S. Sutharsan, A. Kirubarajan, and A. Sinha, "Adapting the sample size in particle filters through KLD-sampling," *International journal of robotics research*, vol. 22, pp. 985–1004, October 2003.

[45] G. Kitagawa and S. Sato, "Monte Carlo smoothing and self-oragising state-space model," in *Sequential Monte Carlo Methods in Practice* (A. Doucet, N. de Freitas, and N. J. Gordon, eds.), New York: Springer, 2001.

[46] M. Pitt and N. Shephard, "Auxiliary variable based particle filters," in *Sequential Monte Carlo Methods in Practice* (A. Doucet, N. de Freitas, and N. J. Gordon, eds.), New York: Springer, 2001.

[47] J. Stroud, N. Polson, and P. Mueller, "Practical filtering for stochastic volatility models," in *State Space and Unobserved Component Models* (A. Harvey, S. Koopman, and Shephard, eds.), Cambridge University Press, 2004.

[48] P. Fearnhead, "Using random Quasi-Monte-Carlo within particle filters with application to financial time series," *Journal of Computational and Graphical Statistics*, To appear.

[49] P. J. van Leeuwen, "Nonlinear ensemble data assimilation for the ocean," in *Seminar on recent developments in data assimilation for atmosphere and ocean*, (Shinfield Park, Reading, UK), pp. 265–286, European Centre for Medium-Range Weather Forecasts, September 2003.