# Efficient delay-tolerant particle filtering through selective processing of out-of-sequence measurements

**Xuan Liu, Boris N. Oreshkin and Mark J. Coates**
Department of Electrical and Computer Engineering
McGill university
Montreal, Quebec, Canada
{xuan.liu2, boris.oreshkin}@mail.mcgill.ca; mark.coates@mcgill.ca

**Abstract** − *This paper proposes a novel algorithm for delay-tolerant particle filtering that is computationally efficient and has limited memory requirements. The algorithm estimates the informativeness of delayed (out-of-sequence) measurements (OOSMs) and immediately discards uninformative measurements. More informative measurements are then processed using the storage efficient particle filter proposed by Orguner et al. If the measurement induces a dramatic change in the current filtering distribution, the particle filter is re-run to increase the accuracy. Simulation experiments provide an example tracking scenario where the proposed algorithm processes only 30-40% of all OOSMs using the storage efficient particle filter and 1-3% of OOSMs by re-running the particle filter. By doing so, it requires less computational resources but achieves greater accuracy than the storage efficient particle filter.*

**Keywords:** Tracking, particle filtering, out of sequence measurement (OOSM), resource management.

## 1 Introduction

Tracking is frequently performed using multiple sensor platforms, with measurements being relayed to a central fusion site over a wireless network. This can lead to some measurements being delayed when adverse environmental conditions cause packet losses to occur. The fusion centre is then faced with out-of-sequence measurements (OOSMs). Simply discarding the delayed OOSMs can waste important information and lead to much poorer tracking performance. On the other hand, incorporating the measurements into a particle filter in an efficient manner can be a challenging task.

In this paper, we propose a novel algorithm for delay-tolerant particle filtering that is computationally efficient and has limited memory requirements. The algorithm first applies an inexpensive computational procedure based on extended Kalman filtering to estimate the informativeness of the OOSMs. It immediately discards any measurements which are deemed uninformative. The more informative measurements are processed using the storage efficient particle filter proposed by Orguner et al. in [1]. This algorithm is usually accurate, but it can fail when a measurement is extremely informative, because it does not change the locations of particles but just updates their weights. It thus cannot address situations in which the current particle filter distribution should be changed significantly because of the new out-of-sequence measurement. Our algorithm applies a second test to detect these highly-informative OOSMs, and incorporates them by re-running the particle filter from the time-step of the OOSM. Gaussian approximations of past particle distributions are stored to permit the initialization of the re-run particle filter.

Our simulation experiments provide an example tracking scenario where the proposed algorithm processes only 30-40% of all OOSMs using the storage efficient particle filter and 1-3% of OOSMs by rerunning the particle filter. By doing so, it requires less computational resources but achieves greater accuracy than the storage efficient particle filter proposed in [1].

### 1.1 Paper Organization

The rest of the paper is organized as follows. Section 1.2 discusses the related work. Section 2 provides the problem statement. Section 3 provides an overview of memory efficient OOSM particle filters and Section 4 presents the novel algorithm. The performance of the proposed methods is assessed through simulation in Section 5. We make concluding remarks in Section 6.

### 1.2 Related Work

There has been a substantial amount of research addressing tracking with out-of-sequence measurements. The initial work on this topic focused on tracking systems with linear state and measurement models (see [2] and references therein). In order to address nonlinear models, researchers began to explore methods for efficiently processing OOSMs within particle filters. In [3], Orton et al. proposed an approach that employs the set

of particles before and after the time step of the delayed measurement to update the current weights of particles. This method was improved with a Markov chain Monte Carlo (MCMC) smoothing step to mitigate the potential problem of degeneracy in [4]. The OOSM particle filters proposed in [3,4] need to store all of the particles of the last $l$ steps, ($l$ is the predetermined maximum number of lags), which can lead to an excessive consumption of storage resources. To address this, Mallick et al. proposed an approximate OOSM particle filter that only stores the mean and covariance matrix of the particles [5]. These three OOSM particle filtering algorithms require an inversion of the system dynamics to perform "backwards prediction", which is only possible for linear state dynamics. In subsequent work, both for OOSM processing [1] and particle smoothing [6], researchers have identified procedures for approximate inversion for some non-linear systems; it is possible to incorporate these techniques to extend the applicability of the algorithms in [3–5].

In [1], Orguner et al. propose a number of "storage efficient particle filters" (SEPFs) for OOSM processing. As in [5], the particle filters only store statistics (single mean and covariance) of the particle set, rather than the particles themselves, at previous time steps. Auxiliary fixed point smoothers are employed to determine the likelihood of the delayed measurement conditioned on each particle in the current set, and this likelihood is used to update the weight of the particle. The authors compare the performance of the algorithms using three types of smoother: Extended Kalman Smoother (EKS), Unscented Kalman Smoother (UKS) and Particle Smoother (PS). In their simulations, the EKS outperforms the other methods with least computation. Their experiments address highly non-linear filtering problems where the extended Kalman filter often fails to track the target. The EKS is successful because the particle filter provides it with critical side-information.

The storage efficient particle filters perform well for many OOSM tracking tasks, but their performance can suffer when there are a significant number of highly-informative OOSMs (such as the scenario described in Section 5). Incorporating such OOSMs using the auxiliary fixed point smoother can lead to a major decrease in the effective number of particles. The storage efficient particle filters of [1] therefore choose to discard them.

## 2 Problem Statement

We now provide a mathematical formulation of the OOSM filtering problem. We consider the scenario with Markovian and (possibly) non-linear state dynamics and measurements described by (possibly) non-linear functions of the current state. The innovation and observation noises are modeled as additive Gaussian. At each timestep $k$, there is an active set of distributed sensors, $\mathcal{V}_k$, that make measurements. These measurements are relayed to the fusion centre. A subset of them $\mathcal{S}_k$ experience minimal delay and can be processed at time $k$. Other measurements are delayed and only become available for processing at later timesteps. Measurements that are delayed by more than $l$ timesteps are considered uninformative and are ignored.

The system is described by the following equations:

$$\mathbf{x}_{k+1} = f_{k+1|k}(\mathbf{x}_k) + \mathbf{v}_{k+1|k} \quad (1)$$

$$\mathbf{y}_k^j = h_k^j(\mathbf{x}_k) + \mathbf{s}_k^j \quad (\forall j \in \mathcal{V}_k) \quad (2)$$

$$\mathcal{Y}_k = \{\mathbf{y}_k^{\mathcal{S}_k} : \mathcal{S}_k \subseteq \mathcal{V}_k\} \quad (3)$$

$$\mathcal{Z}_k = \{\mathbf{y}_{k-l}^{\mathcal{D}_{k-l,k}}, \mathbf{y}_{k-l+1}^{\mathcal{D}_{k-l+1,k}}, \ldots, \mathbf{y}_{k-1}^{\mathcal{D}_{k-1,k}}\} \quad (4)$$

Here $\{\mathbf{x}_k\}$ denotes the state sequence, which is a Markov process with initial distribution $\mathbf{x}_0 \sim p(\mathbf{x}_0)$, and $\{\mathbf{y}_k^j\}$ denotes the measurement sequence at the $j$-th sensor. $\mathbf{v}_{k+1|k}$ is the transition noise with Gaussian distribution $\mathcal{N}(0, \mathbf{V}_{k+1|k})$, and $\mathbf{s}_k^j$ is the measurement noise with Gaussian distribution $\mathcal{N}(0, \mathbf{Q}_k^j)$. The functions $f_{k+1|k}(.)$ and $h_k^j(.)$ are the transition and measurement functions. $\mathcal{Y}_k$ denotes the set of non-delayed measurements received at time $k$. $\mathcal{Z}_k$ denotes the set of OOSMs received at time $k$. The set $\mathcal{D}_{\tau,k}$ is the subset of active sensors at time $\tau$ whose measurements are received at time step $k$; $\mathbf{y}_{k-l}^{\mathcal{D}_{k-l,k}}$ is the set of measurements made at time $k-l$ that arrive at the fusion centre at time $k$.

Let $\widetilde{\mathcal{W}}_k^{i:j}$ denote the set of measurements generated in the interval $[i, j]$ available at the fusion centre by time $k$. This includes all the non-delayed measurements $\mathcal{Y}_{i:j} = \cup_{m=i}^{j} \mathcal{Y}_m$ and OOSMs $\{\mathbf{y}_\tau^{\mathcal{D}_{\tau,m}} \in \mathcal{Z}_m : \tau \in [i, j], m \in [1, k]\}$. We also denote $\mathcal{W}_k^{i:j} = \widetilde{\mathcal{W}}_k^{i:j} \setminus \mathcal{Z}_k$, i.e. the set of all measurements available at time $k$ except those in $\mathcal{Z}_k$. Lastly, let $\mathcal{W}_k^j \equiv \mathcal{W}_k^{j:j}$ and $\widetilde{\mathcal{W}}_k^j \equiv \widetilde{\mathcal{W}}_k^{j:j}$.

The OOSM filtering task is to form an estimate of the posterior distribution $p(\mathbf{x}_k|\widetilde{\mathcal{W}}_k^{1:k})$ (and hence an estimate of the state $\mathbf{x}_k$).

## 3 OOSM Particle Filters

OOSM particle filters differ in how they incorporate the OOSMs from the set $\mathcal{Z}_k$. The simplest approach is to discard them, but this can result in poor tracking performance. Another obvious approach is to restart the filter at the time step immediately prior to the time step associated with the earliest OOSM in $\mathcal{Z}_k$ and re-run to the current time step $k$. This requires that we record all the particles, weights and the measurements for the window of time over which OOSMs are considered useful. We call this approach the "OOSM re-run particle filter" and consider it as a benchmark for the accuracy that can be achieved. This method has two unattractive qualities: the storage requirements can be immense (consider 10,000 particles stored for a window of 10 timesteps), and the computation cost is high.

As discussed in Section 1.2, several methods have been proposed to alleviate these costs. In this section, we provide a brief review of the storage efficient particle filter of [1] and introduce a seemingly novel (but obvious) alternative algorithm. In both algorithms, the memory requirements are reduced by storing statistics of the particle sets from past time steps instead of the particles themselves. Effectively, the past particle distributions are represented by Gaussian approximations. The stored information is then the mean and covariance matrix of particles at each time step from $k-l-1$ to $k$. Denote, respectively, by $\boldsymbol{\xi}_k$, $\boldsymbol{\omega}_k$ the sets of the values and weights of particles at time $k$, and let $\boldsymbol{\mu}_k$, $\mathbf{R}_k$ denote their mean and covariance. The stored information is then

$$\boldsymbol{\Omega}_k = \{\boldsymbol{\mu}_{k-l-1:k}, \mathbf{R}_{k-l-1:k}, \mathcal{W}_k^{k-l:k}\}, \tag{5}$$

A generic storage efficient OOSM particle filtering algorithm is summarized in Algorithm 1. If there are no OOSMs at time $k$, we write $\mathcal{Z}_k = \emptyset$.

**Algorithm 1**: Generic OOSM Particle Filter

1 At time $k$
  **Input**: $\mathcal{Z}_k$, $\boldsymbol{\Omega}_k$
2 $(\boldsymbol{\xi}_k, \boldsymbol{\omega}_k) \leftarrow$ `ParticleFilter(`$\mathcal{Y}_k$, $\boldsymbol{\xi}_{k-1}$, $\boldsymbol{\omega}_{k-1}$`)` ;
3 $(\boldsymbol{\mu}_k, \mathbf{R}_k) \leftarrow$ `SaveGauss(`$\boldsymbol{\xi}_k$, $\boldsymbol{\omega}_k$`)` ;
4 **if** $\mathcal{Z}_k \neq \emptyset$ **then**
5  $(\boldsymbol{\xi}_k, \boldsymbol{\omega}_k, \boldsymbol{\Omega}_k) \leftarrow$ `ProcessOOSM(`$\mathcal{Z}_k$, $\boldsymbol{\xi}_k$, $\boldsymbol{\omega}_k$, $\boldsymbol{\Omega}_k$`)`;

In this algorithm, the function `ParticleFilter` can be any standard particle filtering method. If $\mathcal{Y}_k = \emptyset$, `ParticleFilter` only propagates the particles and skips the measurement processing step. The function `SaveGauss` uses the maximum likelihood estimator of the mean and covariance given the weighted sample set $\boldsymbol{\xi}_k$, $\boldsymbol{\omega}_k$:

$$\boldsymbol{\mu}_k = \sum_{i=1}^{N} \boldsymbol{\omega}_k^{(i)} \boldsymbol{\xi}_k^{(i)} \tag{6}$$

$$\mathbf{R}_k = \sum_{i=1}^{N} \boldsymbol{\omega}_k^{(i)} (\boldsymbol{\xi}_k^{(i)} - \boldsymbol{\mu}_k)(\boldsymbol{\xi}_k^{(i)} - \boldsymbol{\mu}_k)^T \tag{7}$$

## 3.1 Gaussian Approximation Re-run Particle Filter (OOSM-GARP)

A simple modification of the re-run particle filter involves storing only Gaussian approximations of the particle distributions at previous timesteps. When a batch of OOSMs arrives, the particle filter is re-run from the time step preceding the earliest OOSM. Since the particle set from that time step is unavailable, particles are generated from the stored approximation.

When OOSM-GARP receives $\mathcal{Z}_k$ at time $k$, it returns to the time step $\widetilde{\tau}_k - 1$ (let $\widetilde{\tau}_k$ denote the earliest time step of all OOSMs in $\mathcal{Z}_k$). It samples particles from $\mathcal{N}(\boldsymbol{\mu}_{\widetilde{\tau}_k-1}, \mathbf{R}_{\widetilde{\tau}_k-1})$, propagates them to the time step $\widetilde{\tau}_k$ and runs the filter as standard particle filter using all stored measurements $\widetilde{\mathcal{W}}_k^{\widetilde{\tau}_k:k}$. At each step, it updates the mean and covariance matrix in the stored set $\boldsymbol{\Omega}_k$ as described in Algorithm 2.

**Algorithm 2**: ProcessOOSM-GARP

  **Input**: $\mathcal{Z}_k$, $\boldsymbol{\Omega}_k$
1 $\widetilde{\tau}_k = \min_{\tau}\{\tau : \mathbf{y}_\tau \in \mathcal{Z}_k\}$ ;
2 $\{\boldsymbol{\xi}_{\widetilde{\tau}_k-1}^{(i)}\}_{i=1}^N \sim \mathcal{N}(\mathbf{x}_{\widetilde{\tau}_k-1}, \boldsymbol{\mu}_{\widetilde{\tau}_k-1}, \mathbf{R}_{\widetilde{\tau}_k-1})$ ;
3 $\boldsymbol{\omega}_{\widetilde{\tau}_k-1}^{(i)} = 1/N,\ i = 1 \dots N$ ;
4 **for** $j = \widetilde{\tau}_k, \dots, k$ **do**
5  $(\boldsymbol{\xi}_j, \boldsymbol{\omega}_j) \leftarrow$ `ParticleFilter(`$\widetilde{\mathcal{W}}_k^j$, $\boldsymbol{\xi}_{j-1}$, $\boldsymbol{\omega}_{j-1}$`)`;
6  $(\boldsymbol{\mu}_j, \mathbf{R}_j) \leftarrow$ `SaveGauss(`$\boldsymbol{\xi}_j$, $\boldsymbol{\omega}_j$`)` ;
7 **endfor**

In many tracking tasks, the Gaussian provides a reasonable approximation to the particle distributions, particularly as it is only used to restart a particle filter. OOSM-GARP thus performs almost as well as the basic re-run particle filter but requires much less memory. However, OOSM-GARP is relatively computationally complex since it reprocesses all the particles for $k-\widetilde{\tau}_k+1$ steps.

## 3.2 Storage Efficient Particle Filter with EKS (SEPF-EKS)

We now provide a brief review of the storage efficient OOSM particle filter from [1], which forms part of our proposed algorithm. We focus on the filter that employs EKS, since it is less computationally demanding but offers equivalent or better accuracy. In order to simplify the presentation, we consider a single OOSM $\mathbf{y}_\tau \equiv \mathbf{y}_\tau^{\mathcal{D}_{\tau,k}} \in \mathcal{Z}_k$.

The SEPF is based on the following weight-update equation:

$$\boldsymbol{\omega}_k^{(i)} \propto p(\mathbf{y}_\tau | \boldsymbol{\xi}_k^{(i)}, \mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) \boldsymbol{\omega}_{k,\bar{\tau}}^{(i)}. \tag{8}$$

Here $\boldsymbol{\omega}_{k,\bar{\tau}}^{(i)}$ and $\boldsymbol{\omega}_k^{(i)}$ denote the weights before and after processing $\mathbf{y}_\tau$. $p(\mathbf{y}_\tau | \boldsymbol{\xi}_k^{(i)}, \mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$ denotes the likelihood function of $\mathbf{y}_\tau$ given all the current particles $\boldsymbol{\xi}_k^{(i)}$, all received measurements in $\mathcal{W}_k^{1:k}$ and the recently received OOSMs in $\mathcal{Z}_k$ except $\mathbf{y}_\tau$, thus $\mathcal{Z}_{k,\bar{\tau}} = \mathcal{Z}_k \setminus \{\mathbf{y}_\tau\}$.

The SEPF estimates this likelihood expression in two stages. First it forms an approximation of $p(\mathbf{x}_\tau | \boldsymbol{\xi}_k^{(i)}, \mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$. This is achieved by applying an augmented-state extended Kalman smoother [7] to update the stored mean and covariance at time $\tau$ to reflect the information from the measurements $\mathcal{W}_k^{\tau+1:k} \cup \mathcal{Z}_{k,\bar{\tau}}$ and the current particle $\boldsymbol{\xi}_k^{(i)}$ (which is treated as a measurement). SEPF then employs an EKF approximation of $p(\mathbf{y}_\tau | \mathbf{x}_\tau)$ to construct an estimate of the likelihood $p(\mathbf{y}_\tau | \mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}, \boldsymbol{\xi}_k^{(i)})$.

SEPF-EKS achieves significant computational savings because it processes the means and covariances in-

stead of all $N$ particles. The computational complexity is less than the re-run particle filters in most cases[1].

The algorithm is vulnerable to highly informative OOSMs. After processing an OOSM that should lead to a major change in the filtering distribution, the effective number of particles (measured by $1/\sum(\boldsymbol{\omega}^{(i)})^2$) can be greatly diminished. This reduces sample diversity in the particle filter and can cause significant performance deterioration.

# 4 Selectively Processing OOSMs

We now propose a simple but effective two-stage algorithm that processes only the informative OOSMs, leading to significantly increased computational efficiency. The first stage of the algorithm estimates the informativeness of an OOSM (we outline two metrics and estimation schemes below) and discards those deemed uninformative. In the second stage, the informative OOSMs are processed by SEPF-EKS. If significant reduction of the effective sample size is detected after application of SEPF-EKS, we choose to apply OOSM-GARP. The approach combines the advantages of OOSM-GARP and SEPF-EKS in order to achieve a satisfactory tradeoff between performance and complexity. We describe the proposed approach `ProcessOOSM-SP` in Algorithm 3.

In this algorithm, the function `CalcMI` is used to estimate the informativeness of a measurement, and it is discussed in more detail below. The thresholds $\gamma_1$ and $\gamma_2$ govern the trade-off between computational complexity and accuracy. The first threshold $\gamma_1$ determines the proportion of OOSMs that are declared uninformative and immediately discarded. The second threshold defines the proportion of informative OOSMs that are processed using `ProcessOOSM-GARP` which reruns the particle filter from the time $\tau$ when the OOSM was measured. In our experiments we observed that `ProcessOOSM-GARP` can be invoked rarely and yet this substantially improves the quality of tracking.

**Algorithm 3**: ProcessOOSM-SP

**Input**: $\mathcal{Z}_k$, $\boldsymbol{\Omega}_k$
EKSfailed = 0;
**for** $\mathbf{y}_\tau \in \mathcal{Z}_k$ **do**
  $I_{\mathbf{y}_\tau} \leftarrow$ `CalcMI(`$\mathbf{y}_\tau$, $\boldsymbol{\mu}_\tau$, $\mathbf{R}_\tau$, $\mathbf{H}_\tau$`)` ;
  **if** $I_{\mathbf{y}_\tau} < \gamma_1$ **then**
    discard $\mathbf{y}_\tau$ ;
  **else**
    $N_{\text{eff}}^{\text{prior}} = 1/\sum_{i=1}^{N}(\boldsymbol{\omega}_k^{(i)})^2$ ;
    $(\boldsymbol{\xi}_k,\boldsymbol{\omega}_k,\boldsymbol{\Omega}_k) \leftarrow$ `ProcessOOSM-EKS(`$\mathbf{y}_\tau$,$\boldsymbol{\xi}_k$,$\boldsymbol{\omega}_k$,$\boldsymbol{\Omega}_k$`)` ;
    $N_{\text{eff}}^{\text{post}} = 1/\sum_{i=1}^{N}(\boldsymbol{\omega}_k^{(i)})^2$ ;
    **if** $N_{\text{eff}}^{\text{post}}/N_{\text{eff}}^{\text{prior}} < \gamma_2$ **then**
      EKSfailed = 1 ;
      **break** ;
  **endif**
**endfor**
**if** *EKSfailed* **then**
  $(\boldsymbol{\xi}_k,\boldsymbol{\omega}_k,\boldsymbol{\Omega}_k) \leftarrow$ `ProcessOOSM-GARP(`$\mathcal{Z}_k$, $\boldsymbol{\Omega}_k$`)` ;

## 4.1 OOSM Selection Rule

We propose two metrics for assessing the "informativeness" of OOSMs, both based on information-theoretic concepts. For the first metric, the OOSM is treated as a random variable $\mathbf{Y}_\tau$, so the metric and decision do not depend on the actual measured value $\mathbf{y}_\tau \equiv \mathbf{y}_\tau^{\mathcal{D}_{\tau,k}} \in \mathcal{Z}_k$. The first metric is the mutual information between the OOSM $\mathbf{Y}_\tau$ and the state $\mathbf{X}_k$, $I(\mathbf{Y}_\tau, \mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$. The second metric is the Kullback-Leibler divergence (KL-divergence) [8] between the distribution at time $k$, conditioned on all measurements except for $\mathbf{y}_\tau$, and the distribution at time $k$ conditioned on all measurements including $\mathbf{y}_\tau$, $D(p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})\|p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_k))$.

Our goal is to estimate the metric quickly and relatively accurately in order to decide about whether to process the OOSM. We therefore employ Gaussian approximations to the distributions of interest and use the extended Kalman filter to calculate their parameters. We now discuss the individual metrics and the procedures used for their estimation.

### 4.1.1 Mutual Information Metric

The mutual information is defined between measurement $\mathbf{Y}_\tau$ and state $\mathbf{X}_k$ as follows:

$$I(\mathbf{Y}_\tau, \mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) = \int \log\left(\frac{p(\mathbf{y}_\tau, \mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})}{p(\mathbf{y}_\tau|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})}\right) \times p(\mathbf{y}_\tau, \mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})\,\mathrm{d}\mathbf{y}_\tau\,\mathrm{d}\mathbf{x}_k$$

Thus to calculate the mutual information based test statistic it is sufficient to know the joint distribution $p(\mathbf{y}_\tau, \mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$. The mutual information can also be expressed in terms of conditional entropies $H$:

$$I(\mathbf{Y}_\tau, \mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) = H(\mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) - H(\mathbf{X}_k|\mathbf{Y}_\tau, \mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}).$$

We choose to approximate the joint distribution by a Gaussian distribution:

$$p(\mathbf{y}_\tau, \mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) \approx \mathcal{N}\left(\begin{pmatrix}\mathbf{x}_k\\ \mathbf{y}_\tau\end{pmatrix}; \begin{pmatrix}\boldsymbol{\mu}_{\mathbf{x}_k}\\ \boldsymbol{\mu}_{\mathbf{y}_\tau}\end{pmatrix}, \begin{pmatrix}\mathbf{R}_{\mathbf{x}_k} & \mathbf{R}_{\mathbf{x}_k\mathbf{y}_\tau}\\ \mathbf{R}_{\mathbf{y}_\tau\mathbf{x}_k} & \mathbf{R}_{\mathbf{y}_\tau}\end{pmatrix}\right)$$

[1] It is difficult to efficiently extend the SEPF-EKS algorithm to process batches of OOSMs, so the computational savings diminish when it is common for multiple OOSMs to arrive in a given timestep. The OOSM-GARP algorithm readily accommodates such batches.

Let us define $\mathbf{R}_{\mathbf{x}_k|\mathbf{y}_\tau} = \mathbf{R}_{\mathbf{x}_k} - \mathbf{R}_{\mathbf{x}_k\mathbf{y}_\tau}\mathbf{R}_{\mathbf{y}_\tau}^{-1}\mathbf{R}_{\mathbf{y}_\tau\mathbf{x}_k}$. Standard Gaussian marginalization and conditioning formula lead to the following relationships:

$$H(\mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) = \frac{1}{2}\log|2\pi e\mathbf{R}_{\mathbf{x}_k}|$$
$$H(\mathbf{X}_k|\mathbf{Y}_\tau, \mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) = \frac{1}{2}\log|2\pi e\mathbf{R}_{\mathbf{x}_k|\mathbf{y}_\tau}|$$
$$I(\mathbf{Y}_\tau, \mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) = \frac{1}{2}\log\frac{|\mathbf{R}_{\mathbf{x}_k}|}{|\mathbf{R}_{\mathbf{x}_k} - \mathbf{R}_{\mathbf{x}_k\mathbf{y}_\tau}\mathbf{R}_{\mathbf{y}_\tau}^{-1}\mathbf{R}_{\mathbf{y}_\tau\mathbf{x}_k}|}$$

We can devise the following technique for estimating $I(\mathbf{Y}_\tau, \mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$. We assume that the measurement equation at time $\tau$ can be reasonably accurately linearized around the estimate of the state. Defining $\mathbf{H}_\tau = \frac{\partial}{\partial x}h_\tau(\mathbf{x})|_{\mathbf{x}=\boldsymbol{\mu}_{\mathbf{x}_\tau}}$, this implies that $\mathbf{y}_\tau \approx \mathbf{H}_\tau\mathbf{x}_\tau + \mathbf{s}_\tau$. See [9] for further discussion about this assumption.

Commencing with the saved distribution at time $\tau$, $p(\mathbf{x}_\tau|\mathcal{W}_k^{1:\tau}) \approx \mathcal{N}(\mathbf{x}_\tau; \boldsymbol{\mu}_\tau, \mathbf{R}_\tau)$ and using the linearization assumption, we can calculate the Gaussian approximation of the joint distribution at time $\tau$:

$$p(\mathbf{y}_\tau, \mathbf{x}_\tau|\mathcal{W}_k^{1:\tau}) \approx \mathcal{N}(\begin{pmatrix}\mathbf{x}_\tau\\ \mathbf{y}_\tau\end{pmatrix}; \begin{pmatrix}\boldsymbol{\mu}_{\mathbf{x}_\tau}\\ \boldsymbol{\mu}_{\mathbf{y}_\tau}\end{pmatrix}, \begin{pmatrix}\mathbf{R}_{\mathbf{x}_\tau} & \mathbf{R}_{x_\tau\mathbf{y}_\tau}\\ \mathbf{R}_{\mathbf{y}_\tau\mathbf{x}_\tau} & \mathbf{R}_{\mathbf{y}_\tau}\end{pmatrix}),$$

where we set $\mu_{\mathbf{x}_\tau} = \boldsymbol{\mu}_\tau$, $\boldsymbol{\mu}_{\mathbf{y}_\tau} = h_\tau(\boldsymbol{\mu}_\tau)$; $\mathbf{R}_{\mathbf{x}_\tau} = \mathbf{R}_\tau$, $\mathbf{R}_{\mathbf{y}_\tau} = \mathbf{H}_\tau\mathbf{R}_\tau\mathbf{H}_\tau^T + \mathbf{Q}_\tau$, $\mathbf{R}_{\mathbf{x}_\tau\mathbf{y}_\tau} = \mathbf{R}_\tau\mathbf{H}_\tau^T$, $\mathbf{R}_{\mathbf{y}_\tau\mathbf{x}_\tau} = \mathbf{R}_{\mathbf{x}_\tau\mathbf{y}_\tau}^T$.

We now apply a forward EKF recursion, augmenting the state $\mathbf{x}$ with the measurement $\mathbf{y}_\tau$, denoted by $\mathbf{z}$ with its covariance $\mathbf{P}$. The EKF recursion consists of a prediction step:

$$\mathbf{z}_{m+1|m} = \begin{pmatrix}\boldsymbol{\mu}_{\mathbf{x}_{m+1|m}}\\ \boldsymbol{\mu}_{\mathbf{y}_\tau}\end{pmatrix} = \begin{pmatrix}f_m(\boldsymbol{\mu}_{\mathbf{x}_m})\\ \boldsymbol{\mu}_{\mathbf{y}_\tau}\end{pmatrix} \quad (9)$$

$$\mathbf{P}_{m+1|m} = \begin{pmatrix}\mathbf{F}_m & 0\\ 0 & \mathbf{I}\end{pmatrix}\begin{pmatrix}\mathbf{R}_{\mathbf{x}_m} & \mathbf{R}_{\mathbf{x}_m\mathbf{y}_\tau}\\ \mathbf{R}_{\mathbf{y}_\tau\mathbf{x}_m} & \mathbf{R}_{\mathbf{y}_\tau}\end{pmatrix}\begin{pmatrix}\mathbf{F}_m^T & 0\\ 0 & \mathbf{I}\end{pmatrix} + \begin{pmatrix}\mathbf{V}_{m+1|m} & 0\\ 0 & 0\end{pmatrix} \quad (10)$$

and an update step:

$$\mathbf{r}_{m+1} = \widetilde{\mathbf{y}}_{m+1} - \widetilde{h}_{m+1}(\boldsymbol{\mu}_{\mathbf{x}_{m+1|m}}) \quad (11)$$

$$\mathbf{K}_{m+1} = \mathbf{P}_{m+1|m}\widetilde{\mathbf{H}}_{m+1}^T(\widetilde{\mathbf{H}}_{m+1}\mathbf{P}_{m+1|m}\widetilde{\mathbf{H}}_{m+1}^T + \widetilde{\mathbf{Q}}_{m+1})^{-1} \quad (12)$$

$$\mathbf{z}_{m+1} = \mathbf{z}_{m+1|m} + \mathbf{K}_{m+1}\mathbf{r}_{m+1} \quad (13)$$

$$\mathbf{P}_{m+1} = (\mathbf{I} - \mathbf{K}_{m+1}\mathbf{H}_{m+1})\mathbf{P}_{m+1|m} \quad (14)$$

Note that

$$\widetilde{\mathbf{y}}_{m+1} = \begin{pmatrix}\cdots\\ \mathbf{y}_{m+1}^j\\ \cdots\end{pmatrix}$$

is the vector of varying dimensionality that contains stacked measurements from time $m+1$ available at time $k$, i.e. $\widetilde{\mathbf{y}}_{m+1}$ contains $\mathbf{y}_{m+1}^j$ if $\mathbf{y}_{m+1}^j \in \mathcal{W}_k^{m+1} \cup \mathcal{Z}_k$. Likewise, $\widetilde{h}_{m+1}(\cdot)$ is the corresponding non-linear vector function defined similarly to $\widetilde{\mathbf{y}}_{m+1}$. However, $\widetilde{\mathbf{H}}_{m+1} = \frac{\partial}{\partial x}h_{m+1}(x)|_{x=\mathbf{z}_{m+1|m}}$ is the linearization with augmented state. $\widetilde{\mathbf{Q}}_{m+1}$ is the block-diagonal matrix which describes noise terms corresponding to components $\mathbf{y}_{m+1}^j$ of vector $\widetilde{\mathbf{y}}_{m+1}$.

Repeated application of the recursion permits estimation of the joint distribution $p(\mathbf{y}_\tau, \mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$ and this allows us to estimate the mutual information metric $I(\mathbf{Y}_\tau, \mathbf{X}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$ using the expressions above.

#### 4.1.2 KL-divergence metric

The KL-divergence between $p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})$ and $p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_k)$ is calculated using the following formula:

$$D(p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})\|p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_k)) = \frac{1}{2}\log\left(\frac{|\widehat{\mathbf{R}}_k|}{|\mathbf{R}_k|} + \mathrm{tr}(\widehat{\mathbf{R}}_k^{-1}\mathbf{R}_k) + (\widehat{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k)^T\widehat{\mathbf{R}}_k^{-1}(\widehat{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k) - d_{\mathbf{x}}\right) \quad (15)$$

where $d_{\mathbf{x}}$ is the dimensionality of the state $x_k$. In Algorithm 3 we use the symmetrized KL-divergence:

$$I_{\mathbf{y}_\tau} = \big(D(p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}})\|p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_k)) + D(p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_k)\|p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}))\big)/2 \quad (16)$$

We estimate the KL-divergence using Gaussian approximations: $p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_{k,\bar{\tau}}) \approx \mathcal{N}(\mathbf{x}_k, \boldsymbol{\mu}_k, \mathbf{R}_k)$ and $p(\mathbf{x}_k|\mathcal{W}_k^{1:k}, \mathcal{Z}_k) \approx \mathcal{N}(\mathbf{x}_k, \widehat{\boldsymbol{\mu}}_k, \widehat{\mathbf{R}}_k)$. Both distributions are obtained by applying forward EKF recursions starting from time $\tau$. To obtain the latter distribution we first apply a measurement update step at time $\tau$ and then calculate standard EKF recursion. The former distribution is obtained by calculating the standard EKF recursion and excluding the measurement $\mathbf{y}_\tau$.

## 5 Numerical Experiments

In our simulations we consider a two-dimensional scenario with a single target that makes a clockwise coordinated turn of radius $500m$ with a constant speed $200km/h$. It starts in y-direction with initial position $[-500m, 500m]$ and is tracked for 40 seconds.

The target motion is modeled in the filters by the nearly coordinated turn model with unknown constant turn rate and cartesian velocity. The state of the target is given as $\mathbf{x}_k = [p_k^x, p_k^y, v_k^x, v_k^y, \omega_k]^T$, where $p, v$ and $\omega$ denote the position, velocity and turn rate respectively. The dynamic model for the coordinated turn model $f_{k+1|k}(.)$ is

$$\mathbf{x}_{k+1} = \begin{pmatrix}1 & 0 & \frac{\sin(\omega_k)}{\omega_k} & \frac{\cos(\omega_k)-1}{\omega_k} & 0\\ 0 & 1 & \frac{1-\cos(\omega_k)}{\omega_k} & \frac{\sin(\omega_k)}{\omega_k} & 0\\ 0 & 0 & \cos(\omega_k) & -\sin(\omega_k) & 0\\ 0 & 0 & \sin(\omega_k) & \cos(\omega_k) & 0\\ 0 & 0 & 0 & 0 & 1\end{pmatrix}\mathbf{x}_k + \mathbf{v}_{k+1|k}$$

where $\mathbf{v}_{k+1|k}$ is Gaussian process noise, $\mathbf{v}_{k+1|k} \sim \mathcal{N}(0, \mathbf{V}_{k+1|k})$, $\mathbf{V}_{k+1|k} = \text{diag}([30^2, 30^2, 10^2, 10^2, 0.1^2])$, the sampling period is 1 second. We assume that all of the filters initially know little about the state of the target and therefore they are initialized with the state value $\mathbf{x}_0 = [0, 0, 0, 0, 0]^T$ and a large covariance $\mathbf{R}_0 = \text{diag}([1000^2, 1000^2, 30^2, 30^2, 0.1^2])$.

There are three sensors S1, S2 and S3 sending bearing-only measurements of the target to a common fusion centre. The sensor locations are $[S_1^x, S_1^y] = [-200, 0]$, $[S_2^x, S_2^y] = [200, 0]$, $[S_3^x, S_3^y] = [-750, 750]$ and the bearings-only measurement function is

$$h_k(\mathbf{x}_k) = \arctan\left(\frac{p_k^y - S_j^y}{p_k^x - S_j^x}\right) \; j = 1, 2, 3 \qquad (17)$$

The measurements from the sensors are corrupted with additive independent Gaussian noises with zero mean and standard deviation $\sigma_s = 0.05$. All sensors are assumed to have communication issues leading to OOSMs. An OOSM arrives at the fusion centre from a given sensor with probability $p_{osm}$ and delay $d$. The delay $d$ is uniformly distributed in the interval $[0, 5]$. The probability $p_{osm}$ is set to 0.7, which characterizes the reliability of OOSM delivery (a portion of the OOSMs are lost on the way to the fusion centre).

## 5.1 Benchmarked Filters

We have implemented six different particle filters, all based on the Sampling Importance Resampling (SIR) filtering paradigm [10]. The prior distribution is used as the importance function. The filters were implemented in Matlab and the code was highly optimized.

*PFall*: collects all measurements from all active sensors (no OOSMs). This is an idealized filter that provides a performance benchmark.

*PFmis*: discards all OOSMs and therefore only processes the measurements with zero delay.

*PFEKS*: Storage efficient particle filter using EKS smoothing as described in [1] (see Section 3.2).

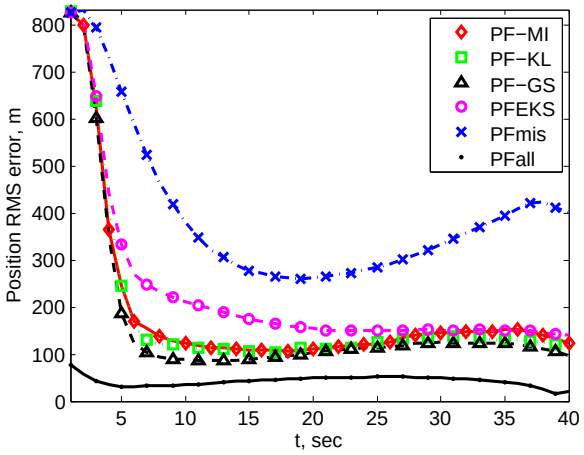
*PF-GS*: The OOSM-GARP algorithm described in Algorithm 2.

*PF-MI*: Selective OOSM processing based on the mutual information metric.

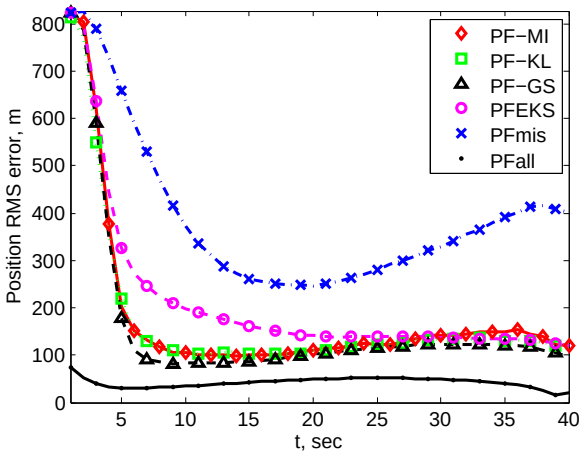
*PF-KL*: Selective OOSM processing based on the KL-divergence metric.

We use the root mean-squared (RMS) position error to compare the performances of particle filters. Let $(p_k^x, p_k^y)$ and $(\hat{p}_{k,i}^x, \hat{p}_{k,i}^y)$ denote the true and estimated target positions at time step $k$ for the $i$-th of $M$ Monte-Carlo runs. The RMS position error at $k$ is calculated as

$$\text{RMS}_k = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{p}_{k,i}^x - p_k^x)^2 + (\hat{p}_{k,i}^y - p_k^y)^2} \qquad (18)$$




(a) N=2000




(b) N=5000

Figure 1: Tracking performance of the particle filters as a function of time using RMS error as a performance metric. The curves show the means of 1000 Monte-Carlo trials. (a) Filters use 2000 particles; (b) Filters use 5000 particles.

## 5.2 Results and Discussion

The computational complexity versus accuracy trade-off can be tuned by adjusting the thresholds. We illustrate this in our experiment, where we vary the computational complexity of the proposed algorithms *PF-KL* and *PF-MI* by varying the respective thresholds and plot the RMS vs. complexity curve measured in MATLAB. $\gamma_1 = 0 : 0.2 : 1.8$ for *PF-MI* and $\gamma_1 = 0 : 0.5 : 4.5$ for *PF-KL*. $\gamma_2 = 2.5\%$ for both of them. These results are reported in Fig. 3. In this figure we show the relationship between complexity and performance for the proposed algorithms *PF-KL* and *PF-MI* with ten values of the first stage threshold $\gamma_1$ and results of 10 simulations for other algorithms. Each simulation involves 1000 Monte Carlo runs. We compare the performance of all particle filters when they use 2000

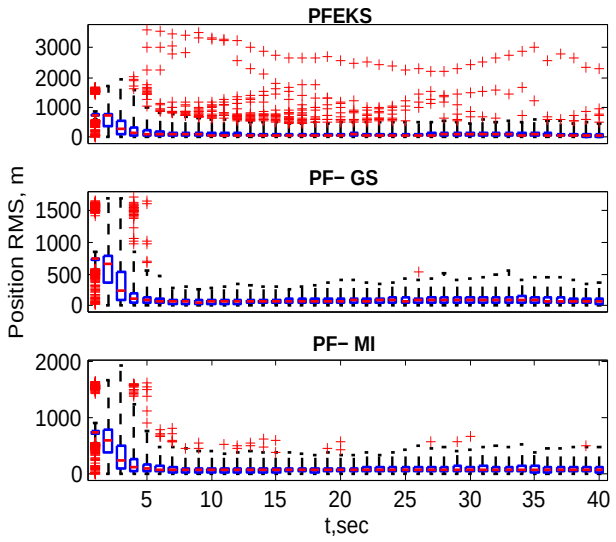



Figure 2: Errorbars showing the variation of position RMS for *PFEKS*, *PF-GS* and *PF-MI*, when they use 5000 particles. The box has lines at the lower quartile, median(red line), and upper quartile values. Outliers (red '+') are values beyond the range of 5 times the interquartile range from the ends of the box.

and 5000 particles. When the thresholds are chosen so that the selective processing filter has the same computational complexity as the storage efficient particle filter of [1], it achieves significantly better tracking performance. Alternatively, for the same fixed RMS error performance, the selective processing algorithm reduces the computation time by approximately 50%.

From Fig. 3, we can see that the following thresholds optimize the trade-off between complexity and accuracy: $\gamma_1 = 0.4$ for *PF-MI* and $\gamma_1 = 0.5$ for *PF-KL*. In Fig. 1, we plot the RMS position performance for $40s$ of the algorithms with these settings. From the results, the number of individual OOSMs to be processed by the EKS after the first threshold $\gamma_1$ is 30.66% and 30.88% for *PF-MI* and 35.67% and 35.92% for *PF-KL* with 2000 and 5000 particles respectively. After the second threshold the number of most informative OOSMs processed by rerunning the particle filter is 1.42% and 1.34% for *PF-MI* and 3.13% and 2.86% for *PF-KL* with 2000 and 5000 particles respectively. Errorbar plots of RMS performance of *PFEKS*, *PF-GS* and *PF-MI* with 5000 particles are shown in Fig. 2, which indicates that the performance of *PFEKS* is not as stable as *PF-GS* and *PF-MI*. Despite processing only a relatively small fraction of the OOSMs, the proposed algorithm performs almost as well as the much more complex OOSM-GARP algorithm (*PF-GS*). The calculation of the selection criterion has minimal overhead, so discarding the uninformative measurements results in significant computational savings.

# 6 Conclusions

This paper presents a computationally efficient algorithm for delay-tolerant particle filtering that has limited memory requirements. By identifying and discarding the uninformative delayed measurements, the algorithm reduces the computational requirements. By processing the most informative measurements with a rerun particle filter, the algorithm achieves better tracking performance than the storage efficient particle filter of [1]. In future work we plan to develop an adaptive process that modifies the thresholds during operation until a satisfactory balance between computational load and accuracy is achieved. We will also explore whether the fusion centre can provide feedback to the sensor nodes so that they can locally assess measurement informativeness. This would allow sensor nodes to avoid unnecessary energy expenditure by discarding uninformative measurements prior to transmission.

# References

[1] U. Orguner and F. Gustafsson, "Storage efficient particle filters for the out of sequence measurement problem," in *Proc. Int. Conf. Information Fusion*, (Cologne, Germany), July 2008.

[2] Y. Bar-Shalom, H. Chen, and M. Mallick, "One-step solution for the multistep out-of-sequence-measurement problem in tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, pp. 27–37, Jan. 2004.

[3] M. Orton and W. Fitzgerald, "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Trans. Signal Process.*, vol. 50, pp. 216–223, Feb. 2002.

[4] M. Orton and A. Marrs, "Particle filters for tracking with out-of-sequence measurements," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, pp. 693–702, Feb. 2005.

[5] M. Mallick, T. Kirubarajan, and S. Arulampalam, "Out-of-sequence measurement processing for tracking ground target using particle filters," in *Proc. IEEE Aerospace Conf.*, vol. 4, pp. 1809–1818, Mar. 2002.

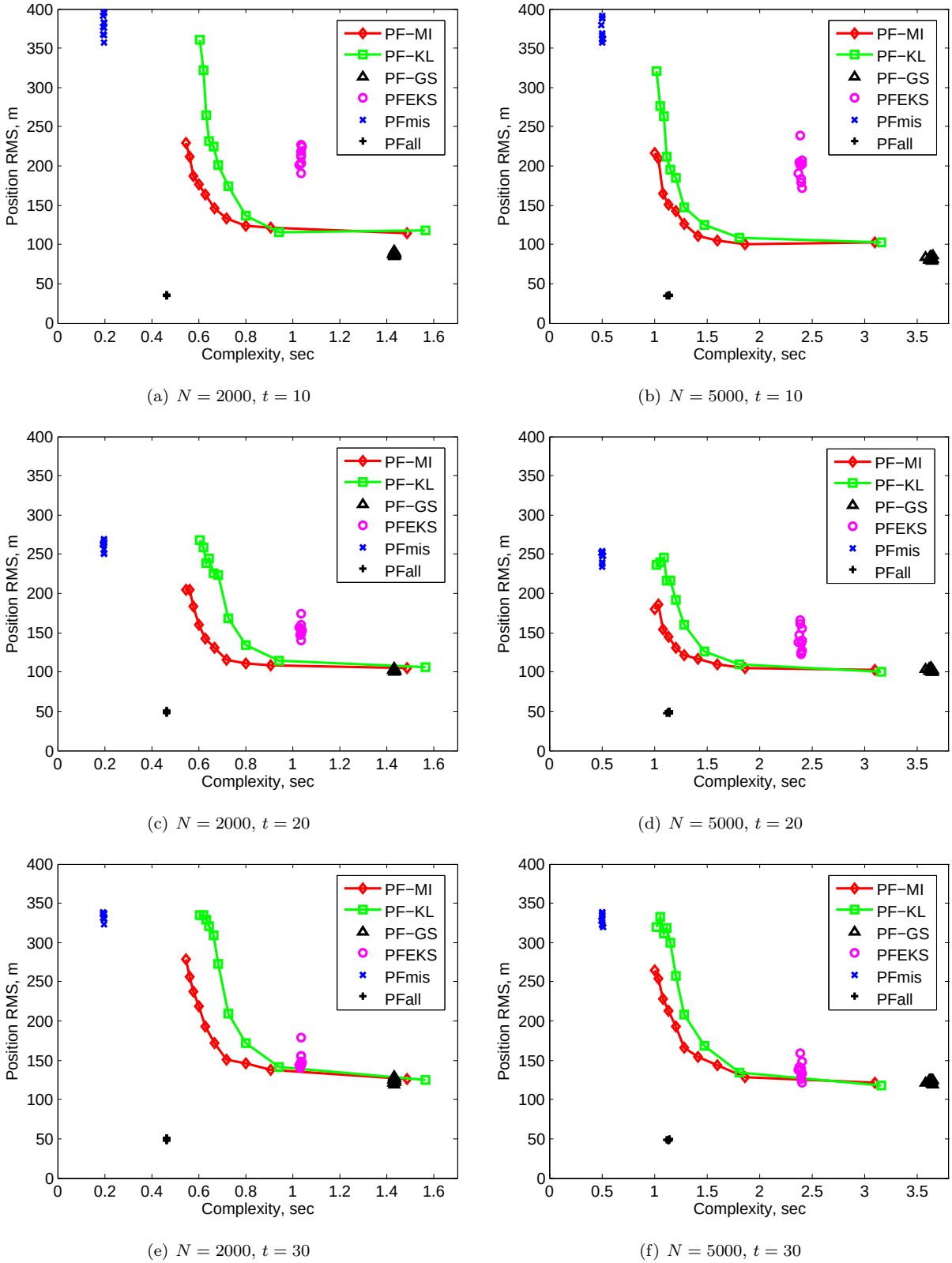
[6] P. Fearnhead, D. Wyncoll, and T. Tawn, "A sequential smoothing algorithm with linear computational cost," Tech. Rep. 8844, Lancaster University, May 2008.

[7] K. K. Biswas and A. Mahalana, "Suboptimal algorithms for nonlinear smoothing," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 9, pp. 529–534, Apr. 1973.

[8] S. Kullback and R. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 1317–1339, 1951.

[9] J. L. Williams, J. W. Fisher, and A. Willsky, "Approximate dynamic programming for communication-constrained sensor network management," *IEEE Trans. Signal Process.*, vol. 55, pp. 3995–4003, Aug. 2007.

[10] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *Radar and Signal Processing, IEEE Proceedings F*, vol. 140, pp. 107–113, Apr. 1993.




(a) $N = 2000$, $t = 10$

(b) $N = 5000$, $t = 10$

(c) $N = 2000$, $t = 20$

(d) $N = 5000$, $t = 20$

(e) $N = 2000$, $t = 30$

(f) $N = 5000$, $t = 30$

Figure 3: RMS vs Complexity from 10 simulations with different $\gamma_1$ for *PF-MI*(from 0 to 1.8) and *PF-KL*(from 0 to 4.5). Each simulation shows the average of 1000 MC runs.We select three timesteps, $t = 10, 20, 30$ for filters with 2000 and 5000 particles. The complexity is measured by running time for tracking 40s of each filter. The results are run on a Dell laptop with Genuine Intel(R) CPU T2400 1.83GHz, 0.99GB RAM and Win-XP OS.