Feedback Particle Filter: Application and Evaluation

Berntorp, K.

Abstract

Recent research has provided several new methods for avoiding degeneracy in particle filters. These methods implement Bayes rule using a continuous transition between prior and posterior. The feedback particle filter (FPF) is one of them. The FPF uses feedback gains to adjust each particle according to the measurement, which is in contrast to conventional particle filters based on importance sampling. The gains are found as solutions to partial differential equations. This paper contains an evaluation of the FPF on two highly nonlinear estimation problems. The FPF is compared with conventional particle filters and the unscented Kalman filter. Sensitivity to the choice of the gains is discussed and illustrated. We demonstrate that with a sensible approximation of the exact gain, the FPF can decrease tracking errors with more than one magnitude while significantly improving the quality of the particle distribution.

Feedback Particle Filter: Application and Evaluation

Karl Berntorp

Mitsubishi Electric Research Laboratories

Cambridge, MA 02139

E-mail: karl.o.berntorp@ieee.org

Abstract—**Recent research has provided several new methods for avoiding degeneracy in particle filters. These methods implement Bayes' rule using a continuous transition between prior and posterior. The feedback particle filter (FPF) is one of them. The FPF uses feedback gains to adjust each particle according to the measurement, which is in contrast to conventional particle filters based on importance sampling. The gains are found as solutions to partial differential equations. This paper contains an evaluation of the FPF on two highly nonlinear estimation problems. The FPF is compared with conventional particle filters and the unscented Kalman filter. Sensitivity to the choice of the gains is discussed and illustrated. We demonstrate that with a sensible approximation of the exact gain, the FPF can decrease tracking errors with more than one magnitude while significantly improving the quality of the particle distribution.**

I. INTRODUCTION

The aim in continuous-discrete time Bayesian filtering is to estimate the posterior filtering density $p(\boldsymbol{x}(t)|\mathcal{Y}_k)$, or at least the relevant moments, at each time $t \in \mathbb{R}$. Here, $\mathcal{Y}_k := \{\boldsymbol{y}(t_0), \ldots, \boldsymbol{y}(t_k)\}$ denotes the set of measurements, obtained at discrete time steps. Many models in continuous-discrete time estimation are written on state-space form as

$$\begin{aligned} \mathrm{d}\boldsymbol{x}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), t)\mathrm{d}t + \mathrm{d}\boldsymbol{\beta}(t), \\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{e}_k, \end{aligned} \tag{1}$$

where $\boldsymbol{x} := \boldsymbol{x}(t) \in \mathbb{R}^n$ is the state; $\boldsymbol{y}_k := \boldsymbol{y}(t_k) \in \mathbb{R}^m$ is the discrete-time measurement at time t_k; $\boldsymbol{f}$ and $\boldsymbol{h}$ are the drift and measurement function, respectively; and $\boldsymbol{\beta}$ and $\boldsymbol{e}$ are process and measurement noise, respectively. Sometimes a discretized counterpart to (1) is used, resulting in (with different $\boldsymbol{f}$)

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{f}(\boldsymbol{x}_k, k) + \boldsymbol{w}_k, & (2\mathrm{a}) \\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{e}_k. & (2\mathrm{b}) \end{aligned}$$

In the following, $\boldsymbol{f}_k^i := \boldsymbol{f}(\boldsymbol{x}_k^i, k)$, and both process and measurement noise are assumed Gaussian distributed with zero mean and covariance matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, where $\boldsymbol{R}$ is diagonal.

Extended Kalman filters (EKFs) and unscented Kalman filters (UKFs) [1], [2] are popular estimation methods for nonlinear systems. When the dynamics is highly nonlinear and/or the posterior is not well represented by the first few moments, the EKF and UKF may perform poorly. In these cases, particle filters (PFs) are often preferred. PFs [3], [4] have successfully estimated the states of (1) and (2) in many applications, see [5]–[11] for some examples. PFs solve the Bayesian recursions using a set of weighted particles, which are propagated forward in time. One problem with PFs is the inevitable particle degeneracy [12] (i.e., only a few particles,

or even one, have nonzero weight), caused by the way Bayes' rule is implemented [13]. Degeneracy leads to decreased performance, or even filter divergence. To mitigate degeneracy, PFs use resampling, which consists of replacing particles having low likelihood with more probable particles. The resampling step makes PFs practically useful, but introduces other negative effects, such as sample impoverishment and increased variance [4]. Note that although the noise sources are often assumed, or approximated as, Gaussian additive with zero mean, PFs apply to more general noise distributions.

Since a number of years there exist variants of the PF that remove degeneracy and therefore also the need for resampling. Instead of implementing Bayes' rule in one step, these algorithms all have in common that they rely on a gradual transition from the prior to the posterior. As a consequence, resampling is avoided and importance sampling is not used. In [14], a framework for gradual transition from prior to posterior was introduced—see also [15], [16]. The particle-flow filter has been introduced and improved in a series of papers, see, for example, [17]–[20]. A related filter is the feedback particle filter (FPF) [21]–[25]. The FPF applies a feedback structure to each particle. It can be seen as a generalization of the linear-regression filters in literature, such as the UKF or the smart-sampling Kalman filter [26]. The feedback gains are determined as the solutions to optimization problems, where the Kullback-Leibler divergence serves as cost function [25].

This paper presents application and evaluation of the FPF using two classical target-tracking problems—the vehicle reentry problem and the planar two-body problem. These problems have in common that both the dynamics and measurement equations are highly nonlinear. We consider both periodic and infrequent measurements. The FPF is implemented with two different approximations of the optimal feedback gains. Comparisons are made with the UKF, the bootstrap particle filter (PF), the Rao-Blackwellized particle filter (RBPF) [27], and a particle filter using optimal sampling with linearized likelihood [4], [8]. The purpose of this paper is to fill a void in terms of thorough evaluations of the FPF on relevant and challenging applications. The aim is to provide insight for when the FPF outperforms more traditional approaches and when it does not. There exist many variants of the PF, but we here focus on two of the most commonly used. There has been a few comparisons between PFs and the FPF before (e.g., [28]), but we go more into depth. In particular, we quantify and illustrate how the choice of feedback gains can drastically affect performance, which has not been discussed before.

II. Background on Particle Filters

Sampling-based approaches typically have in common that they propagate samples (hypotheses) of the states, but the samples can either be chosen deterministically or randomly. We will next go through the conventional PF (discrete-time case, but continuous versions exist) and the FPF.

One key ingredient in the derivation of recursive state estimators using probability density functions is Bayes' theorem (or Bayes' rule): for two variables x and y,

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}. \tag{3}$$

The posterior $p(\boldsymbol{x}_k|\mathcal{Y}_k)$ can be rewritten using (3), which gives

$$p(\boldsymbol{x}_k|\mathcal{Y}_k) \propto p(\boldsymbol{x}_k|\boldsymbol{y}_k, \mathcal{Y}_{k-1}) = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\mathcal{Y}_{k-1})}{p(\boldsymbol{y}_k|\mathcal{Y}_{k-1})}. \tag{4}$$

Thus, in (4), the measurement $\boldsymbol{y}_k$ is used for updating the prior density to obtain the posterior probability density function.

A. Particle Filtering

Assume that N particles (hypotheses) $\{\boldsymbol{x}_k^i\}_{i=1}^N$ are sampled from the one-step prior as

$$\boldsymbol{x}_k^i \sim p(\boldsymbol{x}_k|\mathcal{Y}_{k-1}). \tag{5}$$

The PF [3], [4] then approximates this density with

$$p(\boldsymbol{x}_k|\mathcal{Y}_{k-1}) \approx \hat{p}(\boldsymbol{x}_k|\mathcal{Y}_{k-1}) = \sum_{i=1}^N w_{k|k-1}^i \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i). \tag{6}$$

Here, $\delta(\cdot)$ is the Dirac delta function. The importance weights $\{w_{k|k-1}^i\}_{i=1}^N$ indicate how likely each particle is, and fulfill

$$\sum_{i=1}^N w_{k|k-1}^i = 1, \quad w_{k|k-1}^i \geq 0, \quad \forall i \in \{1, \ldots, N\}.$$

Often it is impossible to sample from (5) directly. Instead, let $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \mathcal{Y}_k)$ be a *proposal density* (the importance density) from which it is possible to sample. Using the importance density, the one-step prior is rewritten as

$$p(\boldsymbol{x}_k|\mathcal{Y}_{k-1}) = \int q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \mathcal{Y}_k) \frac{p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})}{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \mathcal{Y}_k)}$$
$$\cdot p(\boldsymbol{x}_{k-1}|\mathcal{Y}_{k-1})\, \mathrm{d}\boldsymbol{x}_{k-1}. \tag{6}$$

By drawing N samples from the proposal density, the integral in (6) approximates to a sum according to

$$p(\boldsymbol{x}_k|\mathcal{Y}_{k-1}) \approx \sum_{i=1}^N w_{k|k-1}^i \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i). \tag{7}$$

The weights $w_{k|k-1}^i$ are computed as

$$w_{k|k-1}^i \propto \frac{p(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i)}{q(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i, \mathcal{Y}_k)} p(\boldsymbol{x}_{k-1}^i|\mathcal{Y}_{k-1}). \tag{8}$$

By inserting (7) into (4), the posterior is obtained as

$$p(\boldsymbol{x}_k|\mathcal{Y}_k) = \sum_{i=1}^N \underbrace{p(\boldsymbol{y}_k|\boldsymbol{x}_k^i) w_{k|k-1}^i}_{w_k^i} \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i),$$

One obvious choice of proposal density is

$$q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \mathcal{Y}_k) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i). \tag{9}$$

With this choice, the weight update equation reads

$$w_k^i \propto p(\boldsymbol{y}_k|\boldsymbol{x}_k^i) w_{k-1}^i, \tag{10}$$

since $w_{k|k-1} = w_{k-1}^i$ in (8). With the proposal (9), the PF is called the bootstrap PF. It was introduced in [3]. This version is, by far, the most common in the PF literature. Another option is to choose the proposal

$$q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \mathcal{Y}_k) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \mathcal{Y}_k), \tag{11}$$

which leads to the weight update

$$w_k^i \propto p(\boldsymbol{y}_k|\boldsymbol{x}_{k-1}^i) w_{k-1}^i. \tag{12}$$

Eq. (12) implies that the weight is independent of the sample $\boldsymbol{x}_k^i$, and is optimal in the sense that it maximizes the effective number of samples [8] (all other alternatives will lead to increased variance of the weights). The proposal (11) is generally difficult to sample from exactly. However, for a linear, Gaussian measurement relation (2b) in the form

$$\boldsymbol{y}_k = \boldsymbol{H}\boldsymbol{x}_k + e_k,$$

the expression is analytic. For a nonlinear measurement relation a linearized version can be used, leading to

$$q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \boldsymbol{y}_k) = \mathcal{N}\left(\boldsymbol{x}_k|\bar{\boldsymbol{x}}_k^i, (\boldsymbol{\Sigma}^i)^{-1}\right) \tag{13}$$

where

$$\bar{\boldsymbol{x}}_k^i = \boldsymbol{f}_{k-1}^i + \boldsymbol{L}_k^i(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k^i),$$
$$\boldsymbol{\Sigma}^i = \left((\boldsymbol{H}_k^i)^\mathrm{T} \boldsymbol{R}^{-1} \boldsymbol{H}_k^i + \boldsymbol{Q}^{-1}\right)^{-1},$$
$$\boldsymbol{L}_k^i = \boldsymbol{Q}_k(\boldsymbol{H}_k^i)^\mathrm{T}(\boldsymbol{H}_k^i \boldsymbol{Q}(\boldsymbol{H}_k^i)^\mathrm{T} + \boldsymbol{R})^{-1},$$
$$\boldsymbol{H}^i = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{f}_{k-1}^i},$$

and the measurement likelihood is approximated as

$$p(\boldsymbol{y}_k|\boldsymbol{x}_{k-1}^i) = \mathcal{N}\left(\boldsymbol{y}_k|\hat{\boldsymbol{y}}_k^i, \boldsymbol{H}_k^i \boldsymbol{Q}(\boldsymbol{H}_k^i)^\mathrm{T} + \boldsymbol{R}\right).$$

Irrespective of whether (9) or (11) is used, the PF includes a necessary resampling step: when the effective number of samples N_eff becomes too small (in this paper when $N_\mathrm{eff} \leq 2N/3$), N particles are chosen (with replacement), where the probability of choosing $\boldsymbol{x}_k^i$ is w_k^i. In this paper, we will use both (9) and (13) for comparison with the FPF.

B. Feedback Particle Filter

A brief recapitulation of the continuous-discrete time FPF follows next—see [22] for a more complete treatment. The FPF approximates the posterior p with N unweighted samples, or particles, $\boldsymbol{x}^i$ as

$$p(\boldsymbol{x}|\mathcal{Y}_k) \approx \hat{p}(\boldsymbol{x}|\mathcal{Y}_k) = \frac{1}{N} \sum_{i=1}^N \delta(\boldsymbol{x} - \boldsymbol{x}^i).$$

Unlike conventional PFs, the FPF models the measurement update of the ith particle as a controlled system (Fig. 1):

$$\mathrm{d}\boldsymbol{x}^i = \boldsymbol{f}^i \mathrm{d}t + \mathrm{d}\boldsymbol{\beta}^i + \boldsymbol{U}_k^i.$$

To incorporate the new measurement $\boldsymbol{y}_k$, a particle flow $\boldsymbol{S}_k^i := \boldsymbol{S}_k^i(\lambda)$ and a control input $\boldsymbol{U}_k^i := \boldsymbol{U}_k^i(\lambda)$ are introduced:

$$\frac{\mathrm{d}\boldsymbol{S}_k^i}{\mathrm{d}\lambda} = \boldsymbol{U}_k^i, \tag{14}$$

where $\lambda \in [0, 1]$ is the *pseudo-time*. $\boldsymbol{S}_k^i$ is initialized ($\lambda = 0$) to equal the ith particle before the measurement update and $\boldsymbol{U}_k^i$ is designed such that the distribution generated by $\{\boldsymbol{S}_k^i\}_{i=1}^N$ approximates the posterior at $\lambda = 1$, see Fig. 2. This leads to a simulation-based implementation of Bayes' rule, unlike traditional PFs. This approach is made possible by a log-homotopy transformation, which transforms the discrete-time Bayesian measurement update to a continuously evolving process. In (14), $\boldsymbol{U}_k^i$ has the form

$$\boldsymbol{U}_k^i(\lambda) = \boldsymbol{K}_k^i \boldsymbol{I}_k^i + \frac{1}{2}\boldsymbol{\Omega}_k^i,$$

where $\boldsymbol{K}_k^i = \begin{bmatrix} \boldsymbol{k}_1^i & \cdots & \boldsymbol{k}_m^i \end{bmatrix} := \boldsymbol{K}^i(\boldsymbol{S}_k^i, \lambda) \in \mathbb{R}^{n \times m}$ is the feedback gain, $\boldsymbol{\Omega}_k^i := \boldsymbol{\Omega}(\boldsymbol{S}_k^i, \lambda)$ is the *Wong-Zakai correction term* [29], and $\boldsymbol{I}^i$ is the innovation error, which equals

$$\boldsymbol{I}_k^i = \boldsymbol{y}_k - \frac{1}{2}(\boldsymbol{h}^i + \hat{\boldsymbol{h}}), \tag{15}$$

where

$$\hat{\boldsymbol{h}} = \mathbb{E}(\boldsymbol{h}(\boldsymbol{x})) := \int \boldsymbol{h}(\boldsymbol{x})p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \approx \frac{1}{N}\sum_{i=1}^N \boldsymbol{h}(\boldsymbol{x}^i).$$

The innovation process (15) includes the predicted measurement of particle i and the average of all particles. The control synthesis is done by solving an optimal-control problem, with the Kullback-Leibler divergence as the cost [25]. By defining $(\boldsymbol{\phi}^i)^{\mathrm{T}} = \begin{bmatrix} \phi_1 & \cdots & \phi_m \end{bmatrix} := \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}, \lambda)$, $\boldsymbol{k}_j := \nabla\phi_j(\boldsymbol{x}, \lambda)$ is a solution to the partial differential equation

$$\nabla \cdot (p\nabla\phi_j) = -\frac{1}{R_{jj}}(h_j - \hat{h}_j)p, \tag{16}$$

for $j = 1, \ldots, m$ and each time t_k [24] (similarly for $\boldsymbol{\Omega}_k$), and where R_{jj} is the variance of the jth element in $\boldsymbol{y}_k$. In analogy with the PF, the FPF is consistent (i.e., $\hat{p}(\boldsymbol{x}|\mathcal{Y}_t) = p(\boldsymbol{x}|\mathcal{Y}_t)$ for all t given correct initial distribution) when N is inifinitely large. However, the simulation-based update in the FPF removes the need for resampling, which is present in PFs. The consistency result for the FPF only holds for an exact expression of the feedback gain. In fact, the main difficulty in the implementation of the FPF is to find solutions to (16). Typically it must be solved by numerical techniques, such as direct numerical solutions or Galerkin solutions based on the weak formulation of (16) [30]. This leads to suboptimal feedback gains, but initial tests indicate that constant-gain approximations provide competitive performance in some cases [24], [25], [28]. Approximations of varying complexity can be
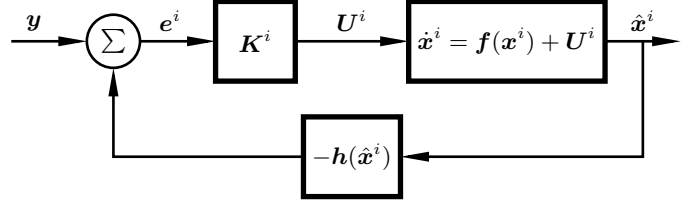

Fig. 1. Simplified block diagram of the FPF. It uses feedback gains $\{\boldsymbol{K}^i\}_{i=1}^N$ to control the particles $\{\boldsymbol{x}^i\}_{i=1}^N$. This is in contrast to the conventional PF, where only the particles' weights are changed in the measurement update.
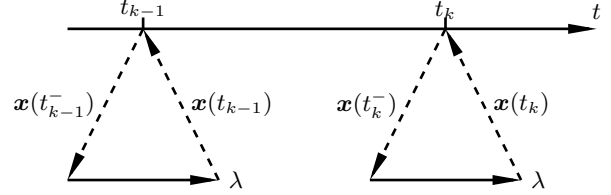

Fig. 2. Illustration of the measurement update in the FPF. The state $\boldsymbol{x}$ is predicted up to t_{k-1}. When $\boldsymbol{y}_{k-1}$ arrives, a simulation-based update, using pseudo-time λ, which corrects the predicted state estimate is performed.

computed based on the weak formulation of (16). A function ϕ_j is a weak solution to (16) if

$$\mathbb{E}(\nabla\phi_j \cdot \nabla\psi) = \mathbb{E}\left(\frac{1}{R_{jj}}(h_j - \hat{h}_j)\psi\right) \tag{17}$$

for all functions ψ belonging to the Sobolev space $H^1(\mathbb{R}^n, p)$ [24]. By restricting ψ to belong to the subspace of $H^1(\mathbb{R}^n, p)$ spanned by $\{\psi_l\}_{l=1}^L$, ϕ_j is approximated as

$$\phi_j = \sum_{l=1}^L \kappa_j^l \psi_l. \tag{18}$$

Hence, (18) is a weighted sum of basis functions $\{\psi_l\}_{l=1}^L$, where $\{\kappa_j^l\}_{l=1}^L$ are constants for a fixed t_k. This implies that

$$\boldsymbol{k}_j = \sum_{l=1}^L \kappa_j^l \nabla\psi_l. \tag{19}$$

Eq. (19) leads to a finite-dimensional approximation of (17):

$$\sum_{l=1}^L \kappa_j^l \mathbb{E}(\nabla\psi_l \cdot \nabla\psi) = \mathbb{E}\left(\frac{1}{R_{jj}}(h_j - \hat{h})\psi\right). \tag{20}$$

In practical implementations, by substituting ψ with each ψ_l and approximating the expectation using the particle distribution, (20) becomes a linear matrix equation

$$\boldsymbol{A}\boldsymbol{\kappa}_j = \boldsymbol{b}_j. \tag{21}$$

Note that the equation system is the same for all particles. Hence, element sl of $\boldsymbol{A}$, A_{sl}, and element l of $\boldsymbol{b}_j$, b_j^l, are found as

$$A_{sl} = \frac{1}{N}\sum_{i=1}^N \nabla\psi_l^i \cdot \nabla\psi_s^i,$$

$$b_j^s = \frac{1}{R_{jj}N}\sum_{i=1}^N (h_j^i - \hat{h}_j)\psi_s^i.$$

A computationally simple approximation of $\boldsymbol{K}_k^i$ is found by choosing $\{\psi_l\}_{l=1}^n = \{x_l\}_{l=1}^n$. With this choice, $\boldsymbol{A}$ in (21) becomes the identity matrix, and we therefore end up with an approximation that is the same for all particles, the constant-gain approximation:

$$\boldsymbol{K}_k \approx \begin{bmatrix} \boldsymbol{c}_1 & \cdots & \boldsymbol{c}_m \end{bmatrix} \boldsymbol{R}^{-1},$$
$$\boldsymbol{c}_j := \frac{1}{N}\sum_{i=1}^N \left(h_j^i - \hat{h}_j \right) \boldsymbol{S}_k^i. \tag{22}$$

In the numerical study, we will use both (22) and an approximation that depends nonlinearly on particle i, that is,

$$\boldsymbol{k}_j^i = \sum_{l=1}^L \kappa_j^l \nabla\psi_l^i. \tag{23}$$

As shown in the next section, whether using (22) or (23) can have drastic impact on estimation performance.

III. Applications

The numerical study contains two examples. One performance measure we use is the root-mean-square error (RMSE): Let $\hat{x}_{k,j}$ denote the estimated mean (weighted mean for the PFs) at time t_k for the jth of M Monte-Carlo simulations. Then the RMSE is computed as

$$\text{RMSE} = \sqrt{\frac{1}{M}\sum_{j=1}^M (x_{k,j} - \hat{x}_{k,j})^2}.$$

The RMSE is not necessarily the best performance measure for nonlinear, non-Gaussian systems, but is used because it is the standard method for comparing estimation performance. The compared methods are:

- FPF: the FPF with constant-gain approximation (22)
- FPFG: the FPF with the feedback gain (23) approximated as a sum of circular basis functions
- UKF: the continuous-discrete time UKF in [31]
- PF: a bootstrap PF with prior sampling (9)
- LLPF: a PF using optimal sampling with linearized likelihood (13)
- RBPF: the Rao-Blackwellized particle filter [27]

The PFs use systematic resampling [32], which has linear complexity in the number of particles.

A. Application 1—Reentry problem

The first application considers state estimation of a vehicle that enters the atmosphere at high speed, see Fig. 3. This problem has been used as a benchmark problem in several papers, for example, in [1], [33] for demonstrating the performance of the discrete-time UKF. Later, the example was used in [31]
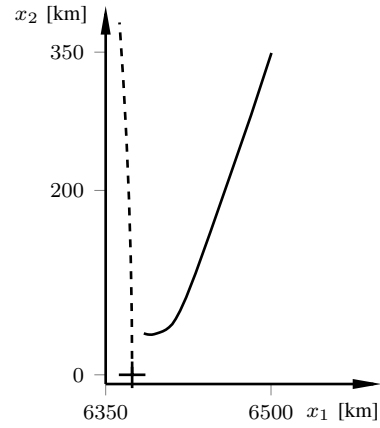

Fig. 3. One realization of the path (black) for the reentry problem. The earth's surface is shown as dashed and the radar location is the '+' in the lower left corner. The coordinate system is located at the earth center.

for verifying the performance of the continuous-discrete time UKF. The motion equations are

$$\dot{x}_1 = x_3, \tag{24a}$$
$$\dot{x}_2 = x_4, \tag{24b}$$
$$\dot{x}_3 = Dx_3 + Gx_1 + w_1, \tag{24c}$$
$$\dot{x}_4 = Dx_4 + Gx_2 + w_2, \tag{24d}$$
$$\dot{x}_5 = w_3, \tag{24e}$$
$$D = b\exp\left(\frac{r_0 - r}{h_0}\right)v, \tag{24f}$$
$$b = b_0\exp(x_5), \tag{24g}$$
$$r = \sqrt{x_1^2 + x_2^2}, \ v = \sqrt{x_3^2 + x_4^2}, \tag{24h}$$
$$G = -\frac{\mu}{r^3}, \tag{24i}$$
$$r_0 = 6374, \ b_0 = -0.59783, \tag{24j}$$
$$h_0 = 13.406, \ \mu = 398601.2, \tag{24k}$$

where (24f) is the drag force, (24i) is the gravity force from the earth, and $\{w_j\}_{j=1}^3$ are zero-mean, white, Gaussian process-noise sources with joint spectral density

$$\boldsymbol{Q} = \begin{bmatrix} 2.4064\cdot 10^{-4} & 0 & 0 \\ 0 & 2.4064\cdot 10^{-4} & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The true initial state is Gaussian distributed with mean and covariance

$$\boldsymbol{x}_0 = \begin{bmatrix} 6500.4 & 349.14 & -1.8093 & -6.7967 & 0.6932 \end{bmatrix}^{\text{T}},$$
$$\boldsymbol{P}_0 = \text{diag}([10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 0]),$$

where $\text{diag}(\cdot)$ is the diagonal matrix with the argument on the diagonal. To enhance UKF performance, the variance of w_3 are in all filters set to 10^{-5}. This noise term is redundant in the FPF, but is included for sake of comparison. Note that this increase in the process noise (roughening) is crucial for the

PFs to have a chance of estimating the static parameter x_5 [34]. For the filters, the initial mean and covariance are

$$\hat{\boldsymbol{x}}_0 = \begin{bmatrix} 6500.4 & 349.14 & -1.8093 & -6.7967 & 0 \end{bmatrix}^{\mathrm{T}},$$
$$\hat{\boldsymbol{P}}_0 = \mathrm{diag}([10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 1]).$$

The vehicle is tracked by a radar that measures the range r and bearing θ with the rate of 2 Hz—that is,

$$\boldsymbol{y} = \begin{bmatrix} r \\ \theta \end{bmatrix} = \begin{bmatrix} \sqrt{x_1^2 + x_2^2} \\ \arctan\left(\dfrac{x_2}{x_1}\right) \end{bmatrix} + \boldsymbol{e}, \tag{25}$$

where the measurement noise is Gaussian distributed with covariance matrix

$$\boldsymbol{R} = \mathrm{diag}([10^{-6}, (0.01\pi/180)^2]).$$

Further details and motivation are found in [1], [33].

The simulated data is generated by forward-propagating (24) using the Euler-Maruyama integrator [35] with step size $\Delta t = 0.001$ s (i.e., 500 steps between each measurement). The filters are discretized using the same scheme with step size $\Delta t = 0.01$ s. The measurement-update step (14) in the FPF is discretized with $\Delta\lambda = 0.01$ (i.e., the particle flow is implemented using 100 time steps). All results are for 100 Monte-Carlo simulations.

1) Results: The time-averaged RMSEs as function of the number of particles are shown in Fig. 4. The PF using optimal proposal with linearized likelihood (LLPF) performs slightly better than PF, but neither of them perform as good as UKF in this example. Furthermore, both of the conventional PFs fail in providing reliable estimates of x_5, despite that roughening is employed. FPF, however, has better performance than UKF already for 100 particles; especially x_5 is estimated more consistently with the FPF.

Fig. 5 presents the minimum execution times as function of time-averaged RMSE of x_1 for one Monte-Carlo simulation, measured with MATLAB's `tic` and `toc` functionality. The minimum of the execution times is chosen because it reduces the effects of disturbances, such as memory management. The implementations utilize vectorization. As seen, for a given execution time, FPF improves RMSE with up to three orders of magnitude compared with PF.

B. Application 2—Two-Body Problem with Bearing Sensors

The next example involves estimating the motion of a satellite that orbits around earth (i.e., a two-body problem). Simplified two-dimensional equations of motion relative to the earth-fixed, earth-centered, inertial frame are given by

$$\begin{aligned} \dot{p}_X &= v_X, \\ \dot{p}_Y &= v_X, \\ \dot{v}_X &= -\mu\frac{p_X}{r^3} + \frac{1}{m}F_X + w_3, \\ \dot{v}_Y &= -\mu\frac{p_Y}{r^3} + \frac{1}{m}F_Y + w_4, \end{aligned} \tag{26}$$
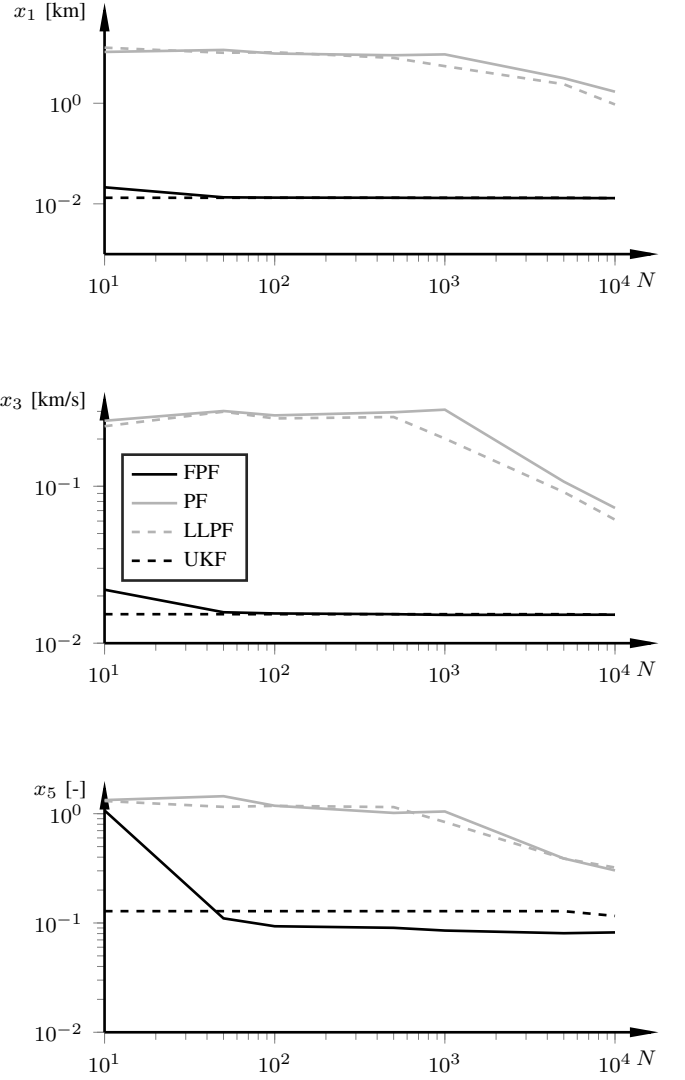

Fig. 4. Time-averaged RMSEs for x_1, x_3, and x_5 as function of the number of particles for the reentry problem. The differences are small, but FPF has slightly smaller RMSE for both x_1 and x_3 for $N \geq 100$. The RMSE values are computed for $N = 10, 50, 100, 500, 1000, 5000, 10000$.
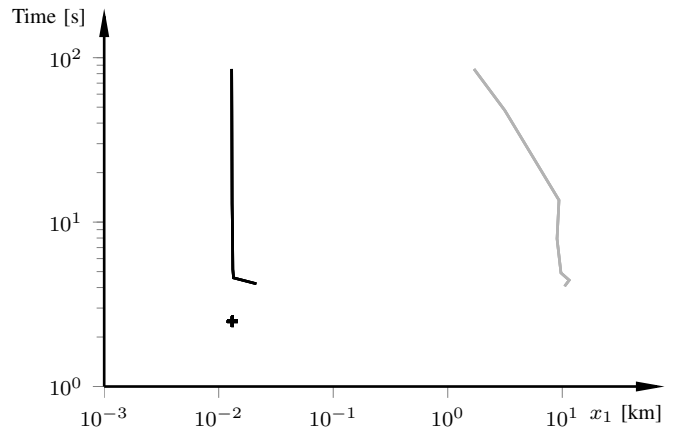

Fig. 5. Minimum execution times for one simulation (20000 time steps) as function of time-averaged RMSE of x_1. FPF is in black and PF in gray. UKF is shown (+) for reference.

where $[p_X, p_Y]$ are the longitudinal and lateral positions in the earth-fixed frame, respectively, and $[v_X, v_Y]$ are the corresponding velocities. F_X and F_Y are the external forces applied to the satellite to correct for the perturbation accelerations w_3 and w_4, $r = \sqrt{p_X^2 + p_Y^2}$, $\mu = 398601.2$ is earth's gravitational constant, and m is the satellite mass. For simplicity, $F_X = F_Y = 0$ in what follows. The perturbations w_3 and w_4 are both assumed Gaussian distributed with zero mean and standard deviation 0.1 m/s^2.[1] The initial conditions are

$$\boldsymbol{x}_0 = \begin{bmatrix} 7000 & 0 & 0 & -7.54 \end{bmatrix}^{\mathrm{T}}, \qquad (27)$$

in km and km/s, respectively (i.e., a low-earth orbit).

Two bearing sensors measure the angle relative to the satellite. The sensors are located at

$$S_1 = (r_0, 0), \quad S_2 = (-r_0, 0),$$

where $r_0 = 6374$ km. The measurement model is

$$\boldsymbol{y}_k = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} \arctan\left(\dfrac{p_Y}{p_X - R_0}\right) \\ \arctan\left(\dfrac{p_Y}{p_X + R_0}\right) \end{bmatrix} + \boldsymbol{e},$$

and both sensors have Gaussian distributed, independent noise, with zero mean and standard deviation 1 deg. Each sensor is only able to track objects that reside in a cone with 40 deg opening angle. When the satellite is within the respective X-axis aligned cone, the sensor provides measurements at 0.1 Hz. Fig. 6 shows a schematic of the setup. Note that the sensors are never active simultaneously. Furthermore, because of the infrequent measurements, there can be a severe mismatch between actual measurement and predicted measurement.

In this example, we use two versions of the FPF. FPF is the constant-gain FPF (22). In addition, FPFG is an FPF where (23) is comprised of 12 circular basis functions, with the arguments normalized between $[0, 2\pi]$. Hence, a 12×12 matrix equation system (21) is solved at each time step in the particle-flow update (14), which results in (23). The initial orbit is assumed uncertain for all filters, with mean (27) and covariance matrix

$$\boldsymbol{P}_0 = \mathrm{diag}([4, 4, 0.04, 0.04]).$$

The simulated data is generated by forward propagating (26) using the Euler-Maruyama scheme with step size $\Delta t = 0.01$ s. The filters are discretized with step size $\Delta t = 0.1$ s, and each simulation lasts for 230 min, corresponding to approximately 2.5 orbits. In the FPFs, $\Delta\lambda = 0.001$.

1) Results: Table I displays the time-averaged RMSEs for the time steps when the satellite is within either of the measurement cones, for 10 Monte-Carlo simulations using 100 particles. FPFG outperforms all other filters, sometimes with more than one magnitude, and neither PF nor RBPF manage to estimate the mean with high accuracy. It is clear that UKF is unable to handle the combined severe nonlinearities and infrequent measurements.

[1]Keplerian orbits do not exist in practice because of perturbation forces. Satellites drift from their assigned orbital positions because of, e.g., solar radiation pressure and atmospheric drag, if not accounted for. These disturbances, i.e., w_3 and w_4, are modeled as Gaussians here.
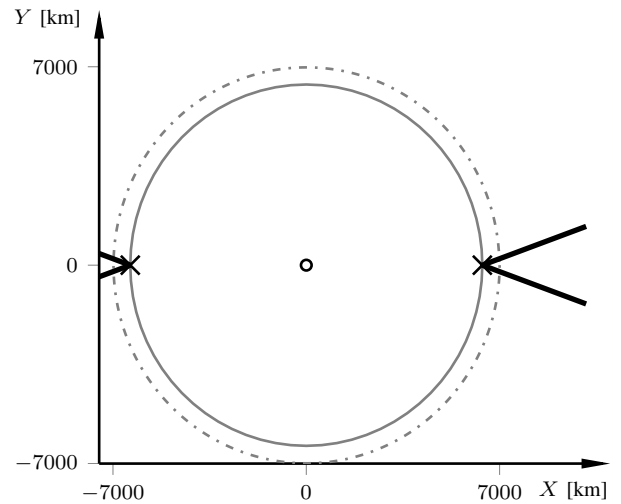

Fig. 6. The two-body problem with two bearing sensors (crosses) that measure the respective angle to the satellite. The earth surface is indicated with the large solid circle and the small solid circle indicates the earth center. The true satellite path for one orbit realization is the dash-dotted circle. The cones illustrate the sensors' viewing angles (40 deg). When the satellite is outside these cones, the estimation algorithms rely solely on prediction.

TABLE I
TIME-AVERAGED RMSEs FOR THE TWO-BODY PROBLEM WHEN EXECUTING 10 MONTE-CARLO SIMULATIONS FOR A SIMULATION TIME OF 230 MIN. THE PARTICLE FILTERS USE $N = 100$. THE RMSE IS ONLY COMPUTED FOR THE TIME STEPS WHEN EITHER OF THE SENSORS TRACK THE SATELLITE.

Algorithm	p_X	p_Y	v_X	v_Y
FPFG	29.7	42.9	0.06	0.04
FPF	36.5	55.5	0.09	0.05
PF	110.2	441.9	0.49	0.12
RBPF	112.7	434.7	0.47	0.12
UKF	127.1	227.7	0.34	0.17

The RMSE measures the accuracy of the estimated mean, but does not necessarily give a proper measure of how well the filters estimate the posterior. The dynamics is governed by an approximately circular orbit; hence, combined with the measurements it is possible to conclude that the posterior should be approximately directed along the orbit. To give an indication of the different filters' abilities to capture the constrained motion and thus represent the posterior, Figs. 7 and 8 show the estimated posteriors for each filter after approximately two orbits, for $N = 100$ and $N = 1000$, respectively. Only FPFG (part a) is able to capture the constrained motion for 100 particles. Both PF and RBPF (part c) give skewed posteriors and significantly biased estimates (compare with Table I). In addition, PF clearly has a scattered particle distribution. The resampling causes this behavior, and results in that the particles are concentrated around clusters. For 1000 particles (Fig. 8), RBPF and PF produce particle distributions that are more aligned with the orbit, but they are still slightly skewed. In both figures, FPF estimates the posterior to be skewed with respect to the orbit, and the tail of the posterior is significant for 1000 particles. This is an effect of the constant feedback gain, which moves all particles
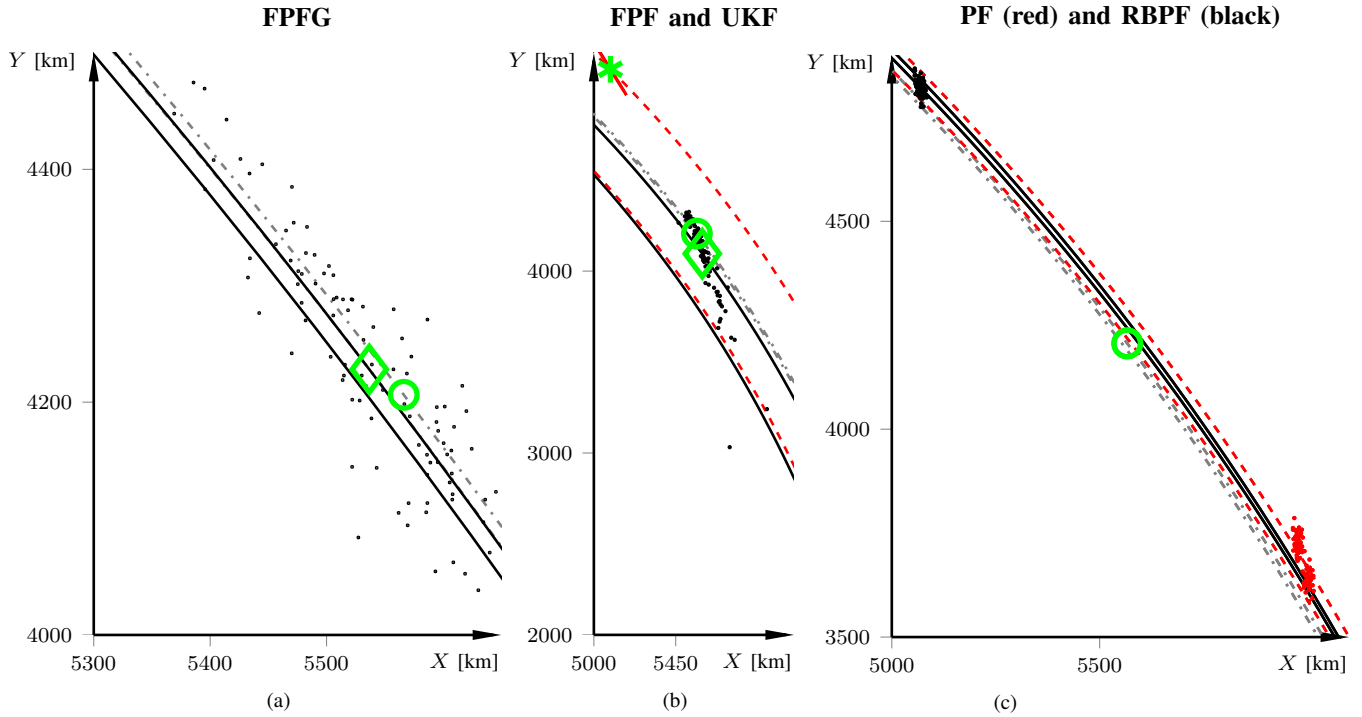
Fig. 7. Particle clouds and (weighted) mean for $N = 100$ after roughly two orbits for one realization. True position is indicated by the green circle and true path is in dash-dotted gray. Part (a) shows the particle cloud, estimated path (solid), and estimated mean (green diamond) for FPFG . Part (b) displays the same for FPF, and estimated mean and 3σ ellipse (solid red) for UKF. Part (c) shows particle distributions for PF in red and RBPF in black. Only FPFG captures that the posterior should be aligned along the orbit without severe bias. The distributions of PF and RBPF are heavily biased and skewed. In addition, the effect of resampling is clearly seen for PF. The distribution of FPF is also skewed, because the constant-gain approximation does not utilize the constraints in the dynamics. The 3σ ellipse for UKF does not cover the true position. Note that the scales differ for the subfigures.
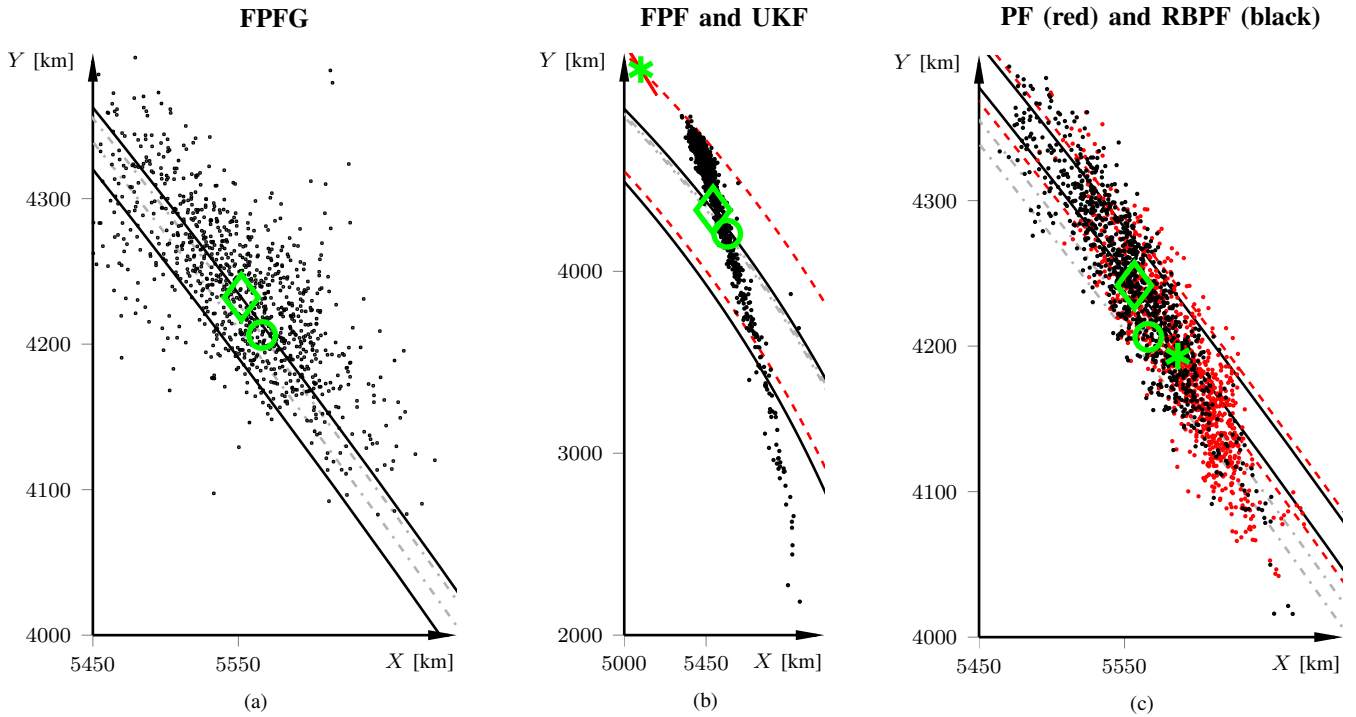

Fig. 8. Same notation as in Fig. 7 for $N = 1000$. For part (c), the green diamond indicates the weighted mean for RBPF. The conventional particle filters now also perform well. The estimated mean is quite accurate with the constant-gain approximation FPF (FPF), but the posterior has large tails. Subfigures (a) and (c) have the same scaling. The covariance matrix for UKF is almost degenerate., and the performance of UKF is clearly inferior compared with all other filters. The plots give a representation of the estimated posteriors, but does not necessarily represent the average accuracy in the mean.

along the same direction whenever a measurement is available (22). When using more complex expressions for the feedback gain that accounts for the dynamics, the constrained motion is more accurately captured. This results in a posterior estimate that is aligned with the orbit. Note that the mean of UKF is heavily biased. The (almost degenerate) 3σ ellipse does not cover the true position. In fact, when inspecting the estimated paths (not shown here), it is clear that the estimated path of UKF slowly diverges for each orbit.

IV. Concluding Discussion

We presented results and assessed the performance of the FPF compared with more traditional filtering techniques for two estimation problems. The considered problems highlight that nonlinearities alone do not justify the use of particle-filter approaches over the UKF. However, combined with large uncertainties, particle-filter approaches is the preferred choice.

The first problem was the reentry problem. With the parameters and noise levels in this particular example, it is clear that the UKF is very competitive. Still, the FPF performs slightly better than the UKF already for 100 particles, especially for estimation of the aerodynamic parameter x_5. The performance of either PF approach is worse than the UKF for reasonably small values of N. All filters used the same noise levels. It is highly likely that roughening/dithering [3], [8] can improve performance, but roughening is already used for x_5. For the combined state and parameter estimation problem, there exist particle filters that give better performance [34]. However, it is notable how much more accurate the FPF is than the PFs without sacrificing computational efficiency.

The two-body problem again highlighted the FPF's relative estimation increase per particle compared with the PFs, when using few particles. In addition, it is clear that with a sensible choice of feedback gain, the posterior estimate is highly accurate even for a small number of particles. The downside is the necessity for solving a linear matrix equation in the measurement update, but it is encouraging that only using 12 basis functions (three per state) results in much improved particle distribution compared with the constant-gain approximation. Future work is to provide guidelines for how to choose the problem-dependent feedback gain.

References

[1] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[2] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Process., Communications, and Control Symp.*, Lake Louise, AB, Oct. 2000.

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

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

[5] D. Fox, S. Thrun, F. Dellaert, and W. Burgard, "Particle filters for mobile robot localization," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. New York: Springer Verlag, 2000.

[6] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, 2007.

[7] D. Hähnel, W. Burgard, D. Fox, and S. Thrun, "A highly efficient Fast-SLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Las Vegas, NV, Nov. 2003.

[8] F. Gustafsson, *Statistical Sensor Fusion*. Lund, Sweden: Utbildningshuset/Studentlitteratur, 2010.

[9] ——, "Particle filter theory and practice with positioning applications," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 25, no. 7, pp. 53–82, 2010.

[10] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 425–437, 2002.

[11] K. Berntorp, "Particle filter for combined wheel-slip and vehicle-motion estimation," in *Am. Control Conf.*, Chicago, IL, Jul. 2015, accepted.

[12] 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.

[13] F. Daum and J. Huang, "Particle degeneracy: root cause and solution," in *Proc. SPIE*, vol. 8050, 2011.

[14] U. D. Hanebeck, K. Briechle, and A. Rauh, "Progressive Bayes: a new framework for nonlinear state estimation," in *Proc. SPIE*, vol. 5099, 2003.

[15] M. F. Huber, F. Beutler, and U. D. Hanebeck, "Semi-analytic Gaussian assumed density filter," in *Am. Control Conf.* San Francisco, CA, Jun. 2011.

[16] P. Bunch and S. Godsill, "Particle filtering with progressive Gaussian approximations to the optimal importance density," in *5th Int. Workshop Computational Adv. Multi-Sensor Adaptive Process.*, Saint Martin, Dec 2013.

[17] F. Daum and J. Huang, "Nonlinear filters with particle flow," in *Proc. SPIE*, vol. 7445, 2009.

[18] F. Daum, J. Huang, and A. Noushin, "Exact particle flow for nonlinear filters," in *Proc. SPIE*, vol. 7697, 2010.

[19] F. Daum and J. Huang, "Particle flow with non-zero diffusion for nonlinear filters," in *Proc. SPIE*, vol. 8745, 2013.

[20] T. Ding and M. J. Coates, "Implementation of the Daum-Huang exact-flow particle filter," in *IEEE Statistical Signal Process. Workshop*, Ann Arbor, MI, Aug. 2012.

[21] T. Yang, P. G. Mehta, and S. P. Meyn, "A mean-field control-oriented approach to particle filtering," in *Am. Control Conf.*, San Francisco, CA, Jun. 2011.

[22] T. Yang, "Feedback particle filter and its applications," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2014.

[23] T. Yang, H. Blom, and P. Mehta, "The continuous-discrete time feedback particle filter," in *Am Control conf.*, Portland, OR, Jun. 2014.

[24] T. Yang, R. Laugesen, P. Mehta, and S. Meyn, "Multivariable feedback particle filter," in *51st Conf. Decision and Control*, Grand Wailea, Maui, Hawaii, Dec. 2012.

[25] T. Yang, P. Mehta, and S. Meyn, "Feedback particle filter," *IEEE Trans. Autom. Control*, vol. 58, no. 10, pp. 2465–2480, 2013.

[26] J. Steinbring and U. D. Hanebeck, "S 2 KF: The smart sampling Kalman filter," in *16th Int. Conf. Information Fusion*, Istanbul, Turkey, Jul. 2013.

[27] T. B. Schön, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear nonlinear state-space models," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2279–2289, 2005.

[28] A. K. Tilton, S. Ghiotto, and P. G. Mehta, "A comparative study of nonlinear filtering techniques," in *16th Int. Conf. Information Fusion*, Istanbul, Turkey, Jul. 2013.

[29] G. Tessitore and J. Zabczyk, "Wong-Zakai approximations of stochastic evolution equations," *J. evolution eqs.*, vol. 6, no. 4, pp. 621–655, 2006.

[30] A. Ern, *Theory and practice of finite elements*. Springer, 2004.

[31] S. Särkkä, "On unscented Kalman filtering for state estimation of continuous-time nonlinear systems," *IEEE Trans. Autom. Control*, vol. 52, no. 9, pp. 1631–1641, 2007.

[32] J. D. Hol, T. B. Schön, and F. Gustafsson, "On resampling algorithms for particle filters," in *IEEE Nonlinear Statistical Signal Process. Workshop*, Cambridge, UK, Sep. 2006.

[33] S. Julier and J. Uhlmann, "Corrections to unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 12, p. 1958, 2004.

[34] O. Cappé, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proc. IEEE*, vol. 95, no. 5, pp. 899–924, 2007.

[35] P. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*. Springer-Verlag, 1999.