Particle Flow Auxiliary Particle Filter

Yunpeng Li
Dept. of Electrical and Computer Engineering
McGill University
Montréal, Québec, Canada
Email: yunpeng.li@mail.mcgill.ca

Lingling Zhao
School of Computer Science and Technology
Harbin Institute of Technology
Harbin, China
Email: zhaoll@hit.edu.cn

Mark Coates
Dept. of Electrical and Computer Engineering
McGill University
Montréal, Québec, Canada
Email: mark.coates@mcgill.ca

***Abstract*—Particle flow filters have been recently developed as an alternative approach for nonlinear filtering. The particles approximating the prior are migrated using differential equations to be distributed according to the posterior. Computationally tractable exact solutions only exist for linear Gaussian models. For other scenarios, approximations are required and it is not fully understood how these approximations impact the movement of the particles and the subsequent propagation of error in the filter. An alternative approach is to use the particle flow methods to perform the importance sampling step within a particle filtering framework. Existing methods along these lines involve either intensive calculation or the construction of a transport map, which can be challenging. In this paper, we propose to use existing particle flow methods in an auxiliary particle filter. The flows are used to sample auxiliary variables; and these allow us to identify importance sampling distributions that are well-matched to the posteriors. Simulations results indicate that the auxiliary particle filters we develop have accuracy and computational complexity similar to that of the underlying particle flow filters.**

I. Introduction

Particle filters struggle to perform well in high dimensional state spaces unless one employs a very large number of particles [1], [2]. Particle flow algorithms were recently proposed to migrate particles from the prior to the posterior distribution to avoid the degeneracy induced by sampling in a high dimensional space. The "flow" involves propagating particles according to a partial differential equation. The initial version of the particle flow filtering algorithm [3], [4] involves an incompressible particle flow. A series of papers provide different solutions to the problem based on different assumptions about the evolution of the density [5]–[8].

Most of the derived solutions are computationally intractable and require approximations when implemented. An important exception is when the prior and the likelihood distributions are both from the exponential family, e.g. Gaussian. The exact particle flow filter [5] was developed for this case. The papers [3]–[8] are rich in algorithmic innovation but provide few implementation details. A more detailed algorithmic description of the exact flow filter was provided in [9]. The algorithm employs an extended or unscented Kalman filter in parallel to estimate the covariance matrix required in the particle flow equations. In this paper, we refer to the algorithm in [9] as the exact Daum and Huang filter (EDH). A more computationally-intensive variant was described in [10]. We call this the localized exact Daum and Huang filter (LEDH) because it computes an individual flow for each particle rather than evaluating a single flow at the mean of the particle distribution.

In the past three years, Daum and Huang have described several new particle flow algorithms. A non-zero diffusion particle flow filter (NZDDH) was proposed in [11]. Khan et al. compared the performance of the NZDDH and the EDH/LEDH algorithms for a bearings-and-range multiple target tracking problem in [12]. They observed that NZDDH outperformed EDH and LEDH, but was itself outperformed by a bootstrap particle filter (BPF) with a much larger number of particles.

Although particle flow algorithms display very promising characteristics, there is not yet a good theoretical or practical understanding of the impact of approximations that must be incorporated when implementing the algorithms. An alternative approach is to use particle flow methods to derive an importance sampling distribution within a particle filtering framework. The approximation error in such a combined algorithm is solely the sampling error, and the many results from the sequential Monte Carlo filtering literature concerning error propagation, stability, and convergence then apply.

There have only been a few proposals for using particle flows or transport procedures to derive sampling distributions. In [13], Reich described a procedure that moved particles through coupling and he derived the equations required to appropriately update importance weights. The method requires the careful design of a complex transport map function and the design process can be challenging for some filtering problems. In [14], Bunch et al. derived an algorithm, which we refer to as GPFIS, that maintains a correctly weighted particle representation of the posterior distribution at all stages of the flow. Particles at each stage are treated as being drawn from a proposal distribution. The weights are corrected accordingly, but calculating the update requires the computationally-expensive solution of many differential equations.

In this paper, we propose a much simpler approach. We embed particle flow techniques within an auxiliary particle filter framework. The flows are used to sample auxiliary variables that define an importance sampling distribution that is well-matched to the posterior. The computational overhead is much less than that of the procedure in [14], being approximately equivalent to that of the underlying particle flow algorithm. In contrast to the method described in [13], we can avoid the identification of a transport map, and directly insert off-the-shelf particle flow algorithms.

The paper is organized as follows. Section II provides

a problem statement. The importance sampling after particle flow algorithm is proposed in Section III. Section IV presents simulation results and Section V makes concluding remarks.

II. Problem statement

We address a nonlinear filtering task with the following models:

$$x_k = g(x_{k-1}, u_k) \tag{1}$$
$$z_k = h(x_k, v_k)\,. \tag{2}$$

Here the observation z_k is related to the unobserved state x_k, $g()$ is the state-transition function, and $h()$ is a nonlinear measurement function. u_k and v_k are the process and measurement noises, respectively. The nonlinear filtering goal is to estimate the marginal posterior distribution $p(x_k|z_{1:k})$, given a sequence of observations $z_{1:k} = \{z_1, \ldots, z_k\}$.

III. Particle flow auxiliary particle filter

Our proposed algorithm is constructed with an auxiliary particle filter framework. After the completion of step $k-1$, we obtain particles $\{(x_{k-1}^i, w_{k-1}^i)\}_{i=1}^{N_p}$ that approximate the marginalized posterior distribution $p(x_{k-1}|z_{1:k-1})$. Auxiliary variables $\{\mu_k^i\}_{i=1}^{N_p}$ are generated by moving particles first using the system model without noise, then through the particle flow process. $\{\{\mu_k^i\}_{i=1}^{N_p}, w_k^i\}$ are then used to produce particles that are drawn from a proposal distribution which conditions on the new measurement z_k. Importance weights are calculated for each particle in the last step.

A. Exact Gaussian flow algorithms

We first use existing particle flow algorithms to generate auxiliary variables $\{\mu_k^i\}_{i=1}^{N_p}$. Two exact Gaussian flow algorithms that suit our needs are described as follows.

1) DH exact Gaussian flow with zero diffusion: The flow of auxiliary particles $\{\mu_k^i\}_{i=1}^{N_p}$ can be modelled to follow an exact Gaussian flow with zero diffusion, as proposed in [5]:

$$\frac{d\mu_k^i}{d\lambda} = \zeta(\mu_k^i, \lambda) = A(\lambda)\mu_k^i + b(\lambda) \tag{3}$$

where

$$A(\lambda) = -\frac{1}{2}PH^T(\lambda HPH^T + R)^{-1}H \tag{4}$$
$$b(\lambda) = (I + 2\lambda A)[(I + \lambda A)PH^TR^{-1}z_k + A\bar{\mu}_k^i] \tag{5}$$

in which $\bar{\mu}_k^i$ is the predicted value of μ_k^i, i.e. the mean of the prior distribution. P is the covariance matrix of the prediction error for the prior distribution, which can be estimated by the sample covariance matrix, the extended Kalman filter (EKF), or the unscented Kalman filter (UKF). For nonlinear models, the linearization of the measurement model can be used to construct the measurement matrix H, i.e. $h(\mu_k^i) = H\mu_k^i$. R is the covariance matrix of the measurement error. The pseudocodes of two typical algorithms, the EDH [9] and the LEDH, are both presented in [10].

2) DH exact Gaussian flow with non-zero diffusion: A non-zero diffusion particle flow algorithm is developed in [11]. The flow equation is expressed as

$$\zeta(\mu_k^i, \lambda) = -[\nabla^2\phi(\mu_k^i, \lambda)]^{-1}\nabla\log(l(\mu_k^i))\,, \tag{6}$$

where $l()$ is the likelihood function, and the Hessian of the log-homotopy function $\phi(\mu_k^i, \lambda)$ can be approximated by

$$\nabla^2\phi(\mu_k^i, \lambda) \approx -P^{-1} + \lambda\nabla^2\log l(\mu_k^i)\,. \tag{7}$$

For an additive Gaussian likelihood function with $l(z_k|x_k) = N(z_k; h(x_k), \Sigma_k)$,

$$\nabla\log l(\mu_k^i) = -[\nabla h(\mu_k^i)]^T\Sigma_k^{-1}(h(\mu_k^i) - z_k)\,. \tag{8}$$

The (m,n)-th element of $\nabla^2\log l(\mu_k^i)$ is

$$\begin{aligned}[\nabla^2\log l(y)]_{m,n} = &-[\nabla^2_{m,n}h(\mu_k^i)]^T\Sigma_k^{-1}(h(\mu_k^i) - z_k) \\ &-[\nabla_i h(\mu_k^i)]^T\Sigma_k^{-1}\nabla_j h(\mu_k^i)\,.\end{aligned} \tag{9}$$

B. Importance sampling

Once the auxiliary variables $\{\mu_k^i\}_{i=1}^{N_p}$ have been calculated, the generated particles are drawn from a proposal distribution $q(x_k^i|x_{k-1}^i, z_k) = N(x_k^i; \mu_k^i, \Sigma_k^c)$. The choice of the covariance matrix Σ_k^c is problem dependent. The importance weights can be easily calculated. We refer to this algorithm as PF-APF; pseudocode is presented in Algorithm 1.

IV. Simulation and Results

We evaluate the performance of PF-APF based on different algorithms with a multi-target acoustic sensor example. Comparison algorithms are EDH, LEDH, NZDDH, GPFIS and BPF.

A. Simulation setup

The multi-target simulation setup we use was proposed in [15] and adapted in [10]. Four targets move independently following a state-transition equation given by $x_k^{(p)} = Gx_{k-1}^{(p)} + Wu_k^{(p)}$, where $x_k^{(p)} = (\mathrm{x}_k^{(p)}, \mathrm{y}_k^{(p)}, \dot{\mathrm{x}}_k^{(p)}, \dot{\mathrm{y}}_k^{(p)})^T$ contains the position and velocity of target p in an x-y plane. $G \in \mathbb{R}^{4\times4}$ and $W \in \mathbb{R}^{4\times4}$ are system matrices. $u_k^{(p)} \sim N(0, \sigma_u^2 I_4)$ is an i.i.d. Gaussian noise vector. There are 25 acoustic amplitude sensors deployed in a region of size 40 m×40 m. Each target emits a sound of amplitude A, which is sensed by sensor j at position ξ^j with an amplitude

$$\psi^j(x_k) = \sum_{p=1}^{P}\frac{A}{||(\mathrm{x}_k^{(p)}, \mathrm{y}_k^{(p)})^T - \xi^j||^\kappa + d_0}\,.$$

The measurement z_k^j of sensor j at time k due to the additive contribution of each target is modelled as

$$z_k^j = \psi^j(x_k) + v_k^j$$

where $x_k = (x_k^{(1)}; x_k^{(2)}; x_k^{(3)}; x_k^{(4)})$ is the overall state vector of dimension 16, $v_k^j \sim N(0, \sigma_v^2)$.

Algorithm 1: Particle flow auxiliary particle filter.

Initialization: Draw $\{x_0^i\}_{i=1}^{N_p}$ from the prior $p_0(x)$;
Set $\{w_0^i\}_{i=1}^{N_p} = \frac{1}{N_p}$;
for k = 1 to T **do**
  // Calculate auxiliary variables μ_k^i;
  for $i = 1, \ldots, N_p$ **do**
    Propagate particles $\mu_k^i = g(x_{k-1}^i, 0)$;
  end for
  Estimate $P_{k|k-1}$ using the sample covariance matrix, EKF, or UKF;
  for $i = 1, \ldots, N_p$ **do**
    Set $\lambda = 0$;
    for $j = 1, \ldots, N_\lambda$ **do**
      Set $\lambda = \lambda + \Delta\lambda(j)$;
      Calculate $\frac{d\mu_k^i}{d\lambda} = \zeta(\mu_k^i, \lambda)$ using (3) or (6);
      Migrate particles: $\mu_k^i = \mu_k^i + \Delta\lambda(j)\frac{d\mu_k^i}{d\lambda}$;
    end for
  end for
  for $i = 1, \ldots, N_p$ **do**
    Draw $x_k^i \sim N(\mu_k^i, \Sigma_k^c)$;
    $w_k^i = \dfrac{p(x_k^i|x_{k-1}^i)p(z_k|x_k^i)}{N(x_k^i; \mu_k^i, \Sigma_k^c)} w_{k-1}^i$;
  end for
  for $i = 1, \ldots, N_p$ **do**
    Normalize $w_k^i = w_k^i / \sum_{s=1}^{N_p} w_k^s$;
  end for
  Estimate $\hat{x}_k$ from $\{x_k^i, w_k^i\}$;
  (Optional) Resample $\{x_k^i, w_k^i\}_{i=1}^{N_p}$ and regularize to obtain $\{x_k^i, \frac{1}{N}\}_{i=1}^{N_p}$;
end for

In the simulation, $P = 4$, $A = 10$, $G = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, and $W = \begin{bmatrix} 1/3 & 0 & 0.5 & 0 \\ 0 & 1/3 & 0 & 0.5 \\ 0.5 & 0 & 1 & 0 \\ 0 & 0.5 & 0 & 1 \end{bmatrix}$. $\sigma_u^2 = 0.05$, $\kappa = 1$, and $d_0 = 0.1$. We set $\sigma_v^2 = 0.01$, indicating that measurements are highly informative. The targets are with initial states $[12, 6, 0.001, 0.001]^T$, $[32, 32, -0.001, -0.005]^T$, $[20, 13, -0.1, 0.01]^T$, $[15, 35, 0.002, 0.002]^T$.

We run the simulation for 100 Monte Carlo trials. In each trial, we generate a different trajectory and an associated set of measurements. Each algorithm then processes the same set of measurements for each trial. The simulation is conducted in Matlab.

B. Parameter values for the filtering algorithms

Exponentially increasing step size is recommended in [11] and [12] for numerical integration within the non-zero diffusion particle flow algorithm. Here we adopt an exponentially spaced sequence of 29 λ values for all particle flow type algorithms. The constant ratio of step sizes is 1.2 and the initial step size is approximately 0.001 (the exact value is chosen so that the sum of step sizes is equal to 1). The covariance of prior distribution are all estimated with an EKF executed in parallel. The number of particles N_p is 100 unless explicitly mentioned.

We adopt the redraw strategy in [10] for the EDH, LEDH, and NZDDH filters at the start of each time step. The diffusion terms for the NZDDH and GPFIS algorithms are both set to 0 based on preliminary numerical simulation results, which is also the suggested value in [11] and [14].

For all filtering algorithms, we sample the initial mean m_0 from a Gaussian centered at the true initial states with variance 1 for positions and 0.000025 for velocity elements. The covariance of dynamic noise is modelled as $\begin{bmatrix} 3 & 0 & 0.1 & 0 \\ 0 & 3 & 0 & 0.1 \\ 0.1 & 0 & 0.03 & 0 \\ 0 & 0.1 & 0 & 0.03 \end{bmatrix}$, larger than that used to generate tracks. The variances of the position components are modelled to be larger than the velocity components, which agrees with the initial target states. In PF-APF algorithms, the covariance matrix of the correction noise Σ_k^c is set to $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}$, smaller than that of the dynamic noise, as we assume auxiliary variables have been moved close to the target distribution. We perform resampling when the effective sample size is less than $\frac{N_p}{2}$. We set the covariance matrix of the regularization noise we add in the regularization step to be the same as Σ_k^c.

C. Experimental results

Figure 1 shows the average position errors at each time step for the various tracking algorithms we compare. The LEDH method exhibits the smallest average tracking error. EDH, GPFIS, and PF-APF based on LEDH or EDH have slightly greater errors. The average error from NZDDH is not displayed, since 9 out of 100 trials result in lost tracks, which are declared when the average position error among four targets reaches more than 40 m at any time step of the tracking. The NZDDH filter is prone to numerical approximation error, and if we do not redraw particles every time step from an approximated Gaussian, then all trials result in lost tracks. The reason is that the flow calculation in (6) is not numerically stable, and can often lead to unreasonably large movement for individual particles if the uneven intermediate step size is also large. The auxiliary particle filtering framework corrects for this and prevents lost tracks for most trials. All auxiliary particle filtering approaches also have significantly smaller average tracking errors than BPF with 10000 particles. Boxplots of the performances of several of the algorithms are shown in Figure 2. The LEDH filter has very few outliers and much smaller median errors in the last few time steps when compared to the other approaches. The computational cost of PF-APF is almost the same as the particle flow methods it utilizes, and is much smaller that of GPFIS, as shown in Table I.

V. Conclusion

In this paper, we have proposed a particle flow auxiliary particle filter algorithm. Although theoretically appealing, particle flow algorithms often involve multiple stages of model

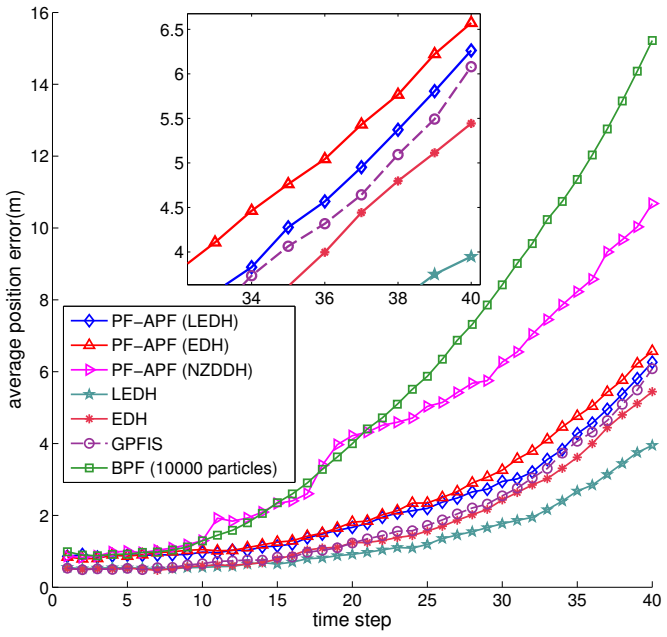

Fig. 1. Average errors of different filtering algorithms at each time step. The number of particles is 100 for all algorithms except the BPF.

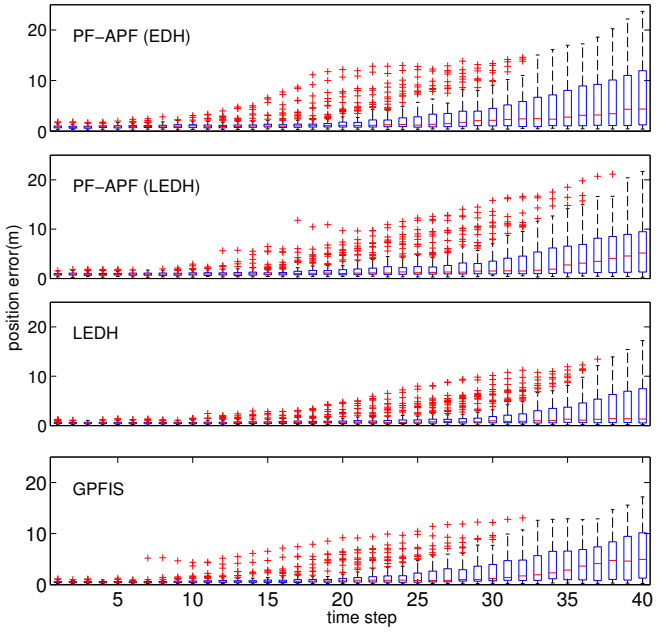

Fig. 2. Error boxplots of different filtering algorithms at each time step.

TABLE I. TYPICAL EXECUTION TIME PER TIME STEP OF DIFFERENT ALGORITHMS. THE BPF USES 10000 PARTICLES, AND ALL OTHER ALGORITHMS USE 100 PARTICLES. RESULTS ARE PRODUCED USING A WORKSTATION WITH AN INTEL I7-4770K CPU AND 32 GB RAM.

Algorithm	PF-APF (EDH)	PF-APF (LEDH)	PF-APF (NZDDH)	EDH	GPFIS	BPF
Avg. exec. time per step (s)	0.007	0.35	3.5	0.007	29	0.03

approximations and flow assumptions which can lead to a discrepancy between the target distribution and the actual distribution. Including particle flow into the auxiliary particle filtering framework transforms these approximation errors into well-studied sampling errors.

We have observed that adding the importance sampling step does not significantly impact the performance of particle flow algorithms in our simulation. It also has a huge computational cost saving compared with other algorithms that link particle flow with importance sampling.

Our future work will include more extensive simulation experiments, which will help us develop a more comprehensive understanding of the behaviour and performance of the particle flow filters, and help us address the question of whether the incorporation of an importance sampling framework is useful or desirable.

REFERENCES

[1] T. Bengtsson, P. Bickel, and B. Li, "Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems," in *Probability and Statistics: Essays in Honor of David A. Freedman*, D. Nolan and T. Speed, Eds. Beachwood, OH, USA: Institute of Mathematical Statistics, 2008, vol. 2, pp. 316–334.

[2] C. Snyder, T. Bengtsson, P. Bickel, and J. Anderson, "Obstacles to high-dimensional particle filtering," *Monthly Weather Review*, vol. 136, no. 12, pp. 4629–4640, 2008.

[3] F. Daum and J. Huang, "Nonlinear filters with log-homotopy," in *Proc. SPIE Signal and Data Processing of Small Targets*, Sep. 2007, p. 669918.

[4] ——, "Particle flow for nonlinear filters with log-homotopy," in *Proc. SPIE Signal and Data Processing of Small Targets*, Apr. 2008, p. 696918.

[5] F. Daum, J. Huang, and A. Noushin, "Exact particle flow for nonlinear filters," in *Proc. SPIE Conf. Signal Proc., Sensor Fusion, Target Recog.*, Apr. 2010, p. 769704.

[6] F. Daum and J. Huang, "Exact particle flow for nonlinear filters: Seventeen dubious solutions to a first order linear underdetermined PDE," in *Asilomar Conf. Signals, Systems and Computers (ASILOMAR)*, Pacific Grove, CA, USA, Nov. 2010, pp. 64–71.

[7] F. Daum, J. Huang, and A. Noushin, "Coulomb's law particle flow for nonlinear filters," p. 81370B, Sep. 2011.

[8] F. Daum and J. Huang, "Small curvature particle flow for nonlinear filters," in *Proc. SPIE Signal and Data Processing of Small Targets*, May 2012, p. 83930A.

[9] S. Choi, P. Willett, F. Daum, and J. Huang, "Discussion and application of the homotopy filter," in *Proc. SPIE Conf. Signal Proc., Sensor Fusion, Target Recog.*, May 2011, p. 805021.

[10] T. Ding and M. J. Coates, "Implementation of the daum-huang exact-flow particle filter," in *IEEE Statistical Signal Processing Workshop (SSP)*, Ann Arbor, MI, USA, Aug. 2012, pp. 257–260.

[11] F. Daum and J. Huang, "Particle flow with non-zero diffusion for nonlinear filters," vol. 8745, May 2013, p. 87450P.

[12] M. A. Khan and M. Ulmke, "Non-linear and non-Gaussian state estimation using log-homotopy based particle flow filters," in *Proc. Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, Bonn, Germany, Oct. 2014, pp. 1–6.

[13] S. Reich, "A guided sequential Monte Carlo method for the assimilation of data into stochastic dynamical systems," in *Recent Trends in Dynamical Systems*. Springer Basel, 2013, vol. 35, pp. 205–220.

[14] P. Bunch and S. Godsill, "Approximations of the optimal importance density using gaussian particle flow importance sampling," *arXiv preprint arXiv:1406.3183*, 2014.

[15] O. Hlinka, O. Sluciak, F. Hlawatsch, P. M. Djuric, and M. Rupp, "Distributed Gaussian particle filtering using likelihood consensus," in *Proc. Intl. Conf. Acoustics, Speech and Signal Proc. (ICASSP)*, Prague, Czech Republic, May 2011, pp. 3756–3759.