Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Auxiliary-Filter-Free Incompressible Particle Flow Filtering Using Direct Estimation of the Log-Density Gradient with Target Tracking Examples

Yeongkwon Choe* and Chan Gook Park**

*Department of Mechanical and Aerospace Engineering/ ASRI, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Korea, (e-mail: *veritasbbo@snu.ac.kr, **chanpark@ snu.ac.kr).*

Abstract: This paper presents an incompressible particle flow filtering method that does not require an auxiliary filter by estimating log-density gradients directly from particles. Particle flow filter (PFF) is likely to avoid particle impoverishment and degeneracy problems that occur in particle filters because particles themselves move toward desired density to perform measurement updates. There are various implementation forms for PFF depending on the assumptions made about the flow. This paper deals with PFF using incompressible flow. Incompressible PFF requires the log-density gradient to calculate the flow. The well-known gradient estimation method for incompressible PFF is a finite difference method collaborating with k nearest neighbors(kNN) method. Since this method requires the prior knowledge about the prior density value in each particle, it is necessary to use an auxiliary filter or a density estimation technique. As a result, the performance of an auxiliary filter or a density estimation technique can directly affect the PFF performance, and the finite difference method is more likely to be inaccurate than directly estimating the log-density gradient. Therefore, this paper presents a PFF structure applying least-squares log-density gradient (LSLDG) method that estimates the log-density gradient directly from particles. In order to verify the performance of the presented structure, this paper performs both single and multiple target tracking simulations. Simulation results demonstrate that the presented structure has a relatively good estimation performance and works more robustly for various situations.

Keywords: Bayesian methods, Particle filtering, Monte-Carlo methods, Estimation and filtering, Particle flow filtering, Daum-Huang filter, Incompressible flow

1. INTRODUCTION

A particle filter is a Bayesian filter commonly used to handle nonlinear/non-Gaussian models that are difficult to solve analytically. The particle filter is composed of a process of moving particles according to a dynamic model in a prediction phase and importance sampling to get posterior density in a correction phase. For estimating the properties of a probability density, the importance sampling method picks samples from a *proposal density* and estimates the desired density by calculating the ratio between the desired density and the proposal density for each sample. To achieve good results of importance sampling with finite particles, the proposal density should be similar to the desired density. If one uses an improper proposal density, sampling will not occur for the region with the highest probability of the desired density. The result is *particle collapse* that obstructs the filter in working properly. Importance sampling is the process of *trying* with a guess and checking the difference, so as the dimension gets higher, more *trials*, i.e., particles are needed to avoid failure. Numerous studies on particle filters address proposal densities or focus on resampling techniques to solve this problem(T. Li, Bolic, & Djuric, 2015; T. Li, Sun, Sattar, & Corchado, 2014; Musso, Oudjane, & Gland, 2001).

Unlike these studies, (Fred Daum & Huang, 2008) presented a new perspective framework called particle flow for nonlinear filters with log-homotopy. This method constructs a homotopy function that changes from prior density to posterior density with virtual time. Since the homotopy function defines the change in a probability distribution, the Fokker-Planck Equation (FPE) can be used to obtain the imaginary time flow of random variables corresponding to the change in the distribution. Following the flow, each particle can move itself to perform measurement updates. It is feasible to use fewer samples compared to importance sampling because particles move from prior density to posterior density themselves.

There are many ways to get flow models from FPE, and accordingly, various filters have been proposed(Bunch & Godsill, 2016; Fred Daum & Huang, 2012, 2013; Fred Daum, Huang, & Noushin, 2010). Among them, the most commonly used method is the exact Daum-Huang filter (EDHF), which assumes Gaussian prior and posterior distribution and suggests a closed-form solution(Fred Daum et al., 2010; Khan & Ulmke, 2015). Since EDHF requires covariance for prior distribution, extended Kalman filter (EKF) or unscented Kalman filter (UKF) is usually performed in parallel as auxiliary filters. This structure can be affected by the performance of the auxiliary filter and requires a Gaussian assumption. One way to avoid using a Gaussian assumption is to use the incompressible flow assumption. The divergence of incompressible flow is zero, so it is easy to calculate the flow if one knows only the log-density gradient. In conventional incompressible flow-based filters, the k-nearest neighbors (kNN) method is known to be good for obtaining gradients(Fred Daum, Huang, Krichman, & Kohen, 2009). This method takes a finite-difference from nearby k samples to find log-density gradients. For the finite-difference, it is necessary to know the value of the prior density in each of the samples. However, the prior density is given as a set of particles, so the practical implementation requires an auxiliary filter such as EKF or UKF to acquire density values(Choi, Willett, Daum, & Huang, 2011). Alternatively, even if one uses the density estimation methods, the performance may be degraded because the method still differentiates finitely.

Therefore, we use a method of estimating the log-density gradient directly from particles so that a filter does not require an auxiliary filter or a density estimation method. To implement an incompressible flow filter, the log gradient of the prior density at particle moving with the flow should be continuously obtained. Since the moved particles exist in different locations from the particles constituting the prior density, a technique utilizing kernels seems to be suitable. This paper applies the least-squares log-density gradient (LSLDG) method proposed in (Sasaki, Hyvärinen, & Sugiyama, 2014). LSLDG is a non-parametric approach based on score matching. LSLDG consists of constructing basis functions for all or several particles and estimating the weights of the basis functions. Sasaki et al. have generalized this to a method called integrated squared error for density derivatives (ISED)(Sasaki, Noh, Niu, & Sugiyama, 2016).

The contributions of our paper are (i) to propose the use of LSLDG technique in implementing incompressible particle flow filter(PFF) and (ii) to demonstrate the superiority to the existing implementing technique through single target tracking and multi target tracking examples. Through the results, we believe that the proposed structure could be a new alternative to the implementation of incompressible PFF. The paper is organized as follows. First, section 2 summarizes the background theories, including the incompressible PFF and LSLDG. Section 3 discusses an algorithm that applies LSLDG to incompressible PFF structures. Section 4 presents the simulations. The simulations cover both single-target tracking problem and multi-target tracking problem. Section 5 concludes the paper.

2. BACKGROUND

2.1 *Incompressible Particle Flow Filter*

This subsection summarizes the correction process of an incompressible particle flow filter proposed by Daum and Haung(Choi et al., 2011; Fred Daum & Huang, 2008). According to the Bayes law, the posterior density $p(\mathbf{x}_k \mid \mathbf{Z}_k)$ is defined as

$$p(\mathbf{x}_k \mid \mathbf{Z}_k) = \frac{p(\mathbf{z}_k \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{Z}_{k-1})}{p(\mathbf{z}_k \mid \mathbf{Z}_{k-1})} \tag{1}$$

by the prior distribution $p(\mathbf{x}_k \mid \mathbf{Z}_{k-1})$ and the likelihood function $p(\mathbf{z}_k \mid \mathbf{x}_k)$. Here, $\mathbf{x}_k$ is the state vector, and $\mathbf{z}_k$ is the measurement at time k. $\mathbf{Z}_k$ means the set of all measurements up to the time k, and $p(\mathbf{z}_k \mid \mathbf{Z}_{k-1})$ is the normalization term. Ignoring the normalization term, the log-homotopy function $\log p(\mathbf{x}_k, \lambda)$ between the prior and posterior density is

$$\log p(\mathbf{x}_k, \lambda) = \log g(\mathbf{x}_k) + \lambda \log h(\mathbf{x}_k), \tag{2}$$

where $g(\mathbf{x}_k)$ represents the prior density $p(\mathbf{x}_k \mid \mathbf{Z}_k)$, $h(\mathbf{x}_k)$ represents the likelihood function $p(\mathbf{z}_k \mid \mathbf{x}_k)$. By understanding λ as a concept of time, we can assume that random variables $\mathbf{x}_k$ varying with λ follow the Ito stochastic differential equation (SDE),

$$d\mathbf{x}_k = f(\mathbf{x}_k, \lambda) d\lambda + \eta(\mathbf{x}_\lambda, \lambda) d\varepsilon_\lambda, \tag{3}$$

where $f(\mathbf{x}_k, \lambda)$ is a drift term, which is called the flow in this paper, $\eta(\mathbf{x}_k, \lambda)$ is a diffusion term, and ε_λ is the Wiener process. Considering zero-diffusion process, the flow $f(\mathbf{x}_k, \lambda)$ can be reformulated as

$$f(\mathbf{x}_k, \lambda) = \frac{d\mathbf{x}_k}{d\lambda}. \tag{4}$$

The Fokker-Planck equation, which describes the time evolution of the density following Ito SDE in (3), can be summarized by substituting the zero-diffusion assumption and (2) as follows:

$$-\log h(\mathbf{x}_k) - \nabla \cdot f(\mathbf{x}_k, \lambda) = \nabla \log p(\mathbf{x}_k, \lambda) \cdot f(\mathbf{x}_k, \lambda). \tag{5}$$

Assuming an incompressible flow, the divergence of the flow is zero, that is, $\nabla \cdot f(\mathbf{x}_k, \lambda) = 0$, (5) can be reformulated as

$$-\log h(\mathbf{x}_k) = \nabla \log p(\mathbf{x}_k, \lambda) \cdot f(\mathbf{x}_k, \lambda). \tag{6}$$

The minimum norm solution for (6) is

$$f(\mathbf{x}_k, \lambda) = -\frac{(\nabla \log p(\mathbf{x}_k, \lambda))^T}{\|\nabla \log p(\mathbf{x}_k, \lambda)\|^2} \log h(\mathbf{x}_k). \tag{7}$$

As a result, if the filter moves each sample by integrating the flow of (7) from $\lambda = 0$ to $\lambda = 1$, one can obtain a sample set following the posterior distribution.

2.2 *Least-Squared Log-Density Gradient*

This subsection summarizes the least-squares log-density gradient (LSLDG) method proposed by (Sasaki et al., 2014). To directly find the log-density gradient, LSLDG uses the loss function,

$$J_j(\hat{d}_j) = \int \left(\hat{d}_j(\mathbf{x}) - d_j(\mathbf{x}) \right)^2 p(\mathbf{x}) d\mathbf{x}, \tag{8}$$

where $d_j(\mathbf{x})$ represents the j-th element of the log-gradient of density $p(\mathbf{x})$ and $\hat{d}_j(\mathbf{x})$ is the estimate of $d_j(\mathbf{x})$. By substituting $d_j(\mathbf{x}) = \partial_j p(\mathbf{x}) / p(\mathbf{x})$ for (8), the solution which minimizes (8) is identical to the solution for the loss function,

$$J_j(\hat{d}_j) = \int \hat{d}_j(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} + 2 \int \partial_j \hat{d}_j(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}, \tag{9}$$

where ∂_j denotes the j-th element of the partial derivative. By Monte-Carlo integration, (9) can be obtained by the empirical approximation,

$$\hat{J}_j(\hat{d}_j) = \frac{1}{N}\sum_{i=1}^{N}\hat{d}_j(\mathbf{x}^{(i)})^2 + \frac{2}{N}\sum_{i=1}^{N}\partial_j\hat{d}_j(\mathbf{x}^{(i)}) . \qquad (10)$$

Here, N is the number of samples. The estimate of the log-density gradient can be modeled as a combination of basis functions and their weights, i.e.

$$\hat{d}_j(\mathbf{x}) = \boldsymbol{\theta}_j^T\boldsymbol{\Psi}_j(\mathbf{x}) , \qquad (11)$$

where $\boldsymbol{\Psi}_j(\mathbf{x})$ is a vector whose elements are basis functions for each sample, and $\boldsymbol{\theta}_j$ is a weight vector. And the derivative of the model is

$$\partial_j\hat{d}_j(\mathbf{x}) = \boldsymbol{\theta}_j^T\boldsymbol{\Phi}_j(\mathbf{x}) . \qquad (12)$$

The loss function can be redefined by substituting (11), (12), and adding l_2-regularizer as

$$\begin{aligned}\hat{J}_j(\hat{d}_j) &= \boldsymbol{\theta}_j^T\left(\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Psi}_j(\mathbf{x}^{(i)})\boldsymbol{\Psi}_j^T(\mathbf{x}^{(i)})\right)\boldsymbol{\theta}_j \\ &+2\boldsymbol{\theta}_j^T\left(\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Phi}_j(\mathbf{x}^{(i)})\right)+\gamma\boldsymbol{\theta}_j^T\boldsymbol{\theta}_j\end{aligned} , \qquad (13)$$

where $\gamma \geq 0$ is the regularization parameter. In the sense of least-squares, the solution $\hat{\boldsymbol{\theta}}_j$ minimizing (13) is

$$\begin{aligned}\hat{\boldsymbol{\theta}}_j &= -\left\{\left(\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Psi}_j(\mathbf{x}^{(i)})\boldsymbol{\Psi}_j^T(\mathbf{x}^{(i)})\right)+\gamma I_{N\times N}\right\}^{-1} \\ &\cdot\left(\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Phi}_j(\mathbf{x}^{(i)})\right)\end{aligned} . \qquad (14)$$

As a result, the log-density gradient is given as

$$\hat{d}_j(\mathbf{x}) = \hat{\boldsymbol{\theta}}_j^T\boldsymbol{\Psi}_j(\mathbf{x}) . \qquad (15)$$

3. INCOMPRESSIBLE PFF USING LSLDG

By substituting (2) for (7), (7) can be reformulated as

$$f(\mathbf{x}_k,\lambda) = -\frac{\left(\nabla\log g(\mathbf{x}_k)+\lambda\nabla\log h(\mathbf{x}_k)\right)^T}{\left\|\nabla\log g(\mathbf{x}_k)+\lambda\nabla\log h(\mathbf{x}_k)\right\|^2}\log h(\mathbf{x}_k) . \qquad (16)$$

Using a specific measurement model, one can obtain the log-gradient of the likelihood function, $\nabla\log h(\mathbf{x}_k)$ of (16) in closed-form. On the other hand, since the prior density is given by a set of particles, acquiring the log-gradient of prior density needs the LSLDG method. This research uses the model for the basis function of the *i*-th particle,

$$\psi_{i,j}(\mathbf{x}) = \frac{1}{\sigma^2}\left(\mathbf{x}^{(i)}-\mathbf{x}\right)_j \exp\left(-\frac{\left\|\mathbf{x}^{(i)}-\mathbf{x}\right\|^2}{2\sigma^2}\right), \qquad (17)$$

where $(\cdot)_j$ is the *j*-th element of the input vector, and σ is a model parameter.

The basis function of (17) is closer to zero as a particle at $0<\lambda\leq 1$ moves away from the particle of the basis function by the exponential term. In general, since the resultant particle set of the correction process moves to a place where the prior density is high, the basis function of (17) may be suitable. Depending on the situation, there is a little concern about a singularity problem when particles are expected to move to a place where the prior density is low, then the gradient value by (17) approaches zero. Thus, one can add the basis function excluding a decaying term to end the concern. However, when the λ is small, the moving particles will not be far from the prior particles, and if the λ is large, there is the likelihood function term. Therefore, we are not very concerned about this problem and use the basis function of (17).

The proposed algorithm summarized as follows:

1) Initialization: Perform random sampling according to the model of the initial states.
2) Prediction: Propagate particles according to system dynamic model, just like a typical particle filter.
3) Correction: First, one constructs a set of basis functions according to (17) for the particles distributed by prior density. The computational demand can increase dramatically due to the number of particles, so one can randomly choose several particles for a set of basis functions. Calculate the weight vector for LSLDG according to (14). The log-prior density gradient values are obtained according to (15) at each particle moving over λ. Calculate the flow by substituting the estimated gradient into (16), and propagate the particles by recursively numerical integration of the calculated flow from $\lambda=0$ to $\lambda=1$.

4. EXAMPLES

4.1 Single Target Tracking with Range and Bearing Measurements

The state variables of the single target tracking example are two-dimensional position p_x and p_y and two-dimensional velocity v_x and v_y. The measurements are the range r_k and bearing angle ψ_k from the origin. This is represented by the state-space model as follows:

$$\begin{bmatrix}p_{x,k+1}\\p_{y,k+1}\\v_{x,k+1}\\v_{y,k+1}\end{bmatrix} = \begin{bmatrix}1&0&1&0\\0&1&0&1\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}p_{x,k}\\p_{y,k}\\v_{x,k}\\v_{y,k}\end{bmatrix} + \begin{bmatrix}1/2&0\\0&1/2\\1&0\\0&1\end{bmatrix}\mathbf{w}_k , \qquad (18)$$

$$\mathbf{w}_k \sim \mathrm{N}\left(0_{2\times 1},\mathrm{diag}\left(\begin{bmatrix}1&1\end{bmatrix}\right)\right), \qquad (19)$$

$$\begin{bmatrix}\psi_k\\r_k\end{bmatrix} = \begin{bmatrix}\tan^{-1}\left(p_{y,k}/p_{x,k}\right)\\\sqrt{{p_{x,k}}^2+{p_{y,k}}^2}\end{bmatrix}+\mathbf{v}_k , \qquad (20)$$

$$\mathbf{v}_k \sim N\left(0_{2\times 1},\mathrm{diag}\left(\begin{bmatrix}0.01^2&0.2^2\end{bmatrix}\right)\right), \qquad (21)$$

$$\mathbf{x}_0 \sim N\left(\begin{bmatrix}-10&-10&0&0\end{bmatrix}^T,P_0\right), \qquad (22)$$

$$\text{and } P_0 = \mathrm{diag}\left(\begin{bmatrix}5^2&5^2&0.5^2&0.5^2\end{bmatrix}^T\right). \qquad (23)$$

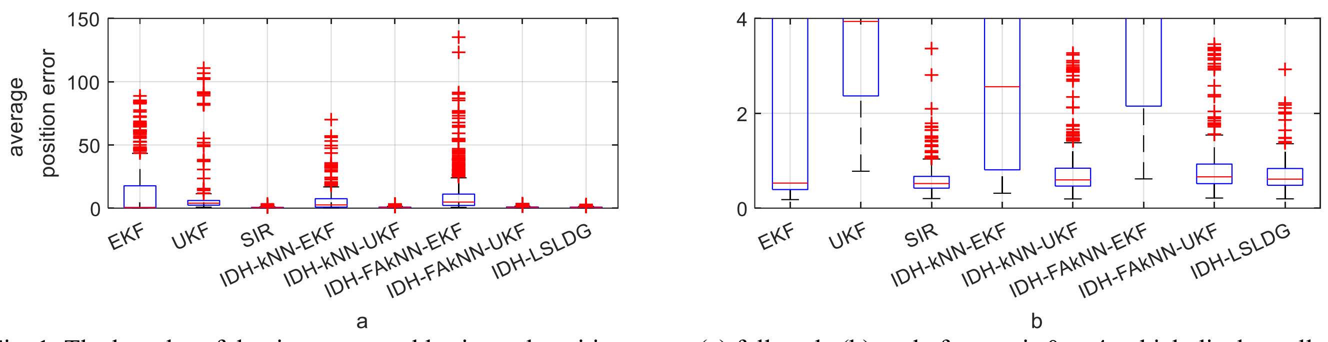

Fig. 1. The boxplot of the time averaged horizontal position error. (a) full-scale (b) scale from axis 0 to 4, which displays all results of only IDH-kNN-UKF, IDH-FAkNN-UKF, and IDH-LSLDG.

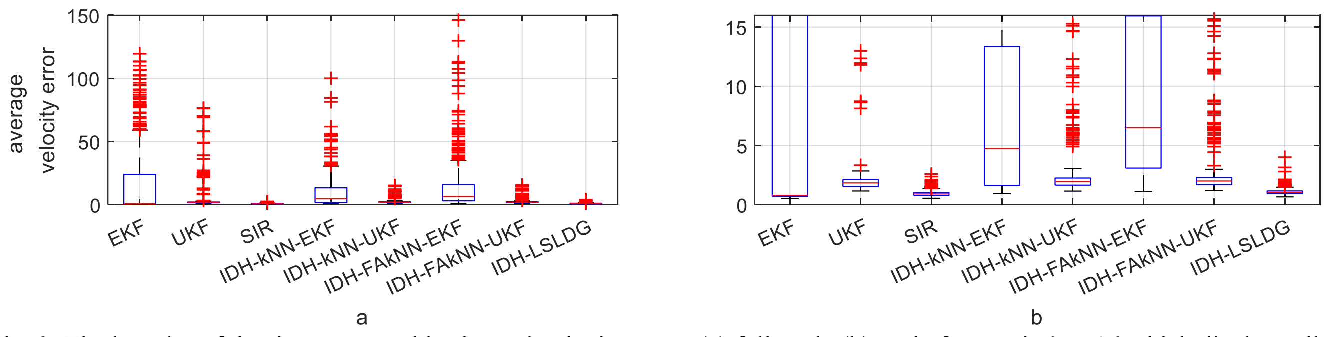

Fig. 2. The boxplot of the time averaged horizontal velocity error. (a) full-scale (b) scale from axis 0 to 16 which displays all results of only IDH-kNN-UKF, IDH-FAkNN-UKF, and IDH-LSLDG.

where $\mathbf{x}_0$ denotes initial states and diag(·) is the function of giving a matrix whose diagonal elements are the elements of an input vector.

Random trajectories are generated according to (18) and (22). The region of interest is set to a 60-by-60 square with the origin as the center, and 100 trajectories only within the region are chosen for results. Five measurement sets for each trajectory are generated, producing a total of 500 ensembles. For comparison, we implement EKF, UKF, and sequential importance resampling (SIR) as a conventional particle filter. Also, we implement the incompressible PFFs (denoted as IDH-kNN) that obtain log-gradients using the finite-difference method in collaboration with the conventional kNN method. We also implement the incompressible PFF filters (denoted as IDH-FAkNN) using the fast approximate kNN method proposed by (Frederick Daum, Huang, Noushin, & Krichman, 2009) to reduce computation. These PFFs use EKF or UKF as an auxiliary filter. SIR uses 300 particles, and PFFs uses 30 particles.

Fig. 1 shows the boxplot results for the time average of the horizontal position error of each ensemble, and Fig. 2 shows that of the horizontal velocity error. SIR seems to be the best in position error, but the result of SIR excluded 37 trials that could not be calculated due to the particle collapse. Therefore, we will mention the results excluding the result of SIR. The performance of IDH-kNN with UKF and the presented incompressible PFF with LSLDG (denoted as IDH-LSLDG) seems to be good. The IDH-kNN-UKF has a slightly smaller median than the IDH-LSLDG, but there are 39 outliers for IDH-kNN-UKF, more than the 13 for IDH-LSLDG. The maximum value of the outlier is also smaller in IDH-LSLDG. Among the velocity error results, IDH-LSLDG shows the best performance, and the maximum value of the outlier is clearly smaller than that of other filters. Although IDH-kNN-UKF and IDH-FAkNN-UKF perform relatively well, the results of these filters have many outliers that are widely distributed. The result of sole UKF implies that the outliers of the PFFs with UKF are originated by the malfunction of the auxiliary filter. EKF is the worst performer, and IDH using it as an auxiliary filter is also poor. As a result, IDH-LSLDG has good estimation performance and works robustly for various situations.

The average elapsed time for a trial was 0.294 msecs for EKF, 0.834 msecs for UKF, 2.31 msecs for SIR, 945 msecs for IDH-kNN-EKF, 887 msecs for IDH-kNN-UKF, 566 msecs for IDH-FAkNN-EKF, 567 msecs for IDH-FAkNN-UKF, and 168 msecs for IDH-LSLDG. Since we implemented unoptimized prototype code on MATLAB, we do not intend to emphasize efficiency from these results, but the results imply that the proposed method will be more advantageous than the existing incompressible PFF implementation method.

4.2 Multi-Target Tracking with Acoustic Measurements Array

Referring to (Y. Li & Coates, 2017), we simulated a system tracking three targets using an array of acoustic sensors. The state variables of the *i*-th target are two-dimensional position $p_x^{(i)}$ and $p_y^{(i)}$ and two-dimensional velocity $v_x^{(i)}$ and $v_y^{(i)}$. Since there are three targets, the state-space is 12-dimensional. The dynamic model of each target is the same as (18). The system noise $\mathbf{w}_k$ follows the distribution,

$$\mathbf{w}_k \sim \mathrm{N}\left(0_{2\times1}, \mathrm{diag}\left(\begin{bmatrix}0.1^2 & 0.1^2\end{bmatrix}\right)\right) \tag{24}$$

The initial state expected values of each target are $\begin{bmatrix}-10 & -15 & -1 & 0\end{bmatrix}^T$, $\begin{bmatrix}0 & 10 & 0 & 1\end{bmatrix}^T$, and $\begin{bmatrix}10 & -10 & -0.5 & -0.5\end{bmatrix}^T$ in the order of x and y position and x and y velocity. The standard deviation of each axis of the initial position is 2, and the velocity is 0.3. The trajectory is constrained to within the 40-by-40 rectangular region of interest. Twenty-five acoustic sensors are placed in the region of interest at intervals of 10. The measurement of the j-th sensor is modeled as

$$z_k^{(j)} = \sum_{i=1}^{3} \frac{10}{\sqrt{\left(p_{x,k}^{(i)} - r_x^{(j)}\right)^2 + \left(p_{y,k}^{(i)} - r_y^{(j)}\right)^2 + 0.1}} + \upsilon_k^{(j)} \tag{25}$$

$$\text{and } \upsilon_k^{(j)} \sim N\left(0, 0.1^2\right). \tag{26}$$

Here, $r_x^{(j)}$ and $r_y^{(j)}$ are the x and y position of the j-th sensor. Fig. 3 represents example trajectories and sensor placements.

Five measurement sets are randomly generated for every 100 random trajectories. The length of the simulation is 80 epochs. For comparison, the same eight filters implemented in a single target tracking example are tested. SIR uses 100 thousand particles, and PFFs use 100 particles. Conditions not mentioned are the same as for the single target tracking simulation.

The optimal mass transfer (OMAT) metric is used to represent the results of multiple targets(Schuhmacher, Vo, & Vo, 2008). The distance metric for the OMAT is Euclidean distance, and the OMAT parameter, p is set to 1. Fig. 4 shows

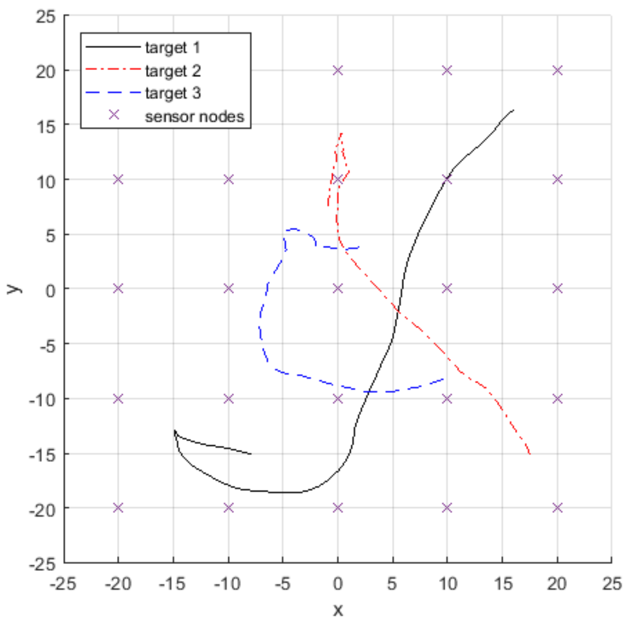

Fig. 3. Example trajectories and sensor placements.

the boxplot of the time-averaged OMAT. To compare the velocity estimation results, Fig. 5 shows the target-averaged velocity error between the true targets and the corresponding estimated targets with respect to the optimal transportation of the OMAT.

The results seem to indicate that the performance of SIR is the best, but the results do not include the failed 145 trials of SIR. Excluding the result of SIR, the results for inliers appear

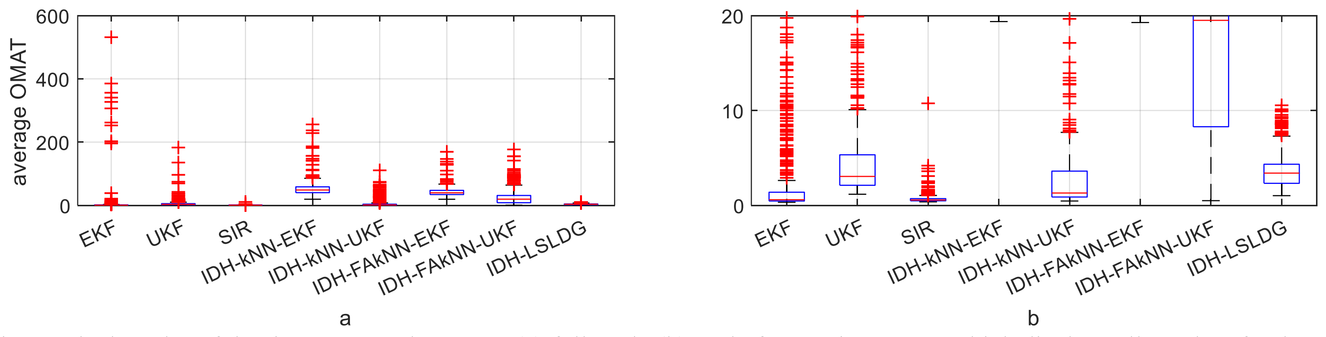

Fig. 4. The boxplot of the time averaged OMAT. (a) full-scale (b) scale from axis 0 to 20, which displays all results of only IDH-LSLDG.

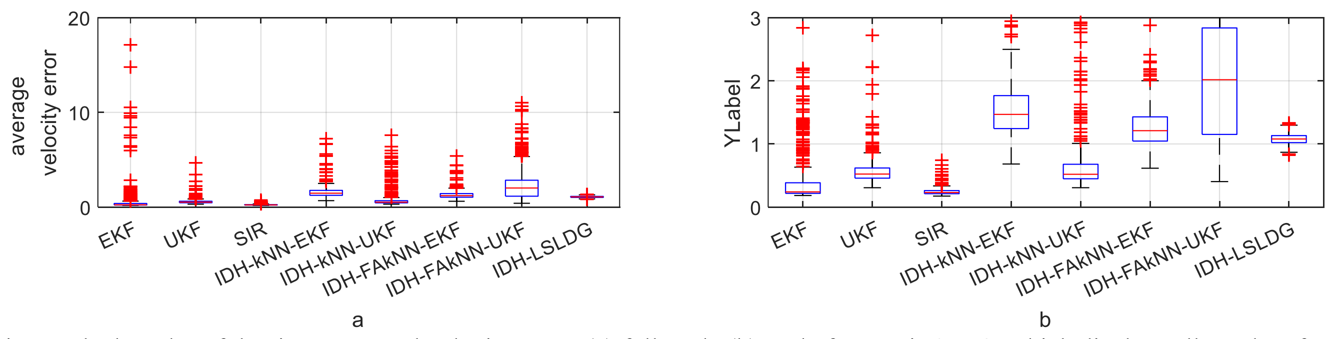

Fig. 5. The boxplot of the time averaged velocity error. (a) full-scale (b) scale from axis 0 to 3, which displays all results of only IDH-LSLDG.

to be good in the order of EKF, IDH-kNN-UKF, and IDH-LSLDG. However, more than 20% of EKF trials are outliers, and about 10% of IDH-kNN-UKF are outliers, while the outliers of IDH-LSLDG are less than 5% of all trials. The maximum value of the averaged OMAT is 531 for EKF, 111 for IDH-kNN-UKF, and 11 for IDH-LSLDG. As a result, the proposed filter structure shows relatively good estimation performance for all trials and stable performance in various situations. The outlier results of EKF and UKF seems to affect the performance of IDHs by using them as auxiliary filters.

The average elapsed time for a trial was 2.14 msecs for EKF, 2.22 msec for UKF, 2.67 secs for SIR, 1.42 secs for IDH-kNN-EKF, 1.47 secs for IDH-kNN-UKF, 1.38 secs for IDH-FAkNN-EKF, 1.41 secs for IDH-FAkNN-UKF, and 1.05 secs for IDH-LSLDG.

5. CONCLUSIONS

In this paper, we applied the LSLDG to estimate the log-prior density gradient for an incompressible PFF. To verify the performance of the presented structure, we performed a single target tracking simulation and a multi-target tracking simulation. Simulation results showed that the presented scheme works more robustly for various situations than other filters including the incompressible PFF structure assisted by UKF and kNN.

ACKNOWLEDGMENT

This work has been supported by the Ministry of Science and ICT, Korea Research Foundation under Space Core Technology Development Program (Project number: NRF-2018M1A3A3A02065722).

REFERENCES

Bunch, P., & Godsill, S. (2016). Approximations of the Optimal Importance Density Using Gaussian Particle Flow Importance Sampling. *Journal of the American Statistical Association*, *111*(514), 748–762. https://doi.org/10.1080/01621459.2015.1038387

Choi, S., Willett, P., Daum, F., & Huang, J. (2011). Discussion and application of the homotopy filter. *Signal Processing, Sensor Fusion, and Target Recognition XX*, *8050*(May 2011), 805021. https://doi.org/10.1117/12.886385

Daum, Fred, & Huang, J. (2008). Particle flow for nonlinear filters with log-homotopy. In O. E. Drummond (Ed.), *SPIE Defense and Security Symposium* (Vol. 6969, p. 696918). https://doi.org/10.1117/12.764909

Daum, Fred, & Huang, J. (2012). Small curvature particle flow for nonlinear filters. *Signal and Data Processing of Small Targets 2012*, *8393*(May 2012), 83930A-83930A – 11. https://doi.org/10.1117/12.915183

Daum, Fred, & Huang, J. (2013). Zero curvature particle flow for nonlinear filters. In I. Kadar (Ed.), *International Conference on Acoustics, Speech, and Signal Processing* (Vol. 6969, p. 87450Q). https://doi.org/10.1117/12.2009364

Daum, Fred, Huang, J., Krichman, M., & Kohen, T. (2009). Seventeen dubious methods to approximate the gradient for nonlinear filters with particle flow. *Signal and Data Processing of Small Targets 2009*, *7445*(September 2009), 74450V. https://doi.org/10.1117/12.823519

Daum, Fred, Huang, J., & Noushin, A. (2010). Exact particle flow for nonlinear filters. In I. Kadar (Ed.), *Signal Processing, Sensor Fusion, and Target Recognition XIX* (Vol. 7697, p. 769704). https://doi.org/10.1117/12.839590

Daum, Frederick, Huang, J., Noushin, A. J., & Krichman, M. (2009). Gradient estimation for particle flow induced by log-homotopy for nonlinear filters. In I. Kadar (Ed.), *Signal Processing, Sensor Fusion, and Target Recognition XVIII* (Vol. 7336, p. 733602). https://doi.org/10.1117/12.817391

Khan, M. A., & Ulmke, M. (2015). Improvements in the implementation of log-homotopy based particle flow filters. *2015 18th International Conference on Information Fusion (Fusion)*, 74–81.

Li, T., Bolic, M., & Djuric, P. M. (2015). Resampling Methods for Particle Filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, *32*(3), 70–86. https://doi.org/10.1109/MSP.2014.2330626

Li, T., Sun, S., Sattar, T. P., & Corchado, J. M. (2014). Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches. *Expert Systems with Applications*, *41*(8), 3944–3954. https://doi.org/https://doi.org/10.1016/j.eswa.2013.12.031

Li, Y., & Coates, M. (2017). Particle Filtering With Invertible Particle Flow. *IEEE Transactions on Signal Processing*, *65*(15), 4102–4116. https://doi.org/10.1109/TSP.2017.2703684

Musso, C., Oudjane, N., & Gland, F. (2001). Improving Regularised Particle Filters. In *Sequential Monte Carlo Methods in Practice* (pp. 247–271). https://doi.org/10.1007/978-1-4757-3437-9_12

Sasaki, H., Hyvärinen, A., & Sugiyama, M. (2014). Clustering via Mode Seeking by Direct Estimation of the Gradient of a Log-Density. *ECMLPKDD'14: Proceedings of the 2014th European Conference on Machine Learning and Knowledge Discovery in Databases*, 19–34. https://doi.org/10.1007/978-3-662-44845-8_2

Sasaki, H., Noh, Y.-K., Niu, G., & Sugiyama, M. (2016). Direct Density Derivative Estimation. *Neural Computation*, *28*(6), 1101–1140. https://doi.org/10.1162/NECO_a_00835

Schuhmacher, D., Vo, B.-T., & Vo, B.-N. (2008). A Consistent Metric for Performance Evaluation of Multi-Object Filters. *IEEE Transactions on Signal Processing*, *56*(8), 3447–3457. https://doi.org/10.1109/TSP.2008.920469