Analysis of Log-Homotopy Based Particle Flow Filters

MUHAMMAD ALTAMASH KHAN
MARTIN ULMKE
WOLFGANG KOCH

The state estimation plays an important role in analyzing many real world systems. Such systems can be classified into being linear or non-linear, and depending on the statistical properties of the inherent uncertainties as being Gaussian or non-Gaussian. Unlike linear Gaussian systems, a close form estimator does not exist for non-linear/non-Gaussian systems. Typical solutions like EKF/UKF can fail, while Monte Carlo methods even though more accurate, are computationally expensive. Recently proposed log homotopy based particle flow filters, also known as Daum-Huang filters (DHF) provide an alternative way for non-linear, non-Gaussian state estimation. There have been a number of DHF derived, based on solutions of the homotopy flow equation. The performance of these new filters depends strongly on the implementation methodology. In this paper, we study a non-linear system, perturbed by Gaussian and non-Gaussian noises. We highlight the key factors affecting the DHF performance, and investigate them individually in detail. We then make recommendations based on our results. It is shown that a properly designed DHF can outperform a basic particle filter, with less execution time.

Manuscript received December 3, 2015; released for publication October 6, 2016.

Refereeing of this contribution was handled by Ramona Georgescu.

Authors' addess: Sensor Data and Information Fusion, Fraunhofer FKIE, Wachtberg, Germany (E-mail: {altamash.khan, martin.ulmke, wolfgang.koch}@fkie.fraunhofer.de).

I. INTRODUCTION

The Bayesian estimation framework offers an intuitive way for the estimation of hidden states of a dynamical system based on the observational data. The Bayesian estimation is carried out recursively, typically consisting of a prediction and a correction step. A transition density describes the time evolution of the state conditioned on the previous values, while a measurement density describes the likelihood of measurements given the current state. These densities are then used recursively for the evaluation of prior and posterior state distributions at any given moment of time. The process is known as recursive Bayesian estimation (RBE) and arises in many real scenarios. Finite dimensional analytical solutions to the RBE problem are available only in few cases, mainly when the system model is linear Gaussian (Kalman filter) or a finite state Hidden Markov model (HMM) [1]. Traditional methods for non-linear state estimation include Extended (EKF) and Unscented Kalman filter (UKF). However these methods are generally sub-optimal and their performance degrades with the increase in the non-linearity, and also when the transition and measurement densities are non-Gaussian (e.g. multimodal, exponential).

Particle filters, also known as sequential Monte Carlo (SMC) methods, provide an alternative way to the state estimation. The main idea is to represent the posterior density by a weighted set of random samples (particles), which are then used to form the point estimates, e.g., mean and variance [2]. The posterior density under these settings approximately represents the path distribution, i.e., distribution of the state through the time, conditioned on the measurements. Several version of particle filters have been proposed in the literature, e.g., sampling importance resampling (SIR) filter also known as bootstrap particle filter [3], auxiliary sampling importance resampling (ASIR) filter [4], regularized particle filter (RPF) [5] etc. While particle filters can effectively deal with the non-linearities and non-Gaussian noises, they suffer from the so called *weight degeneracy* and *curse of dimensionality*. Weight degeneracy refers to the fact that after few updates all but one particle have negligable weights. Weight degeneracy occurs when the target distribution does not significantly overlap with the prior distribution. Several solutions have been proposed to address these problem e.g. re-sampling, the use of Markov Chain Monte Carlo (MCMC) methods, use of bridging densities as suggested in [6] and [7]. Bridging densities are obtained by varying the so called *progression parameter*, which corresponds to the gradual introduction of the measurements. In this manner the posterior density can be better approximated. On the other hand, the curse of dimensionality means that to maintain a certain performance level, the required number of particles increases exponentially with the increase in the state dimension, as reported in [8].

A different approach to non-linear filtering has been suggested by Daum and Huang in a series of papers [9]–[15], which is based on the gradual inclusion of the measurements. The key idea is to model the transition of particles from the prior to the posterior density as a physical flow under the influence of an external force (measurements). Particles are sampled from the state transition density and a notion of synthetic time also called the pseudo-time is introduced, in which particles flow until they reach *correct* posterior locations. A stochastic differential equation (SDE) define the flow of particles in pseudo-time, while the Fokker-Planck equation (FPE) describes the density evolution. A flow vector is obtained by solving the FPE under different assumptions, which is then integrated numerically yielding updated states of particles. The new filter is termed as homotopy based particle flow filter or simply Daum-Huang filter (DHF) after the developers. Different flow solutions have been derived, including the incompressible flow [9], zero diffusion exact flow [10], Coulomb's law flow [11] and zero-curvature flow [12] non zero diffusion flow [13].

DHF implementations have been reported in several publications. While conceptually being quite intuitive, DHF performance suffers in practice due to several assumptions, made both in the theory and the implementation. In this paper we identify key factors affecting the performance of the DHF. We study each of those factors in detail and in the light of the results, we suggest possible improvements in the DHF implementation. We consider state estimation of a non-linear system under both Gaussian and non-Gaussian measurement noises. The effect of different methods on the performance of DHF is studied individually for both noise cases. We show that by a careful design, the DHF performance can be substantially improved over the more traditional implementations.

The outline of the paper is given as follows: We present a description of homotopy based particle flow in section II. We start with the general formulation of RBE. We then give a derivation of the generic homotopy based flow equation, which is followed by its specific solutions. Next, in the section III we present a generic algorithm for DHF implementation and highlight the important steps. We describe different possible schemes that could be employed for each of those steps in the section IV. Section V starts with the description of the two models used in the study, followed by the subsection on the parameter settings and the simulation methodology. Results for proposed alternative methods are described in section VI, which is followed by the discussion in section VII. Finally the conclusion is given in section VIII.

II. HOMOTOPY BASED PARTICLE FLOW FILTERS

A. Bayesian recursive estimation

We start with the general formulation of bayesian recursive estimation for a markovian state space system. Let $\mathbf{x}_k \in \mathbb{R}^d$ denote the state vector and $\mathbf{z}_k \in \mathbb{R}^m$ denote the measurement vector at time k. Also let $\mathbf{Z}_k$ denote the set of measurements up to time k including $\mathbf{z}_k$, $\mathbf{Z}_k = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_k\}$. The state space model can be expressed in the terms of conditional probabilities,

$$\mathbf{x}_{k+1} \sim p(\mathbf{x}_{k+1} \mid \mathbf{x}_k) \tag{1}$$

$$\mathbf{z}_{k+1} \sim p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1}) \tag{2}$$

$p(\mathbf{x}_{k+1} \mid \mathbf{x}_k)$ and $p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1})$ are referred to as the transition and the measurement/likelihood densities. Assuming additive process and measurement noises w_k and v_k we can write

$$p(\mathbf{x}_{k+1} \mid \mathbf{x}_k) = p_{w_k}(\mathbf{x}_{k+1} - \phi_k(\mathbf{x}_k)) \tag{3}$$

$$p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1}) = p_{v_k}(\mathbf{z}_{k+1} - \psi_k(\mathbf{x}_{k+1})) \tag{4}$$

where ϕ_k is termed as the process/dynamical model and ψ_k as the measurement model. According to the Chapman-Kolmogrov equation and the Bayes theorem, the prior density $p(\mathbf{x}_{k+1} \mid \mathbf{Z}_k)$ and the posterior density $p(\mathbf{x}_{k+1} \mid \mathbf{Z}_{k+1})$ are recursively defined as,

$$p(\mathbf{x}_{k+1} \mid \mathbf{Z}_k) = \int p(\mathbf{x}_{k+1} \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{Z}_k) d\mathbf{x}_k \tag{5}$$

$$p(\mathbf{x}_{k+1} \mid \mathbf{Z}_{k+1}) = \frac{p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1}) p(\mathbf{x}_{k+1} \mid \mathbf{Z}_k)}{p(\mathbf{z}_{k+1} \mid \mathbf{Z}_k)} \tag{6}$$

where $p(\mathbf{x}_k \mid \mathbf{Z}_k)$ is posterior density at time k. These are also referred to as the process and measurement update equations respectively. The conditional density $p(\mathbf{z}_{k+1} \mid \mathbf{Z}_k)$ appears as a normalization constant in the measurement update formula, and it describes the distribution of measurement at time $k+1$, conditioned on the set of all previous measurements. An exact closed form solution of (5) and (6) is generally not available for non-linear systems. Instead, two main approximate methods are used for the state estimation of such systems. In a first approach, the linearization of the model is performed around the current estimate (EKF) or the so called *Sigma-points* are propagated through the non-linear state space (UKF), thereby providing an approximation to the point estimates e.g. state mean and covariance. Another approach could be to numerically approximate the process and the measurement update equation. This could be done either by numerically evaluating the integrals over the discretized state space region [14], or by employing sequential Monte Carlo methods like particle filters [2], [3].

B. Derivation of generic homotopy flow equation

Log homotopy based particle flow filters also termed as the Daum-Huang particle flow filters (or simply the Daum-Huang filters DHF) as described in [9]–[15], share the importance sampling step with the SIR particle filter, but they specifically use the prior distribution of the state vector $p(\mathbf{x}_{k+1} \mid \mathbf{x}_k)$ as the importance density. The main difference lies in the way measurements are

incorporated to derive the posterior density. The idea here is to model the motion of particles from the prior to the posterior density, in a way analogous to the flow of physical particles. A log-homotopy function $\log p(\mathbf{x}_k, \lambda)$ is defined through the homotopy parameter λ,

$$\log p(\mathbf{x}_{k+1}, \lambda) = \log g(\mathbf{x}_{k+1}) + \lambda \log h(\mathbf{x}_{k+1}) - \log K(\lambda). \tag{7}$$

where $g(\mathbf{x}_{k+1})$ represents the prior $p(\mathbf{x}_{k+1} \mid \mathbf{Z}_k)$, $h(\mathbf{x}_{k+1})$ the likelihood $p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1})$ and λ the pseudo-time varying from 0 to 1. $K(\lambda)$ is the normalization constant independent of $\mathbf{x}_{k+1}$. $\lambda = 0$ sets $p(\mathbf{x}_{k+1}, \lambda)$ equal to the prior density while with $\lambda = 1$ the transformation is completed to the normalized posterior density. From now on we drop the time index k for the sake of convenience. It is supposed that the flow of particle obeys the Itô SDE,

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, \lambda) d\lambda + \boldsymbol{\sigma}(\mathbf{x}, \lambda) d\mathbf{w} \tag{8}$$

where $\mathbf{f}(\mathbf{x}, \lambda)$ is the flow vector, $\mathbf{w}$ is the M-dimensional Wiener process with diffusion matrix $\boldsymbol{\sigma}(\mathbf{x}, \lambda)$. For a flow characterized as in (8), the evolution of the density $p(\mathbf{x}, \lambda)$ w.r.t. the parameter λ is given by the Fokker-Planck equation (also known as Kolmogorov forward equation),

$$\frac{\partial p(\mathbf{x}, \lambda)}{\partial \lambda} = -\sum_{i=1}^{d} \frac{\partial}{\partial \mathbf{x}_i} [f_i(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda)] + \frac{1}{2} \sum_{i=1}^{d} \sum_{j=1}^{d} \frac{\partial^2}{\partial \mathbf{x}_i \partial \mathbf{x}_j} [\mathbf{Q}_{i,j}(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda)] \tag{9}$$

where $\mathbf{Q}(\mathbf{x}, \lambda)$ is the diffusion tensor. This can be written in short hand notion,

$$\frac{\partial p(\mathbf{x}, \lambda)}{\partial \lambda} = -\nabla \cdot (\mathbf{f}(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda)) + \frac{1}{2} \nabla^T \mathbf{Q}(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda) \nabla \tag{10}$$

where ∇ is the spatial vector differentiation operator. From (7), the pseudo-time derivative of the density $p(\mathbf{x}, \lambda)$ can be formulated.

$$\frac{\partial p(\mathbf{x}, \lambda)}{\partial \lambda} = p(\mathbf{x}, \lambda) \left(\log h(\mathbf{x}) - \frac{\partial \log K(\lambda)}{\partial \lambda} \right) \tag{11}$$

By combining equations (10) and (11) we get,

$$p(\mathbf{x}, \lambda) \left(\log h(\mathbf{x}) - \frac{\partial \log K(\lambda)}{\partial \lambda} \right) = -\nabla \cdot (\mathbf{f}(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda)) + \frac{1}{2} \nabla^T \mathbf{Q}(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda) \nabla \tag{12}$$

Using the vector calculus identity,

$$\nabla \cdot (\mathbf{a}\mathrm{b}) = (\nabla \cdot \mathbf{a})\mathrm{b} + \mathbf{a} \cdot (\nabla \mathrm{b})$$

the equation 12 can be further expanded,

$$\log h(\mathbf{x}) - \frac{\partial \log K(\lambda)}{\partial \lambda} = -\mathbf{f}^T(\mathbf{x}, \lambda) \cdot \nabla \log p(\mathbf{x}, \lambda) - \nabla \cdot \mathbf{f}(\mathbf{x}, \lambda) + \frac{1}{2 p(\mathbf{x}, \lambda)} (\nabla^T \mathbf{Q}(\mathbf{x}, \lambda) p(\mathbf{x}, \lambda) \nabla) \tag{13}$$

The objective then becomes to solve the generic flow equation (13) for the yet unknown flow $\mathbf{f}(\mathbf{x}, \lambda)$.

C. Specific flow solutions

Various flow solutions have been obtained by solving (13) under different assumptions. Here we discuss four such flows derived by F. Daum and J. Huang in their series of papers.

1) Incompressible flow: The first solution of (13) appeared in [9], which was based on two distinct assumptions. Firstly, the diffusion term $\sigma(\mathbf{x}, \lambda)$ in (8) is ignored. Secondly, the flow is considered incompressible, i.e. $\nabla \cdot \mathbf{f}(\mathbf{x}, \lambda) = 0$. Also the derivative of the log of normalization constant $\partial \log K(\lambda)/\partial \lambda$ is assumed to be very small, and therefore neglected. Applying these simplifications to the (13) leads to the incompressible flow equation,

$$\mathbf{f}^T(\mathbf{x}, \lambda) \cdot \nabla \log p(\mathbf{x}, \lambda) = -\log h(\mathbf{x}) \tag{14}$$

For one dimensional state space (d= 1) the equation has an exact solution. However, for ($\mathrm{d} \geq 2$) a simple inversion of the vector $\nabla(\log p(\mathbf{x}, \lambda))$ is not possible. Instead a unique minimum norm solution is obtained using the generalized inverse,

$$\mathbf{f}(\mathbf{x}, \lambda) = -\log h(\mathbf{x}) \frac{\nabla \log p(\mathbf{x}, \lambda)}{\|\nabla \log p(\mathbf{x}, \lambda)\|^2} \tag{15}$$

The authors suggest to use the fast kNN method to evaluate the gradients. Implementational details are described in [16]. Incompressible flow is generally inferior to exact flow [17]. Also it was reported in [18] that the incompressible flow could often hit a singularity for d=1. As a rebuttal to this, in [19] it has been argued that the incompressible flow can avoid singularities for $\mathrm{d} \geq 2$, as singularities in higher dimension are just points in the state space and hence they can be bypassed/flown around. In the current work we will consider the incompressible flow for the sake of comparison. The filter based on the incompressible flow is termed as DHF-IC.

2) Exact flow: If the diffusion term is still assumed to be zero and $\partial \log K(\lambda)/\partial \lambda$ is neglected but the flow is allowed to be compressible, following equation can be derived from (13)

$$\log h(\mathbf{x}) + \mathbf{f}^T(\mathbf{x}, \lambda) \cdot \nabla \log p(\mathbf{x}, \lambda) = -\nabla \cdot \mathbf{f}(\mathbf{x}, \lambda) \tag{16}$$

Different flows have been derived in [20] based on solutions to the (16). One particular solution relates to

the case of $\log g(\mathbf{x})$ and $\log h(\mathbf{x})$ being bilinear in the components of vector$\mathbf{x}$, e.g., assuming a Gaussian prior and likelihood.

$$\log g(\mathbf{x}) = \log c_g - \tfrac{1}{2}(\mathbf{x}-\bar{\mathbf{x}})^T\mathbf{P}^{-1}(\mathbf{x}-\bar{\mathbf{x}}) \quad (17)$$

$$\log h(\mathbf{x}) = \log c_h - \tfrac{1}{2}(\mathbf{z}-\psi(\mathbf{x}))^T\mathbf{R}^{-1}(\mathbf{z}-\psi(\mathbf{x})) \quad (18)$$

where $\log c_g$ and $\log c_h$ are the associated log normalization constants. The gradient of the two densities then can be written as,

$$\nabla \log p(\mathbf{x},\lambda) = -\mathbf{P}^{-1}(\mathbf{x}-\bar{\mathbf{x}}) + \lambda\mathbf{H}^T\mathbf{R}^{-1}(\mathbf{z}-\psi(\mathbf{x})) \quad (19)$$

where $\mathbf{H} = (\partial\psi/\partial\mathbf{x})|_{\mathbf{x}_\lambda}$. More generally speaking, if the additive noise processes w_k and v_k belong to the exponential family, then an analytical solution termed as the *Exact flow* can be derived. For the Gaussian case, this is given as,

$$\mathbf{f}(\mathbf{x},\lambda) = \mathbf{A}(\lambda)\mathbf{x} + \mathbf{b}(\lambda) \quad (20)$$

where $\mathbf{A}(\lambda)$ and $\mathbf{b}(\lambda)$ are,

$$\mathbf{A}(\lambda) = -\tfrac{1}{2}\mathbf{P}\mathbf{H}^T(\lambda\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}\mathbf{H} \quad (21)$$

$$\mathbf{b}(\lambda) = (I + 2\lambda\mathbf{A})[(I + \lambda\mathbf{A})\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + \mathbf{A}\bar{\mathbf{x}}] \quad (22)$$

For nonlinear systems, the measurement model can be linearized by the Taylor series expansion up to the first term, such that $\mathbf{z} \approx \mathbf{z} - \psi(\mathbf{x}_\lambda) + \mathbf{H}\mathbf{x}_\lambda$. The derivation of the exact flow has been described in detail in [21]. We abbreviate this filter type as DHF-EF.

3) Coulomb′s law based flow: Yet another solution can be developed in which the flow of particles in the pseudo-time is derived from the gradient of Poisson's equation [11]. Diffusion term in (13) is again assumed to be zero, but the derivative of the normalization constant is not ignored. Instead it is derived and an exact expression is found,

$$\frac{\partial \log K(\lambda)}{\partial\lambda} = \mathbb{E}(\log(h(\mathbf{x}))) \quad (23)$$

Then the equation (13) is written in the form

$$\nabla\cdot\mathbf{q}(\mathbf{x},\lambda) = -\eta(\mathbf{x},\lambda) \quad (24)$$

where $\mathbf{q}(\mathbf{x},\lambda) = \mathbf{f}(\mathbf{x},\lambda)p(\mathbf{x},\lambda)$ and $\eta(\mathbf{x},\lambda) = -p(\mathbf{x},\lambda)\cdot(\log h(\mathbf{x}) - \partial\log K(\lambda)/\partial\lambda)$ It is noticed that the integral of $\eta(\mathbf{x},\lambda)$ w.r.t. x along the flow is zero,

$$\int_\Omega \eta(\mathbf{x},\lambda) = 0 \quad (25)$$

where Ω is the relevant volume of the state space. This is analogous to the zero divergence of the electric flux density out of an enclosed region without any charge (first of the Maxwell's equations), i.e. net field lines entering the an enclosed region equal to the those leaving. Next it is reasoned that, if the function $\mathbf{q}(\mathbf{x},\lambda)$ can be assumed to be the gradient of scalar potential function $V(\mathbf{x},\lambda)$, then the equation (24) can be expressed as the Poisson's equation for the potential $V(\mathbf{x},\lambda)$.

$$\Delta V(\mathbf{x},\lambda) = \eta(\mathbf{x},\lambda) \quad (26)$$

such that,

$$f(\mathbf{x},\lambda) = \frac{\nabla V(\mathbf{x},\lambda)}{p(\mathbf{x},\lambda)} \quad (27)$$

where Δ is the Laplacian operator. Solution to the (26) can be expressed in terms of the convolution integral for $\mathrm{d} \geq 3$,

$$V(\mathbf{x},\lambda) = -\int_\Omega \eta(\mathbf{y},\lambda)\frac{c}{\|\mathbf{x}-\mathbf{y}\|^{d-2}}d\mathbf{y} \quad (28)$$

where $c = (4\pi)^{-d/2}\Gamma((d/2)-1)$ and $\mathbf{y}$ is the running variable. The above equation gives the solution of scaler potential $V(\mathbf{x},\lambda)$, whereas our quantity of interest is its gradient. Taking the gradient of (28) we get,

$$\nabla V(\mathbf{x},\lambda) = \mathbb{E}\left[(\log h(\mathbf{y}) - \mathbb{E}[\log h(\mathbf{x})])\frac{c(2-d)(\mathbf{x}-\mathbf{y})^T}{\|\mathbf{x}-\mathbf{y}\|^d}\right] \quad (29)$$

Using (23) together with the Monte Carlo approximation for integrals, (29) can be approximated as,

$$\nabla V(\mathbf{x}_i,\lambda) \approx \frac{1}{k}\sum_{j\in S_i}\left(\log h(\mathbf{x}_j) - \frac{1}{k}\sum_{l\in S_l}\log h(\mathbf{x}_l)\right)\cdot \left(\frac{c(2-d)(\mathbf{x}_i-\mathbf{x}_j)^T}{\|\mathbf{x}_i-\mathbf{x}_j\|^d + \alpha}\right) \quad (30)$$

The expression for the gradient $\nabla V(\mathbf{x},\lambda)$ is similar to the electromagnetic force equation given by Coulomb's law, hence the name of the flow. In order to reduce the computational complexity, the outer summation is carried over the subset of k nearest neighbors of the ith particle $\mathbf{x}_i$, which is denoted here by S_i. This is motivated by the fact that the as the state space dimension is increased, contribution of particles far apart, approaches zero exponentially. The inner expectation is approximated in a similar way. α is set to $(1/\beta)Tr(P)^{d/2}$, where both α and β are design parameters. Their purpose is to regularize the expression for $\nabla V(\mathbf{x},\lambda)$. Also P can be approximated by a prior covariance matrix estimate. Filter based on this type of flow is referred as DHF-CLF

4) Non zero diffusion constrained flow: The last type of flow we consider, can be derived by not ignoring the diffusion term in equation (13) as suggested in [13]. Taking the gradient we get,

$$\begin{aligned}\nabla\log h(\mathbf{x}) = &-\nabla\log p(\mathbf{x},\lambda)^T\cdot\nabla\mathbf{f}(\mathbf{x},\lambda)\\ &-\mathbf{f}^T(\mathbf{x},\lambda)\cdot\nabla^2\log p(\mathbf{x},\lambda) - \nabla(\nabla\cdot\mathbf{f}(\mathbf{x},\lambda))\\ &+\nabla\left(\frac{1}{2p(\mathbf{x},\lambda)}\nabla^T\mathbf{Q}(\mathbf{x},\lambda)p(\mathbf{x},\lambda)\nabla\right)\end{aligned} \quad (31)$$

Analytical evaluation of the above equation for the flow $\mathbf{f}(\mathbf{x}, \lambda)$ is not possible, though numerical methods can be employed for this purpose. Depending on the dimensionality of the state-space, this could be computationally quite demanding. A little trick can lead to closed form solution for the flow, if the following constraint holds valid,

$$\nabla\left(\frac{1}{2p(\mathbf{x},\lambda)}\nabla^T\mathbf{Q}(\mathbf{x},\lambda)p(\mathbf{x},\lambda)\nabla\right) = \nabla\log p(\mathbf{x},\lambda)^T\cdot\nabla\mathbf{f}(\mathbf{x},\lambda) + \nabla(\nabla\cdot\mathbf{f}(\mathbf{x},\lambda)) \tag{32}$$

This resuults in a simple formula for the flow equation, given by

$$\mathbf{f}(\mathbf{x},\lambda) = -(\nabla^2\log p(\mathbf{x},\lambda))^{-1}(\nabla\log h(\mathbf{x}))^T \tag{33}$$

The flow derivation does not involve neglecting the diffusion term, instead it appears in the constraint equation. Hence this flow is termed as non-zero diffusion constrained flow (NZDCF), and the DHF with this particular flow is termed as DHF-NZDCF.

A closer look at (33) reveals that it requires the hessians of the log prior and the likelihood, as well as the gradient of the log-likelihood. The gradient and the hessian of the log-likelihood, $\nabla\log h(\mathbf{x})$ and $\nabla^2\log h(\mathbf{x})$, can be calculated analytically in most cases. On the other hand, there is no single method for the evaluation of the hessian of the prior density, $\nabla^2\log g(\mathbf{x})$. The most straight forward method is to approximate the prior density by a multivariate gaussian density (e.g. using Laplace approximation), and use the negative of the inverse of the covariance matrix $-\mathbf{P}^{-1}$. This leads to the following,

$$\nabla^2\log p(\mathbf{x},\lambda) = \nabla^2\log g(\mathbf{x}) + \lambda\nabla^2\log h(\mathbf{x}) \tag{34}$$

$$\approx -\mathbf{P}^{-1} + \lambda\nabla^2\log h(\mathbf{x}) \tag{35}$$

It has been suggested that the matrix $\mathbf{P}$ can be set to the prior covariance matrix of a parallel running EKF/UKF. Another suggested method is to use the fast k-NN algorithm to compute the hessian of the prior, similar to the incompressible flow. Alternatively, an approximation can be used instead, where $\mathbf{P}$ is the state covariance matrix computed directly from the prior position of particles.

In this work, we primarily focus on analyzing DHF-NZDCF.

III. IMPLEMENTATION OF PARTICLE FLOW FILTERS

Numerical results for the DHF have been presented in [17]. DHF based on the incompressible and exact flows have been implemented by Choi. et.al. in [16] for non-linear scalar and linear vector system models. Exact flow DHF implementations for multi-target tracking using acoustic measurements have been reported in [21], where mobile targets are tracked based on the their received signal strength at a fixed receivers. In [22], joint probabilistic data association (JPDA) and maximum aposteriori penalty function (MAP-PF) algorithms based on the exact flow DHF have been derived. Recently, many researchers have carried out the comparitive analysis for the DHF-NZDCF, in quite varied application. This include comparing the DHF performance against more traditional methods for angle only filtering in 3D by Gupta et.al. [23], comparing the tracking performance of DHF vs. other methods for supermaneuverable targets by Kreucher et.al. in [24], and the comparison of multisensor fusion using DHF against the particle filters by Mostagh and Chan in [25]. Results show a varying degree of success for DHF. While in some applications particle flow filters are shown to outperform the competitors, in others they do not perform quite well. The main issue is that while particle flow filters are theoretically quite elegant, their performance suffer from approximations made, both in theory and in the practical implementation. This include approximations made while deriving the flow, estimation of the prior density and the use of numerical techniques. This leads to the introduction of bias and loss of asymptotic consistency [26].

There could be several ways in which a DHF can be implemented. In Algorithm 1, we outline the method described by T. Ding and M. Coates in [21].

ALGORITHM 1: *Generic implementation of DHF*

```
Initialize DHF: Generate initial set of particles;
Initialize EKF/UKF: Initial mean and covariance;
Pseudo-time grid discretization;
for Loop over the time do
    Propagate particles using the dynamic model;
    Time update for EKF/UKF;
    Prior covariance matrix estimate from EKF/UKF;
    for Loop over the pseudo-time do
        for Loop over individual particles do
            Integration of the flow equation;
        end
    end
    Measurement Update for EKF/UKF;
    Redraw particles (Optional);
end
```

Particles are generated by sampling the transition density. An EKF/UKF is run in parallel to the main algorithm. This is done to approximate the prior covariance matrix. Next the flow equation is solved in the pseudo-time for all particles. The flow equation uses the prior covariance estimate from the parallel running EKF/UKF. Once done, the mean state vector is estimated and the measurement update is carried out for EKF/UKF. This process in repeated till the end of the simulation time. The steps colored in red are the crucial factors in the performance of the DHF.

The first is the pseudo-time λ discretization strategy together with the numerical integration method. As the DHF flow is described by an ordinary differential equation (ODE), a suitable discretization is essential to capture the flow dynamics. Then the flow equation is integrated w.r.t. λ. While the exact implementation details for references [13], [16],[22] are not clear, authors in [21] have used a single step Euler integration, as mentioned in the pseudo-code. It is simple to implement and is fairly quick. But care has to be taken as the flow ODE can exhibit stiffness. In that case a straight forward λ discretization together with the single step Euler integration might not work. Secondly, all flows described above require an estimate of the prior covariance matrix. While prior covariance estimate from parallel running EKF/UKF can be used as an approximation, this makes the DHF accuracy dependant on EKF/UKF. On the other hand, a sample covariance estimate can often be ill-conditioned. The question then becomes, is there a better method to estimate the prior covariance matrix. Finally, the re-generation of a new set of particles is an important step. Unlike a standard particle filter, the re-sampling/re-drawing step is not mandatory in the DHF, but optional. Instead, it has been mentioned that the homotopy flow moves the particles to their correct locations in the state space. But due to approximations made in the derivations, the flow may not be accurate, which could could reduce the accuracy of the estimates. Hence the effect of particle re-generation is worth investigating. In the current work we look for improvements in the DHF performance by considering changes in the existing implementation architecture, as mentioned in Algorithm 1.

IV. IMPORTANT FACTORS IN DHF

In this section we individually discuss the aforementioned key factors affecting the DHF performance.

A. Pseudo-time discretization

While comparing the two flows, it was shown in [27] that the non zero diffusion flow is considerably stiffer as compared to the exact flow, where authors used 39 exponentially spaced λ points for solving the ODEs. This has also been mentioned in [13] where the usage of exponentially spaced time steps or higher order integration schemes is recommended to solve the issue. In this paper we consider both uniform and non-uniform grid discretization. The idea is to analyze the effect of a particular grid discretization strategy and the numerical integration scheme on the filter performance, in terms of the estimation error and the processing time. While the coarse λ discretization would not result in the correct solution, a fine discretization on the other hand would lead to a substantial increase in the computational cost. Therefore, a middle ground has to be chosen such that, flow dynamics at very small λ values are maximally captured, while only moderately increasing the processing cost.

B. ODE numerical solution

The homotopy flow is defined by a vector ordinary differential equation (ODE). In the current work, we seek for the numerical solution of the ODE. Broadly speaking, ODEs can be catagorized into being stiff and non-stiff. While there is no precise definition of the stiffness, in the literature two criteria are generally mentioned for describing a stiff ODE. First, the condition number of the jacobian matrix $\mathbf{J}(x,\lambda) = \partial\mathbf{f}(x,\lambda)/\partial x$ of a stiff ODE is quite large. As a consequence, multiple timescales exist in the ODE. Time scales, often referred as modes, are defined by the inverse absolute eigenvalues of the Jacobian $J(x,\lambda)$. Secondly, in the Lipschitz's inequality $\|f(x_2,\lambda_2) - f(x_1,\lambda_1)\| \leq L\|\lambda_2 - \lambda_1\|$, the Lipschitz's constant L is typically very high for a stiff ODE. Non-zero diffusion ODE can be characterized as stiff according to both criteria. Therefore, care has to be taken when chosing the numerical integration scheme for solving the flow ODE.

The standard Euler's method has been used for solving the flow ODE in earlier works. It is a first order method with the truncation error in the order of $\mathcal{O}(h^2)$. In this paper, we intend to compare the performance of some other numerical integration (NI) schemes for solving stiff ODEs alongside the Euler's method. There are several choices available. Below, we mention some of the common NI methods for solving stiff ODEs.

1) Forth order Runge-Kutta method: Forth order Runge-Kutta method (RK4) is our second integration method. RK4 method has the local truncation error in the order of $\mathcal{O}(h^5)$, while the total accumulated error is of order $\mathcal{O}(h^4)$.

2) Rosenbrock method: Rosenbrock methods are family of multistep procedures to solve stiff ODEs. Jacobian matrix appears in the integration formula. Like the Runge-Kutta methods, Rosenbrock methods successively form intermediate results. If the Jacobian matrix is ignored then the method turns into the explicit Runge-Kutta scheme. Therefore, they are also called Runge-Kutta-Rosenbrock methods. Rosenbrock methods preserve exact conservation properties due to the use of the analytic Jacobian matrix, and possess optimal linear stability properties for stiff problems.

3) Gear′s method: The Gear's method [28] belongs to the class of methods known as backward differentiation formulae (BDF). It is an implicit integration method and uses the first and higher order derivatives. Also, it is a predictor-corrector type scheme where each time step is initiated by prediction. Corrector iterations are then carried until prescribed convergence criteria are achieved or non-convergence is deemed to have occurred.

C. Prior covariance shrinkage estimation

The evaluation of the flow equation (33) require the availability of the prior covariance estimate. This

can be derived in several ways. The simplest way is to estimate the covariance matrix using the prior particles. This is referred to as the sample covariance estimate S. S is an unbiased estimator of the true prior covariance P, and is also the maximum likelihood estimate if the data is Gaussian distributed. But for non-linear models/non-Gaussian noises, the Gaussian assumption may not remain valid. Also S could progressively get ill-conditioned. i.e. the spread of the eigenvalues gets larger with the passage of time. This is especially the case, when the d/N_p ratio is non-negligable, where d is the state vector dimension and N_p is the number of particles. As a consequence, the matrix inversion could lead to stability issues. For the case of $d > N_p$, the resulting covariance matrix is not even full rank and hence not invertible. An alternative method suggested by authors in [15] is to run an EKF/UKF in parallel to DHF, and to use the prior covariance matrix generated by those filters. We refer to such matrix as P_{XKF}, where XKF could be Extended or Unscented version of the KF. While this method is better than using the raw data based covariance estimate, it ties the DHF estimation accuracy to that of the EKF/UKF. P_{XKF} could also exhibits a wide spread of the eigenvalues.

Therefore, we look for an alternative method for covariance matrix estimation. That method should have two properties: the resulting matrix should always be positive definite (PD) and the matrix should be well-conditioned [29]. One approach could be to start with the sample covariance, and ensure that the matrix is always PD. Such a matrix might not be well-conditioned. Alternatively, variance reduction technqiues could be used to get a well-conditioned matrix, but this could be computationally expensive [30]. There is another approach used in the multivariate statistics literature for the estimation of the covariance matrices, known as the *shrinkage estimation*. The use of such methods dates back to work of Stein [31]. The main idea is to merge the raw estimate (S) which is unbiased but normally with high variance, together with a more structured but typically a biased target (B) through a scale factor, to get the combined estimate (P_*). The objective is to reduce the estimation error, typically in mean squared sense, by achieving an optimal trade off between the biased (B) and the unbiased (S) estimators. The scale factor is also called shrinkage intensity ρ as it shrinks the eigenvalues of S optimally towards the mean of eigenvalues of the true covariance matrix P [32]. The resulting covariance matrix (P_*)will be biased, but will improve on the two aforementioned properties, and is hoped to lower the estimation error. There are several shrinkage estimators mentioned in the literature, with different target covariance matrices. In the current work, we describe some of the more established shrinkage estimators. In subsections IV-C.1 to IV-C.3, shrinkage estimators are defined through a convex combination of the matrices B and S. The objective becomes to find an optimal shrinkage intensity that minimizes the cost function,

$$\min_{\rho} E[\|P_* - P\|^2] \tag{36}$$

where $P_* = \rho B + (1-\rho)S$.

1) Shrinkage towards the Identity matrix: Shrinkage towards the Identity matrix is described in [32]. The two main objectives defined are, to get an asymptotically consistent estimator that is more accurate than the sample covariance matrix S, and is also well-conditioned. No prior structure is assumed for the target matrix B, as it could lead to an increased biasness. Instead a simple matrix with same covariance terms and zero cross-variances (scaled Identity) is chosen as the target. The shrinkage estimator has following form

$$P_* = \frac{\alpha^2}{\delta^2}\mu^2 I + \frac{\beta^2}{\delta^2} S \tag{37}$$

The estimator P_* asymptotically shrinks its eigenvalues towards the mean eigenvalue of the true covariance matrix P, in quadratic mean sense. The terms α, β, δ and μ depend on the unobserved true covariance matrix P. Therefore, a consistent estimator of P_* is derived under the assumptions of general asymptotics. We term this estimator estimator is termed here as P_{LW0}, and has following form,

$$P_{LW0} = \frac{a_n^2}{d_n^2} m_n I + \frac{b_n^2}{d_n^2} S \tag{38}$$

where,

$$m_n = tr(S)/d$$
$$d_n^2 = \|S - m_n I\|^2 \tag{39}$$
$$xinonumber \tag{40}$$
$$\bar{b}_n^2 = \frac{1}{n^2}\sum_{i=1}^{N_p}[\|x_i x_i^T - S\|^2]$$
$$b_n^2 = \min(\bar{b}_n^2, d_n^2)$$
$$a_n^2 = d_n^2 - b_n^2$$

$\|.\|$ is the squared Frobenius norm and x_i is the ith particle. Also the shrinkage intensity ρ is given by a_n^2/d_n^2. It is shown that the MSE for P_{LW0} asymptotically approaches that of P_* i.e. $\lim_{N_p\to\infty} E[\|P_* - P\|_{N_p}^2] - E[\|P_{LW0} - P\|_{N_p}^2] \to 0$. One main advantage of this estimator is that it does not assume any particular distribution for the data, and is therefore *distribution-free*.

2) Shrinkage towards the constant correlation matrix: This estimator is derived in [33], in the context of portfolio optimization. The target matrix is chosen according to the constant correlation model. It means that pairwise correlations are identical, which is given by the

average of all the sample correlations. We denote this estimator by P_{LW1}. The target matrix B is given by

$$B = \begin{cases} S_{ii}: & i = j \\ \bar{r}\sqrt{S_{ii}S_{jj}}: & i \neq j \end{cases}$$

where $\bar{r}$ is the average sample correlation. It is defined as

$$\bar{r} = \frac{2}{d(d-1)} \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \frac{S_{ij}}{\sqrt{S_{ii}S_{jj}}} \tag{41}$$

The shrinkage intensity is defined as $\rho = \max\{0, \min\{1, \kappa/d\}\}$, with $\kappa = (\hat{\pi} - \hat{\varrho})/\hat{\gamma}$. $\hat{\pi}$ denotes the sum of asymptotic variances of the entries of the sample covariance matrix S, while $\hat{\varrho}$ denotes the sum of asymptotic covariances of the entries of the shrinkage target B with the entries of the sample covariance matrix. $\hat{\gamma}$ gives a measure of the misspecification of the shrinkage target. The hat ($\hat{\cdot}$) on the top of terms indicate the fact that these are the estimates of the true values, which are not known. $\hat{\pi}$ are $\hat{\varrho}$ are given by,

$$\begin{aligned} \hat{\pi} &= \frac{1}{d} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{d} \{(x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j) - s_{ij}\}^2 \\ \hat{\varrho} &= \sum_{i=1}^{n} \hat{\pi}_{ii} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{\bar{r}}{2} \left(\sqrt{\frac{S_{jj}}{S_{ii}}} \vartheta_{ii,ij} + \sqrt{\frac{S_{ii}}{S_{jj}}} \vartheta_{jj,ij} \right) \end{aligned} \tag{42}$$

where,

$$\begin{aligned} \vartheta_{ii,ij} &= \frac{1}{d} \sum_{k=1}^{d} \{(x_{ik} - \bar{x}_i)^2 - \bar{x}_i) - S_{ii}\} \\ &\quad \cdot \{(x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j) - S_{ij}\} \\ \vartheta_{jj,ij} &= \frac{1}{d} \sum_{k=1}^{d} \{(x_{jk} - \bar{x}_j)^2 - \bar{x}_j) - S_{jj}\} \\ &\quad \cdot \{(x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j) - S_{ij}\} \end{aligned} \tag{43}$$

Finally $\hat{\gamma}$ is given by

$$\hat{\gamma} = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} (B_{ij} - S_{ij})^2 \tag{44}$$

3) Shrinkage towards the perfect postive correlation matrix: Authors in [34] suggest single-factor matrix as the shrinkage target. The paper is concerned with estimating the structure of the risk in the stock market and the modelling of the stock returns. The fact that stock returns are positively correlated to each other, is exploited. The shrinkage target is given by,

$$B_{ij} = \begin{cases} S_{ii}: & i = j \\ \sqrt{S_{ii}S_{jj}}: & i \neq j \end{cases}$$

The resulting linear estimator is denoted as P_{LW2}. The shrinkage intensity has the same form as for P_{LW1}, but with slighty different formula for $\hat{\varrho}$, which is given below.

$$\hat{\varrho} = \sum_{i=1}^{n} \hat{\pi}_{ii} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{1}{2} \left(\sqrt{\frac{S_{jj}}{S_{ii}}} \vartheta_{ii,ij} + \sqrt{\frac{S_{ii}}{S_{jj}}} \vartheta_{jj,ij} \right) \tag{45}$$

4) Emprical bayesian: In [35], an estimator for multivariate gaussian data is derived. It is given by the linear combination of the sample covariance matrix S and scaled identity matrix. The scaling factor is estimated from the data. We denote this estimator by P_{EB} and it is given by

$$P_{EB} = \frac{N_p d - 2N_p - 2}{N_p^2 d} [det(S)]^{1/d} I + \frac{N_p}{N_p + 1} S \tag{46}$$

5) Stein Haff: This estimator is described in [36]. The general form of the estimtor is $V(S)\Phi(l(S))V(S)^T$, where $V(S)$ matrix contains the eigenvectors of the sample covariance matrix S while $\Phi(l(S))$ is a matrix that is a function of the eigenvalues $l(S)$ of the S. The data is assumed to be normally distributed. and the sample covariance estimate S is therefore Wishart distributed $S \sim \mathcal{W}(P, d)$. The Stein-Haff estimator denoted by P_{SH}, is contructed by leaving the eigenvectors of the S unchanged while replacing the eigenvalues l by $\tilde{\tilde{l}}_i = nl_i/(n - p + 1 + 2l_i \sum_{j=1, j\neq i}^{N_p} (1/(l_i - l_j)))$. Eigenvalues can get disordered by the transformation and might become negative, which could lead to the covariance estimate lossing its positve definiteness. Therefore another algorithm called *isotonic regression* is used in conjunction with the transformation [37]. This lead to eigenvalues $\tilde{\mathbf{l}} = \{\tilde{l}_1, \tilde{l}_2 \ldots \tilde{l}_p\}^T$. Hence, the estimate P_{SH} is given by $V(S)Diag(\tilde{\mathbf{l}})V(S)^T$.

6) Minimax: The final shrinkage estimator considered is derived in [38]. Again Gaussian assumption is made. This estimator is termed minimax because under certain loss function, it has the lowest worst case error [32]. Its structure is similar to the P_{SH} but sample eigenvalues are replaced by $\tilde{l}_i = (n/(n + p - 1 - 2i)l_i)$. This estimator is denoted here by P_{MX}. Isotonizing regression is not applied in this case.

There is another interetsing covariance estimator by Ledoit and Wolf [39] in which non-linear transformation of the sample eigenvalues is considered. Also it requires solving a non-linear optimzation problem using sequential linear programming. It is shown that the new non-linear estimator outperforms the linear shrinkage estimators, described earlier in this section. In the current work we do not consider this method.

D. Re-generating the particles set

In the standard particle filter, new set of particles are generated after the measurement inclusion step. This is done in order to avoid the particle degeneration. A measure of the particle degeneracy is the effective number

of particles N_{eff}. When N_{eff} falls below a certain threshold, resampling of the particles is carried out. Depending on the number of particles, this can be computationally expensive. Homotopy based particle flow filters try to avoid the particle degeneracy by the gradual inclusion of the measurements. Unlike standard particle filters, resampling is not a mandatory step in the DHF [15], as it moves the particles to the correct region of the state-space. However due to the inexactness of the homotopy flow ODE, the particle state update itself is imperfect. Hence the generation of a new particle set could potentially help in relocating/confining the particles to the correct region. Instead of the conventional resampling, an optional redrawing of the particles is hinted out by the Daum and Huang in their papers. We find a single source describing the particles redrawing method. In [21], it is suggested to redraw a new set of posterior particles by sampling a Gaussian distribution. The mean of the distribution is estimated using particles, while the filtered covariance matrix is provided by the EKF. In the current work we consider two redrawing schemes, one using a single multivariate Gaussian distribution (MVG), and other using a Gaussian mixture model (GMM) that is estimated through the kernel density estimation (KDE).

1) MVG: Our first technique is inspired by the one described in the pseudo-code in [21]. The main difference is that we don't re-draw the whole set of particles. Instead, only those particles are redrawn which are deemed too *wayward*. Multivariate gaussian distribution is fitted to the posterior particles. This amounts to the estimation of the mean and variance of the MVG, given the particles. New particles are generated from this MVG.

2) KDE-GMM: Our next re-drawing scheme is based on the intuition that, a Gaussian fit to the posterior distribution might not be well suited for all cases. Hence we look for a non-Gaussian approximation to the filtered particles. The next most intuitive approach is to fit a Gaussian Mixture model (GMM) to the data. The key-factor in the GMM approximation is the number of components, which can be set to a fixed value or could be data driven. The textbook approach to estimate the GMM parameters is the expectation-maximization or EM method [40]. Alternatively, non-paramteric methods like Kernel Density Estimation (KDE) may be employed for the estimation of the probability density, which is given by the sum of estimation kernels with a certain smoothing factor, centered at data points. Smoothing factor is also called bandwidth. In this paper we use the online KDE approach described by Kristan et al. in [41], in which a new method for online KDE is described. The method enables the construction of a multivariate probability density estimate by observing only a single sample at a time. The KDE of the target distribution is estimated using the sample distribution which is constructed by online clustering of the data points. Each new observation is treated as a distribution in the form of Dirac delta functions. In the final form of the sample distribution, Dirac delta functions are smoothed out into Gaussians. Sample distribution is continuously refined and compressed in order to keep the algorithm complexity low.

3) Redrawing Algorithm: The purpose of re-drawing is to reduce the spread of the particles and relocate them in the appropriate region of the state space. We use the Mahalanobis distance (δ_M) for deciding the *waywardness* of particles. Given $\mathcal{N}(\mathbf{x} \mid \bar{\mathbf{x}}^p, P_p)$, the MVG approximation to the posterior, the distance (δ_M) for the posterior particle $\mathbf{x}_i^p$ is given by,

$$\delta_M^{MVG}(i) = (\mathbf{x}_i^p - \bar{\mathbf{x}}^p)^T P_p^{-1} (\mathbf{x}_i^p - \bar{\mathbf{x}}^p) \tag{47}$$

We define a similar measure for the GMM model with K components,

$$\delta_M^{GMM}(i) = \sum_{k=1}^{K} w_k ((\mathbf{x}_i^p - \mu_k)^T \Sigma_k^{-1} (\mathbf{x}_i^p - \mu_k)) \tag{48}$$

where w_k, μ_k and Σ_k are the weight, mean and covariance of the kth component of the GMM estimated through the online-KDE. Inverse of the distance $\varsigma = 1/\delta_M^*$ is a measure of the closeness of a particle to the estimated mean value. We use this value as a sort of weight ascribed to a particle, such that the particles close to the mean value are assigned a higher weight and vice versa. These weights are then normalized. Next, the particle *Assemblage*, denoted as Υ is calculated. Υ has the same form as the Effective Sample Size (ESS), in the traditional particle filter, and is a measure of the particle spread about the mean value. A higher value of Υ indicates an relatively even spread of the particles about the mean, whereas a lower value might suggest fragmentation of the particles into sub-clusters. A detailed analysis of this measure is presented in the Appendix A.

ALGORITHM 2: *Particle redrawing criterion*

$\varsigma(i) = \dfrac{1}{\delta_M^*(i)} \ \forall i;$

$\Upsilon = \dfrac{1}{\left(\sum_{i=1}^{N_p} \dfrac{\varsigma(i)}{\sum_{j=1}^{N_p} \varsigma(j)} \right)^2};$

if $\Upsilon \leq \nu_M \cdot N_p$ **then**

  if $\delta_M^*(i) \geq \sqrt{\dfrac{\Upsilon}{N_p}} \cdot \max \delta_M^* \ \forall i$ **then**

    Redraw from $\mathcal{N}(\mathbf{x} \mid \bar{\mathbf{x}}^p, P_p) / \sum_{j=1}^{K} w_k \mathcal{N}(\mathbf{x}_i \mid \mu_k, \Sigma_k)$;

  else

    NoRedraw;

  end

else

  NoRedraw;

end

$$x^i_{k+1} = x^i_k + \dot{x}^i_k \Delta t + \tfrac{1}{2} a_{x_{k+1}} \Delta t^2 \qquad \Pi^1_{x_k} = \frac{1}{N-1} \sum_{i=2}^{N} \left(\frac{\kappa_1}{\sqrt{(x^1_k - x^i_k)^2 + (y^1_k - y^i_k)^2} + \delta} \right) \frac{v_t^2}{r_t} \cos\left(\frac{v_t}{r_t} k\right)$$

$$y^i_{k+1} = y^i_k + \dot{y}^i_k \Delta t + \tfrac{1}{2} a_{y_{k+1}} \Delta t^2 \qquad \Pi^1_{y_k} = -\frac{1}{N-1} \sum_{i=2}^{N} \left(\frac{\kappa_1}{\sqrt{(x^1_k - x^i_k)^2 + (y^1_k - y^i_k)^2} + \delta} \right) \frac{v_t^2}{r_t} \sin\left(\frac{v_t}{r_t} k\right)$$

$$\dot{x}^i_{k+1} = \dot{x}^i_k + \Pi^i_{x_k} \Delta t + a_{x_{k+1}} \Delta t \qquad \Pi^i_{x_k} = \kappa_2 (x^1_k - x^i_k) - \kappa_3 \dot{x}^i_k$$

$$\dot{y}^i_{k+1} = \dot{y}^i_k + \Pi^i_{y_k} \Delta t + a_{y_{k+1}} \Delta t \qquad \Pi^i_{y_k} = \kappa_2 (y^1_k - y^i_k) - \kappa_3 \dot{y}^i_k \tag{D1}$$

$$r^i_{k+1} = \sqrt{(x^{(i)}_{k+1})^2 + (y^{(i)}_{k+1})^2} + v^i_{r_{k+1}} \qquad \theta^i_{k+1} = \tan^{-1}\left(\frac{y^{(i)}_{k+1}}{x^{(i)}_{k+1}}\right) + v^i_{\theta_{k+1}} \tag{D2}$$

Gaussian noise noise:

$$p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1}) = p(\mathbf{r}_{k+1} \mid \mathbf{x}_{k+1}) p(\boldsymbol{\theta}_{k+1} \mid \mathbf{x}_{k+1})$$
$$= \frac{1}{(2\pi\sigma_r\sigma_\theta)^N} \prod_{i=1}^{N} \exp\left\{ -\frac{1}{2\sigma_r^2} \left(r^{(i)}_{k+1} - \sqrt{(x^{(i)}_{k+1})^2 + (y^{(i)}_{k+1})^2} \right)^2 - \frac{1}{2\sigma_\theta^2} \left(\theta^{(i)}_{k+1} - \tan^{-1}\left(\frac{y^{(i)}_{k+1}}{x^{(i)}_{k+1}}\right) \right)^2 \right\} \tag{D3}$$

Non-Gaussian noise:

$$p(\mathbf{z}_{k+1} \mid \mathbf{x}_{k+1}) = p(\mathbf{r}_{k+1} \mid \mathbf{x}_{k+1}) p(\boldsymbol{\theta}_{k+1} \mid \mathbf{x}_{k+1})$$
$$= \frac{1}{(2\pi\beta^2)^{N/2} |R_r|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{r}_{k+1} - \tilde{\mathbf{r}}_{k+1})^T R_r^{-1} (\mathbf{r}_{k+1} - \tilde{\mathbf{r}}_{k+1}) \right\} \prod_{i=1}^{N} \exp\left\{ -\frac{1}{\beta} \left(\theta^{(i)}_{k+1} - \tan^{-1}\left(\frac{y^{(i)}_{k+1}}{x^{(i)}_{k+1}}\right) \right) \right\} \tag{D4}$$

$$\tilde{\mathbf{r}}_{k+1} = \left[\sqrt{(x^{(1)}_{k+1})^2 + (y^{(1)}_{k+1})^2} \sqrt{(x^{(2)}_{k+1})^2 + (y^{(2)}_{k+1})^2} \cdots \sqrt{(x^{(N)}_{k+1})^2 + (y^{(N)}_{k+1})^2} \right]^T \qquad R_r = \begin{bmatrix} \sigma_r^2 & \sigma_{r_x}^2 & \cdots & \sigma_{r_x}^2 \\ \sigma_{r_x}^2 & \sigma_r^2 & \cdots & \sigma_{r_x}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{r_x}^2 & \sigma_{r_x}^2 & \cdots & \sigma_r^2 \end{bmatrix}$$

Redrawing takes place only when the assemblage falls below a certain value, which in our case equals $\nu_M \cdot N_p$. We call ν_M as the *Redrawing Intensity* and its value can be set to any value between 0 and 1. When ν_M is 0, redrawing never takes place, while redrawing happens surely for the value 1. In our previous work [42], we re-drew the whole set of posterior particles, when the redrawing criterion was met. Here we make a small change and redraw only certain particles, which are deemed too off the mean value. For that purpose, we compare δ^*_M for each particle against a certain threshold which is dependant on the assemblage. If the criterion is met, the particle is redrawn from the MVG or GMM. The procedure is mentioned in the Algorithm 2.

V. MODEL DESCRIPTION

Here we consider a scenario similar to the one described in [27], namely the tracking of multiple targets in a 2D space using range and bearing measurements, in order to study the effects of the methods proposed in the previous sections. States of targets are interdependent, therefore resulting in a non-linear coupled dynamical model. Furthermore, target association is assumed to be perfectly known and hence we do not use any data association algorithm. The state vector for the target i at time instant k is $\mathbf{x}^{(i)}_k = (x^{(i)}_k, y^{(i)}_k, \dot{x}^{(i)}_k, \dot{y}^{(i)}_k)$, where $x^{(i)}_k$ and $y^{(i)}_k$ represent the position while $\dot{x}^{(i)}_k$ and $\dot{y}^{(i)}_k$ representing velocity components along the x and y-axis respectively. The overall state vector is formed by concatenating the individual target state vectors $\mathbf{x}_k = [\mathbf{x}^{(1)}_k, \mathbf{x}^{(2)}_k \ldots \mathbf{x}^{(N)}_k]$. Also the measurement vector for the target i is given by $\mathbf{z}^{(i)}_k = (r^{(i)}_k, \theta^{(i)}_k)$, where $r^{(i)}_k$ is the range to the ith target while $\theta^{(i)}_k$ is its bearing. The overall measurement vector at time k is generated in a similar way. The process model is described in equations (D1), where $a_{x_{k+1}}$ and $a_{y_{k+1}} \sim \mathcal{N}(0, \sigma_a^2)$, Δt is the time discretization step size

and N is the total number of targets. The intuition behind the model is to make the targets motion coupled to each other. The target ($i = 1$) is pursued by all other targets ($i > 1$). The changes in the speed and direction of the targets depend on their relative distances to each other. κ_1, κ_2 and κ_3 are the coupling constants in the model. $\Pi^1_{x_k}$ and $\Pi^1_{y_k}$ control the speed/direction change for the pursued target and is inversely proportional to the sum of its relative distances to the all others. As pursuers come close, the pursued target changes its speed and direction. The direction change is realized through terms $(v_t^2/r_t)\cos((v_t/r_t)k)$ and $(v_t^2/r_t)\sin((v_t/r_t)k)$. r_t and v_t are the turning radius and velocity respectively and δ is a small offset. Similarly, the speed and direction changes for the pursuers are controlled by the terms $\Pi^i_{x_k}$ and $\Pi^i_{y_k}$.

If κ_1, κ_2 and κ_3 are set to zero, then state dynamics corresponds to the standard discrete white noise acceleration (DWNA) model. The measurement model for the ith target is given by (D2). Measurements consist of ranges and angles of the two object types, target and the pursuer. We consider two measurement models, one with uncorrelated Gaussian noises for both range and the angle, while the other with correlated Gaussian range noise and Exponentially distributed angle noise. For the first model, the likelihood is given by the (D3). We assume that both range and and bearing measurement noise $v_{\theta_{k+1}}$ vectors are mutually independent at each time step. Also, both noises are uncorrelated within themselves such that $\mathbb{E}[v^i_{r_{k+1}} v^j_{r_{k+1}}] = 0$ and $\mathbb{E}[v^i_{\theta_{k+1}} v^j_{\theta_{k+1}}] = 0$ for $i \neq j$, In the second measurement model, range measurement noises $v_{r_{k+1}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R_r})$ are mutually correlated but are independent w.r.t. the bearing measurement noises $v_{\theta_{k+1}}$. Bearing measurement noise elements $v^i_{\theta_{k+1}}$ are exponentially distributed with the scale paramter β, such that $\mathbb{E}[(v^i_{\theta_{k+1}})^2] = \beta^2$ and $\mathbb{E}[v^i_{\theta_{k+1}} v^j_{\theta_{k+1}}] = 0$. $\mathbf{R_r}$ represent the covariance matrix of $v_{r_{k+1}}$ with $\sigma_r^2 = \mathbb{E}[(v^i_{r_{k+1}})^2]$ and $\sigma^2_{r_x} = \mathbb{E}[v^i_{r_{k+1}} v^j_{r_{k+1}}]$. $\sigma^2_{r_x}$ is assumed to be same for any two targets. Measurement noises are chosen as such in order to create a challenging estimation scenario, in which the relative strength of the particle flow method can be tested against the more traditional solutions like the EKF and the Particle filter. The likelihood function for this measurement model is given in (D4).

A. Parameters setting

We simulate two targets (N=2) in our analysis. Δt is set to 1, σ_a^2 to 0.5 ms^{-2}, σ_r^2 is set to 2000 m^2, $\sigma^2_{r_x}$ to $(3/10)\sigma_r^2$, while β^2 is set to $(1/10)\text{rad}^2$. In this paper, we work only with the strongly coupled model with coupling constants κ_1, κ_2 and κ_3 set to 8000, 0.01 and 0.1 respectively. The turn radius r_t and turn speed v_t are set to 200 m and 10 ms^{-1} while δ is set to 0.001. We use 100 DHF particles ($N_p = 100$). DHF and SIR-PF particles are initialized by sampling Gaussian

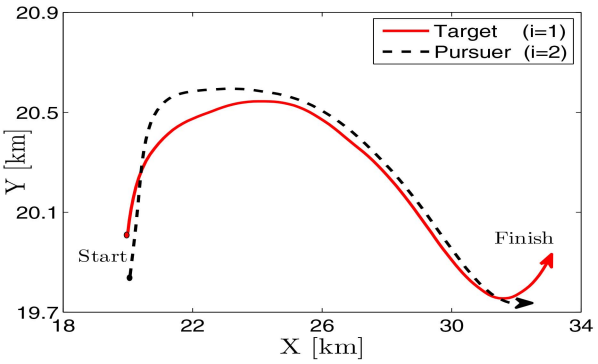

Fig. 1. Sample trajectory

distribution with mean of 20000 m and variance of 5000 m^2 for position elements, while their velocities are sampled from Gaussian distribution with mean and variance of 5 ms^{-1} and 25 m^2s^{-2} respectively. EKF is initialized by sampling the Gaussian with initial state vector as mean and with variances 10^4 and 1 for the position and the velocity respectively. We note that $\sigma_r < D_{i,k}\sigma_\theta \ \forall i,k$, where $D_{i,k}$ represents the distance of ith target from the radar location at time instant k. In figure 1, we show a sample trajectory generated by using these parameters. We note that the target object (i=1) is pursued by the pursueing object (i=2). The target turns and increases speed as it is approached by the pursuer. The trajectory has segments of straight run as well as turns in the middle and at the end. Turning, in particular is challenging for the estimation algorithm, as this in addition to the non-linearity in the measurements, introduces non-linearity in the process model as well.

VI. SIMULATION RESULTS

We use root average mean square error (RAMSE) as the performance metric. It is defined as following. Let M be the total number of simulation runs for a particular scenario, $x_k^{i,m}$ and $y_k^{i,m}$ denote the positions of the ith target along X and Y-axis respectively, at time instant k in the mth trial. Likewise, let $\hat{x}_k^{i,m}$ and $\hat{y}_k^{i,m}$ denote estimated positions for the ith target. The RAMSE Υ_r is then defined as,

$$\epsilon_r = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left[\frac{1}{N}\sum_{i=1}^{N}((x_k^{i,m} - \hat{x}_k^{i,m})^2 + (y_k^{i,m} - \hat{y}_k^{i,m})^2)\right]}$$

We simulate each scenario for a total of fifty times ($M = 50$). First, we describe the effect of the numerical integration schemes.

A. Effect of numerical integration schemes

We compare the performance of the four methods mentioned in subsection IV-B, namely Euler's method, Runge-Kutta scheme of fourth order, Rosenbrock formula of second order and Gear's method. While we wrote scripts for the first two methods, MATLAB provided functions *ode23s* and *ode15s* were used for the

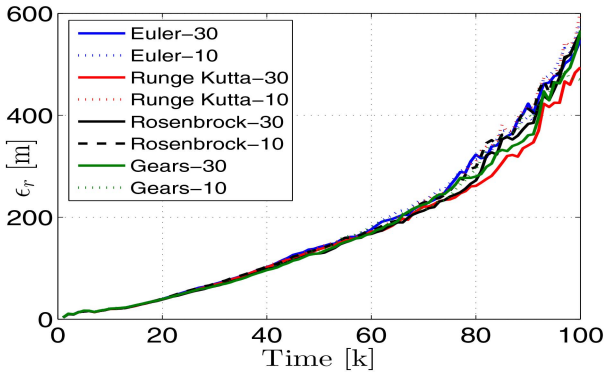

Fig. 2. Comparison of numerical integration schemes for the Gaussian noise

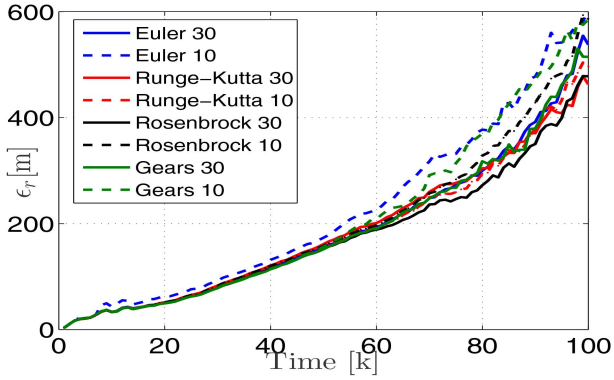

Fig. 3. Comparison of numerical integration schemes for the non-Gaussian noise

Rosenbrock and the Gear's methods respectively. We also compare the effect of grid discretization on the performance of the above schemes. We use two specific cases, 10 uniformly spaced pseudo-time points (coarse discretization) and 30 exponentially spaced points (fine discretization). Also, the prior covariance matrix of a parallel running EKF is used to compute the flow. We plot the RAMSE ϵ_r for different schemes in figures 2a and 2b. We see a general increasing trend in the RAMSE vs. time for all methods. This is due to the specific process model used, which results in the peculiar targets trajectories involving rapid accelerations and sharp turns. It can be observed that the difference in the performance of different integration schemes is more pronounced in the case with non-Gaussian noise, as evident by the wider spread in the error curves. For the Gaussian case, we note that Runge-Kutta method with 30 λ points has the lowest error. Among the integration methods with 30 discretization points, Euler's method has the highest error. We can also note that Gears-10 has the lowest error for all methods employing 10 discretization points. The largest error is exhibited by the Euler-10 method, which happens to be the fastest. On the other hand, Rosenbrock-30 is the slowest of all the methods. Euler-30 ranks second in the processing speed, as it is almost 1.5 times faster than its nearest competitor Runge-Kutta-10, while being 3 times as fast as Gears-10 though slightly inferior in the performance.

Next we discuss the results for the model with non-Gaussian measurement noise. As discussed earlier, the error curves show more spread. We note that the Rosenbrock method with 30 λ points has the lowest RAMSE, while the Euler's scheme with 10 λ points is the worst performer followed closely by the Gears-10.

Runge-kutta methods with both 10 and 30 points are the second best. In fact, the difference in the performance between the two is very small. This is followed by the Gear-30 and the Euler-30 methods. We tabulate the time averaged RAMSE and the average processing time per particle for all methods in the Table I. Note that the time values mentioned only represent the time spent while solving the homotopy ODE for a single particle. The largest and the smallest values are highlighted in red and green respectively. It can be seen that while the Runge-Kutta-30/Rosenbrock-30 are the best methods, they are also computationally very expensive. On the other hand, the Euler-10 is the fastest but the worst performer of all methods. Euler-30 represents a right trade-off between the performance and the processing time. In the proceeding analysis, we use Euler-30 as the default integration scheme.

TABLE I
Comparison for differ integration schemes

Gaussian		
Method	Avg. ϵ_r [m]	Proc.time (pp) [ms]
Euler-30	178.45	6.6
Euler-10	181.70	2.3
Runge-Kutta-30	163.06	27.4
Runge-Kutta-10	180.60	9.1
Rosenbrock-30	169.66	80.5
Rosenbrock-10	178.04	62.8
Gears-30	169.42	27.4
Gears-10	172.37	19.3
Non-Gaussian		
Method	Avg. ϵ_r [m]	Proc.time (pp) [ms]
Euler-30	186.69	5.6
Euler-10	223.07	1.8
Runge-Kutta-30	181.68	38.5
Runge-Kutta-10	184.39	12.7
Rosenbrock-30	173.17	71.9
Rosenbrock-10	196.30	55.8
Gears-30	184.49	26.4
Gears-10	186.69	17.9

B. Effect of shrinkage covariance estimation

Next we analyze the effect of shrinkage estimation schemes. We compare the performance of the six methods mentioned in subsection IV-C, together with that of sample covariance and the prior covariance matrices S and P_{EKF} respectively. We describe the DHF estimate generated using a particular covariance estimation scheme X as DHF-X. We use four metrics to judge the effectiveness of these methods. First and the foremost is the RAMSE of the DHF estimates. This is the central

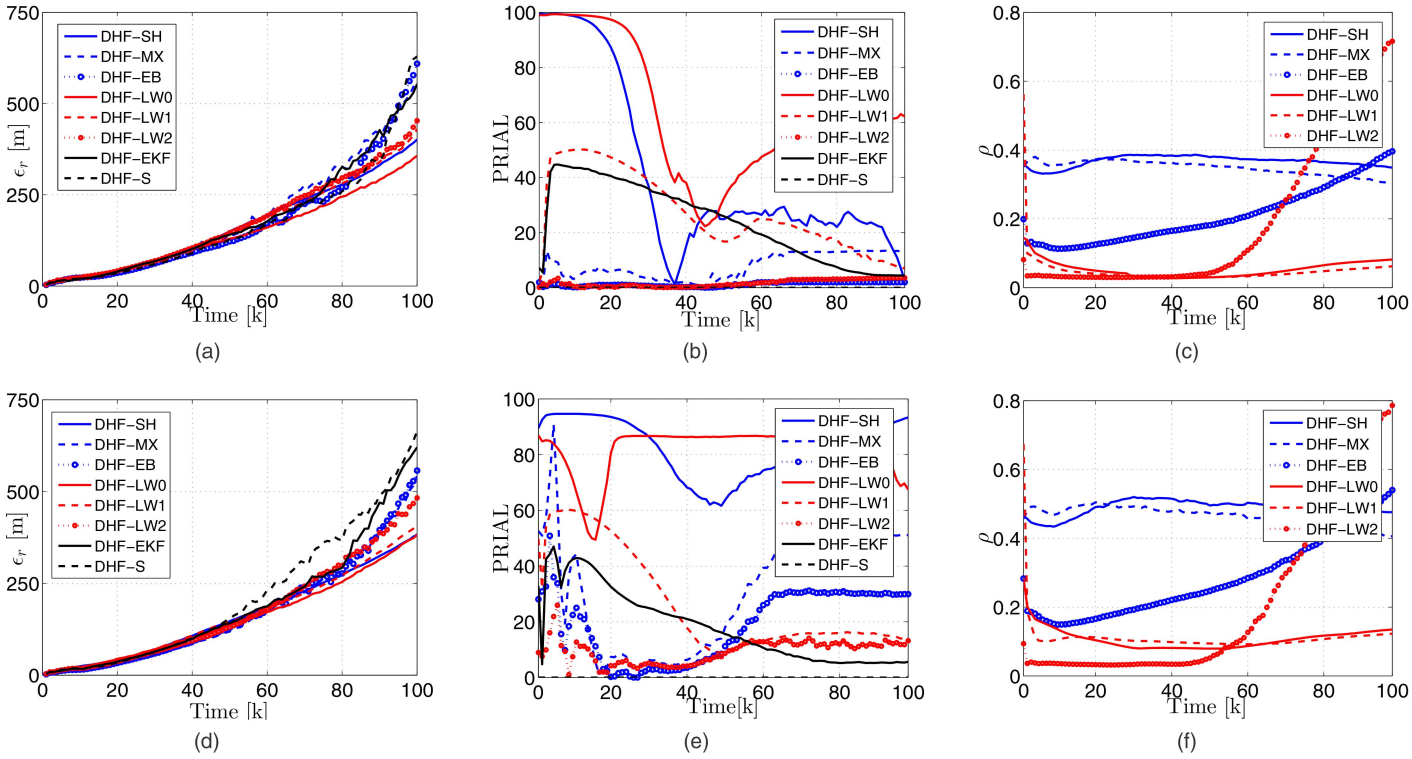

Fig. 4. (a, d) Position RAMSE (ϵ_r), (b, e) PRIAL and (c, f) Shrikage intensity (ρ) vs. time, for different covariance estimation schemes. Subfigures (a, b, c) show results for the case with Gaussian noise, while subfigures (d, e, f) show respective results for the case with non-Gaussian noise.

criterion for judging the effectiveness of the shrinkage schemes, in terms of the accuracy of the DHF estimates. Second is the relative accuracy of the covariance matrix estimates themselves. In the context of the shrinkage estimation, we use the percentage relative improvement in average loss or PRIAL as the measure for the exactness of any shrinkage covariance estimate, as defined in [32],

$$\text{PRIAL} = \left(1 - \frac{\mathbb{E}[\|P_{(.)} - P\|^2]}{\mathbb{E}[\|S - P\|^2]}\right) \times 100 \qquad (49)$$

where $\|(.)\|$ represents the Frobenius norm, S is the sample covariance matrix estimate, while $P_{(.)}$ and P are the shrinked covariance and the true covariance estimates, respectively. As P is not known, in the current scenario this is approximated by the covariance estimate from a sampling importance resampling particle filter (SIR-PF) with 25000 particles. Third, is the shrinkage intensity ρ, which indicates the compromise between the unbiased but more variant sample based estimate and the biased but less variant target. A lower value of ρ represents the closeness of covariance estimate to the Sample covariance matrix S. On the other hand, a higher value highlights a stronger influence of the target matrix B. At last, we use the condition number k_{cond} to analyze the spread in the eigenvalues of covariance estimates over the time. Plots for RAMSE, PRIAL and ρ are shown in figures 4 (a,d), 4 (b,e) and 4 (c,f) respectively, while time averaged k_{cond} is shown only in the tabulated form in the Table II.

First we discuss the RAMSE for DHF with covariance estimates from all methods, for the Gaussian noise model. DHF-MX (Minimax) has the highest error. This can be explained as following. The minimax estimator scales the eigenvalues of the sample covariance matrix in a non-linear fashion. The highest $\lfloor((p-1)/2)\rfloor$ eigenvectors have their eigenvalues shrinked, while for the others the eigenvalues are expanded. Scaling is done just based on the order of the sorted eigenvalues and it does not take into account any other possible information in the structure of the matrix S. This simplicity renders the estimator performing worse as compared to the others. Next in the line is the DHF-EKF. As can be seen in the figure 4d, the error increases sharply after about 80 s.

Although each simulated trajectory is not exactly the same, this is roughly the time when the targets start turning in our coupled motion model in most of those runs. Hence this is a critical point, as this tend to increase the non-linearlity in our motion model. We see that for the DHF based on the EKF prior covariance, error starts rising indicating a failure in proper tracking. This outcome is inline with our previous results [27], where we reported that the standard DHF (EKF based DHF) fails for a coupled motion model. This also proves to be a very strong motivation for the search of an alternative covariance estimation method, which could be better than P_{EKF}. Interestingly, the performance of sample covariance based DHF is better than many other

TABLE II
Comparison for different covariance estimation schemes

Gaussian				
Method	Ave. ϵ_r [m]	Ave. PRIAL	Ave. ρ	Ave. k_{cond}
Stein-Haff	164.29	41.34	0.36	38620
Minimax	188.03	7.86	0.34	272820
Emp.Bayesian	170.94	1.20	0.20	181380
Ledoit-Wolf-0	144.77	63.13	0.05	170
Ledoit-Wolf-1	163.05	27.26	0.05	55610
Ledoit-Wolf-2	171.10	1.63	0.19	60370
EKF covariance	179.23	23.79	0	71460
Sample covariance	168.09	0	0	139760
Non-Gaussian				
Method	Ave. ϵ_r [m]	Ave. PRIAL	Ave. ρ	Ave. k_{cond}
Stein-Haff	161.22	83.30	0.40	45080
Minimax	166.58	32.0711	0.38	55490
Emp.Bayesian	171.28	18.78	0.23	46730
Ledoit-Wolf-0	153.32	81.71	0.09	180
Ledoit-Wolf-1	161.92	27.01	0.09	53220
Ledoit-Wolf-2	171.27	9.38	0.21	48470
EKF covariance	189.80	15.15	0	67770
Sample covariance	213.41	0	0	142260

schemes. In fact for most of the simulation time it has an error comparable to the better performing DHFs. It starts to increase only when targets start turning. After that time, the DHF-S fails to properly cope with the induced process non-linearity and the filter diverges rapidly. All variants of Ledoit-Wolf covariance estimators perform better, with LW0 based DHF outperforming all other filters. This can be attributed to the optimal convex combination (asymptotically) of the sample covariance matrix S and the scaled identity matrix I. This structure of the estimator results in a well-conditioned covariance estimator, that is more stable (from inversion point of view). This property can be critical when considering the turning motion of the targets, as DHF particles can be flung far and wide if the flow is incorrect which of course depends on inverting the prior covariance matrix. DHF with the other two covariance estimators from Ledoit and Wolf perform a little inferior relative to the DHF-LW0. P_{LW1} and P_{LW2} were derived for special problems in portfolio estimation and have very special structures. This lessens their generality and makes them very application specific.

Next we discuss the non-Gaussian case. We note that DHF-S is the worst method. DHF-EKF comes next as its error is also shows steeply diverging trend. This can be explained as follows: given that the measurements are non-linear functions of state variables, and bearing noise is exponentially distributed, the EKF is not a good approximation for the resulting non-linear and non-Gaussian scenario. Hence the covariance estimates generated by the EKF will not be accurate. DHF-LW0 has the lowest average error amongst all methods. This is because P_{LW0} is a distribution free estimator, and hence produces good estimates even in this non-Gaussian scenario. It is followed by the Stein-Haff and Minimax estimators. Compared to the DHF-EKF, all estimators except the sample covariance DHF-S have lower average RAMSE.

Next we discuss the PRIAL for the covariance estimates. The expectation in the formula (49) is calculated by averaging over all simulation runs. A value of 100 means perfect estimation accuracy, while 0 means accuracy as good as the sample covariance matrix S. Again we discuss the Gaussian case first. We note that the PRIAL for P_{LW0} is highest while it is lowest for P_{LW2} Again, this can be attributed to the very specific structure of this estimator. For the non-Gaussian case, we note that the PRIAL for P_{SH} is the highest on the average, while is lowest for the P_{LW2}.

One noteworthy thing is to compare the PRIAL of the estimators in the Gaussian vs. non-Gaussian case. We see that the PRIAL, on average, is lower for the Gaussian case. This can be explained by the fact that PRIAL represents how better an estimator when compared to the sample covariance estimator S. In non-Gaussian case, DHF-S is worse performer, which points to the fact that S is not a well-suited estimator. In fact, all DHFs are better than DHF-S. Hence we see that the PRIAL for the estimators in the non-Gaussian case is significantly higher. On the other hand in the case of Gaussian noise, S is not the worst estimator. This tends to increase the ratio $\mathbb{E}[\|P_{(.)} - P\|^2]/\mathbb{E}[\|S - P\|^2]$, which results in the lower values of PRIAL.

Shrinkage intensities ρ are shown in the figures 4c and 4f. We note that the lowest shrinkage intensity in both cases is exhibited by P_{LW0}. This suggest a higher contribution of the sample covariance than the scaled identity matrix in the optimal combination. P_{SH} has the highest shrinkage intesity on average and is also the most consistent. Shrinkage intensities in the non-Gaussian case are higher, again suggesting the inadequacy of the sample covariance matrix in the non-linear/non-Gaussian scenario. Finally we discuss the average logarithamic condition number $\log k_{cond}$. As expected, P_{LW0} has the lowest condition number over time, at least two orders of magnitude smaller than all other estimators. Also, the S has the highest condition number.

For the subsequent analysis, we consider P_{LW0} as the default covariance estimation scheme.

C. Effect of Redrawing

Once decided upon the pseudo-time discretization, flow integration and prior covariance estiamtion schemes, we now study the effect of redrawing on the performance of the DHF. As mentioned in section III-D, we consider two methods for regenerating the particles. The first method is redrawing from a Multivatiate Gaussian (MVG), and the other from a Gaussian mixture model fitted to the posterior particles, that is estimated

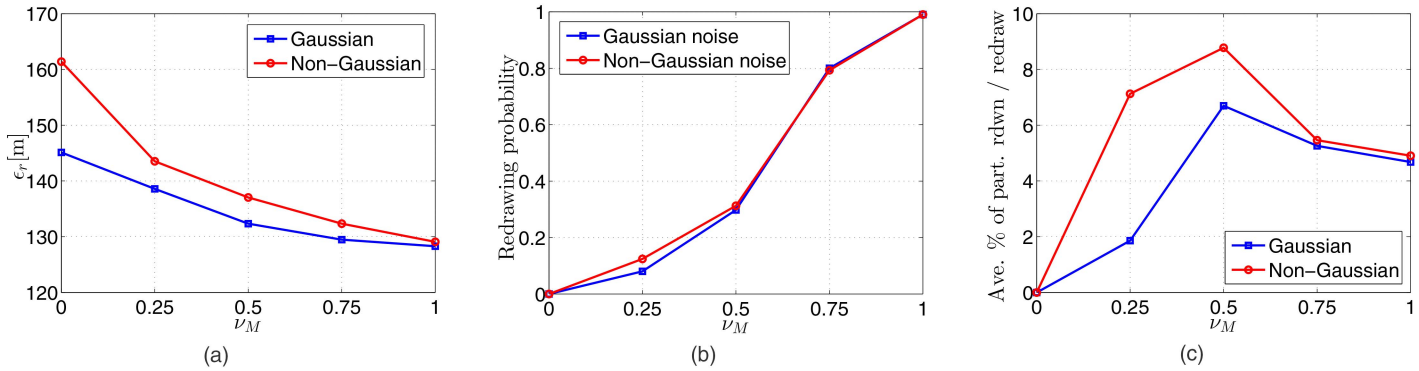

Fig. 5. (a) Time averaged RAMSE, (b) Redrawing probability and (c) Average percentage of particles redrawn vs. ν_M for Gaussian and non-Gaussian models.

using an online Kernel density estimation method. First we discuss the redrawing from MVG.

1) Multivariate Gaussian: We follow the Algorithm II mentioned in the section III-D for redrawing. The main parameter in that algorithm is the redrawing intensity ν_M. We vary ν_M between 0 and 1 and use five distinct values. First we study the effect of ν_M on the estimation accuracy, for which we plot the time averaged RAMSE for both Gaussian and the non-Gaussain cases in the Figure 5a. We see that as the redrawing threshold is increased the error decreases monotonically: the lowest error is for $\nu_M = 1$. We note that the improvement in the performance by increasing ν_M is stronger in non-Gaussian case than in the Gaussian one. This suggests the presence of more wayward particles in the non-Gaussian case, which are subsequently moved to the right regions after getting redrawn.

Next in the Figure 5b, we plot the redrawing probability vs. re-drawing intensity. Redrawing probability is defined as the number of times redrawing event takes place in the simulation divided by the total simulation time. So if particles are redrawn for half of the whole simulation duration, the redrawing probability is 0.5. The value is averaged over all the simuation runs. A higher value indicates a higher chance for particles to be redrawn during the simulation. We note a monotonically increasing relation between ν_M and the redrawing probability, which assumes a value of 1 for ν_M equals 1. This plot can also be used to infer about the assemblage, Υ. The assemblage is always greater than zero, making for the fact that no-redrawing happens for ν_M equals zero. As the ν_M is increased, the probability of finding Υ below the threshold $\nu_M \cdot N_p$ increases. e.g. from the figure 5b, it can be infered that almost 30% of the time Υ value is below $0.5N_p$. This suggest that probability of having fragmentation of particles about the mean into two sub-groups of equal sizes (or any other equivalent scenario resulting in Υ=0.5) is non-negligable. Also almost 50% of the time the value of the assemblage is between 50 and 75, while it is between 75 and 100 for almost 20% of the times. In relation to the RAMSE, we can conclude that the redrawing frequency has a direct positive effect on the estimation error. A higher redrawing probability leads to the reduced estimation error. We note that both the Gaussian and non-Gaussian cases have a similar trend.

But how many particles, on average, are redrawn at a given time instance. While several metrics can be used for this effect, we use in particular the average percentage of particles redrawn, further averaged over the simulation time as plotted in the Figure 5c. We see an interesting trend. The percentage of particles redrawn increases with the increase in the intensity ν_M up to 0.5, at which it hits the maximum 7%–9% of the particles for both cases. Then this value decreases. This can be explained in the light of the redrawing probability. For ν_M between 0 and 0.5, the redrawing probability increases and so does the percentage of redrawn particles. This suggests that even though assemblage can be expected to be below $0.5N_p$ about 30% of the time, at times there is a significant number of particles satisfying the redrawing condition $\delta^*_M(i) \geq \sqrt{\Upsilon/N_p} \cdot \max \delta^*_M$. That is why the redrawing criteria $\Upsilon \leq \nu_M \cdot N_p$ is met in the first place, given the low value for ν_M. As ν_M is increased beyond 0.5, the redrawing probability increases, but the average number of particles satisfying the redrawing conditions decrease. That also points to the increase in the assemblage. We note that, on average, more particles are redrawn in the case of non-Gaussian noise than in the case of Gaussian case. This result is expected as estimation under the non-Gaussian noise is more challenging.

When seen together with the estimation error, we note that although the average rate of particles redrawn at any given time is not more than 10%, but redrawing those particles amounts to a significant reduction in the error. Also the particles redrawn for ν_M equals 1 have the maximum effect on the estimation error as they are the few quite seperated from the rest of the particle cluster(s). If redrawn, they are moved to the correct region of the state-space, and hence contributing effectively to the point estimates.

2) Kernel Density Estimation: Now we discuss the effect of redrawing particles from a GMM, estimated

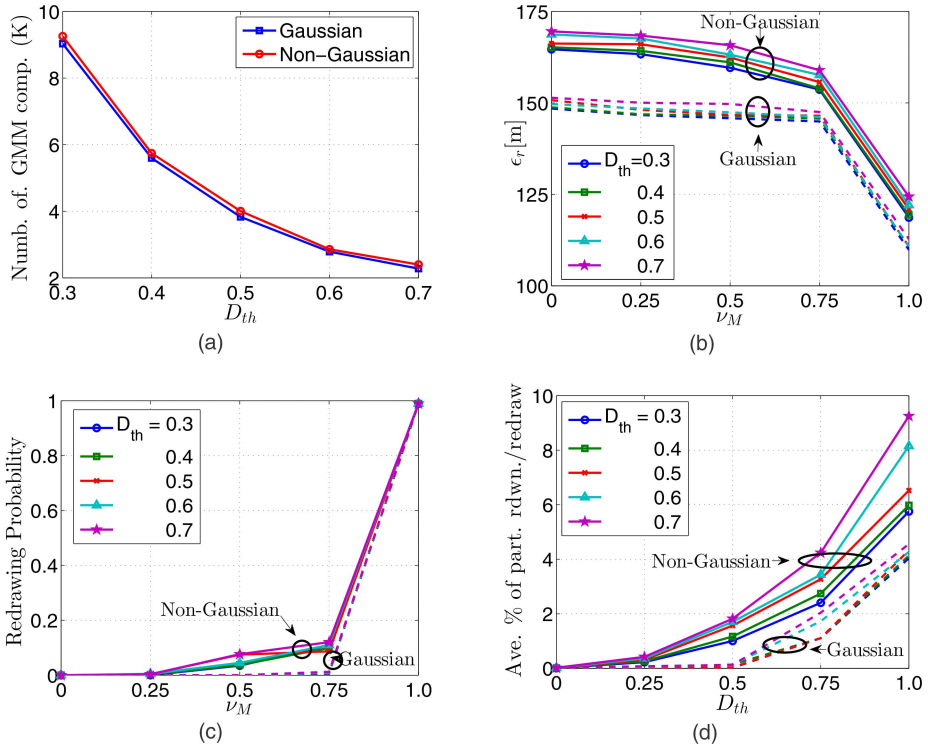

Fig. 6. (a) Average number of GMM components, (b) RAMSE vs. ν_M for different values of D_{th}, (c) Redrawing probability vs. ν_M and (d) Average number of particles redrawn per redraw vs. ν_M.

through the online KDE (oKDE) as described in [41], using the algorithm mentioned in the section III-D. We have used the source code for the oKDE provided by the authors at [43]. Although the method is general and can be used with any estimation kernel, the authors have used a multivariate Gaussian kernel in their work. oKDE method fits a GMM to the online data, which is supposed to arrive sequentially. In our context, we use the oKDE method to approximate the density of the particles after they move through the pseudo-time loop. Hence those particles can be thought of as coming from an importance sampler, and the task is to estimate the corrected posterior distribution. As a result we get an ensemble of weights, mean and covariances, $\{w_k, \mu_k, \Sigma_k\}_{k=1}^{K}$. Next, the averaged distance of each particle given the estimated GMM is calculated, and those particles which are thought to be too wayward are redrawn. As in the MVG case, we vary the redrawing threshold ν_M between 0 and 1.

There are two parameters that control the degree of estimation accuracy: the error threshold D_{th}, which controls the number of Gaussian components fitted to the data, and N_{init} which defines the number of data samples used for the initialization. Through experiments, we have found out that N_{init}, after a certain value, does not strongly influence the estimation accuracy. Therefore in our study we have kept N_{init} fixed to 33 (one third of total number of particles), while the threshold D_{th} is varied between 0.3 and 0.7, in the steps of 0.1. In the figure 6a we plot the average number of GMM components (K) vs. the error threshold D_{th}. We note that as the D_{th} is increased, K decreases exponentially. This can be attributed to the particular implementation method used by the authors in [43].

Next in figure 6b, we show the results for position RAMSE vs ν_M for various values of threshold D_{th}, for both Gaussian and non-Gaussian cases. There are a number of noteworthy things. First, we note that the error for the Gaussian cases is less than that for the non-Gaussian, for all values of ν_M. We saw a similar behaviour in the previous section, where the redrawing was done using a MVG. Secondly, we see that the error only slightly decreases with increasing ν_M up to 0.75. After that we observe a significant reduction in the error for both cases. This is explained in the following way: in contrast to redrawing from a MVG where the particles far from the estimated mean value had lower weight defined by the ς, here such particles can be softly assigned to more than one Gaussian components. And due to the relative weights of the GMM components, the contribution of those particles is lessened. This results in a higher assemblage Υ value, and hence the redrawing criterion is rarely met. But when ν_M is sufficiently high, such that Υ is below $\nu_M.N_p$, redrawing takes place. Particles which meet the redrawing condition are redrawn using the GMM. Statistically, particles are more likely to be redrawn from the components with the higher weights, and hence making those components even stronger while the opposite happens to the original low weight components. As a result, one can expect a

significant reduction in the particle spread after redrawing done in this manner. Lastly, we observe that the error for a lower value of D_{th}(hence higher K) is lower for both cases, for all values of ν_M. Again this is intuitive, as a higher number of GMM components is suggestive of the better accuracy of the fitted distribution to the posterior particles.

Figure 6c shows the redrawing probability vs. ν_M. We use the same definition for this probability as used in the previous section. We note that the redrawing probability for both noise cases is almost zero for ν_M less than or equal to 0.25. Between ν_M 0.25 and 0.75, we see a slight increase for the non-Gaussian case while it is still very close to zero for the Gaussian case. E.g. at $\nu_M = 0.75$, the redrawing probability is 10% for the case with non-Gaussian noise. A shape rise can be seen for both cases between 0.75 and 1. Also the redrawing probability is higher for the lower values of D_{th}. This trend has been explaned in the previous paragraph, where it was mentioned that for the higher assemblage values, the probability of redrawing is quite low. Hence the redrawing probability also reveals information about the distribution of the assemblage. In contrast to the MVG case, the assemblage values are significantly larger (less spread). Therefore redrawing is only expected to happpen for larger values of ν_M. Also a higher D_{th} (less K) tends to make the assemblage lower and hence the increasing the redrawing probability.

The average percentage of particles drawn per redraw is shown in the figure 6d. We observe a monotonically increasing trend for both Gaussian and non-Gaussian noises. We note that while the assemblage Υ value effect the redrawing probability, it is the distribution of the Mahalanobis distance itself that influences the average percentage of particles drawn per redraw. From the results we can infer that Mahalanobis distance distributions for both Gaussian and non-Gaussian noises are similar, although for the latter it is more skewed towards the right, as evident from the higher percentage of redrawn particles. The average percentage of particles drawn per redraw rises sharply for ν_M between 0.75 and 1, hence more particles are redrawn for these values. This can be correlated with the large drop in the estimation error. Altogether, it can be infered that the redrawing done for ν_M between 0.75 and 1 significantly increases the estimation accuracy. It can also be concluded that the GMM provides more accurate description for the posterior distribution. A higher percentage of particles is expected to be redrawn for higher values of D_{th} as the estimated GMM has fewer components, hence is not accurate enough.

D. Comparison against other filters

In this subsection, we compare the performance of our modified DHF against the other versions of DHF mentioned in the section II-C, together with the

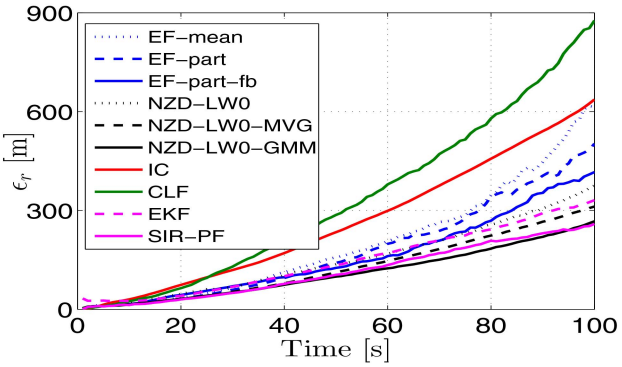

Fig. 7. Comparison for Gaussian noise

EKF and sampling importance resampling particle filter (SIR-PF) with 25000 particles. In total, we present the results for eight different variants for DHF, out of which three are different flavors of exact flow filter (EF), three are variants of non-zero diffusion constrained flow based filters (NZDCF), while the other two are based on the incompressible flow (IC) and the Coulomb's law flow DHF (CLF) respectively. The most basic version of the exact flow based DHF is reported in [16], where the flow equation is solved by linearizing the measurement model about the estimated prior mean value. We call this implementation as EF-mean. The second implementation of the exact flow has been reported by Ding and Coates in [21], and a pseudo-code is also provided. Two distinct changes are made to the EF-mean. In the first modification, the linearization of the measurement equation is carried out for individual particles, as opposed to being done only at the prior mean location. The second modification is related to the feedback of the DHF state estimates to the EKF, making the two filters coupled. In this study we consider these two cases individually i.e. the first modification alone and it together with the feedback. We call these implementations as EF-part and EF-part-fb respectively.

For the incompressible flow filter (IC), the flow equation (14) is solved for individual particles by assuming a Gaussian prior. Finally for the Coulomb's law based DHF (CLF), we use the parameters settings mentioned by the authors in [44]. One third of nearest neighbors are used in the evaluation of the equation (30). We have found that this filter is very sensitive to the parameters settings, and in general is very hard to tune. First we plot the RAMSE for the different filters for the Gaussian case in figure 7. We note that the CLF is the worst performer. The issue with this filter is the estimation of the probability density $p(\mathbf{x}, \lambda)$ for all particles throughout the pseudo-time, which is used in evaluating the flow equation $f(\mathbf{x}, \lambda) = \nabla \mathbf{V}(\mathbf{x}, \lambda)/p(\mathbf{x}, \lambda)$. As this is done using the few available particles, the resulting density estimate is not accuarte enough and the filter is prone to divergence. This is also the issue with the Monte-Carlo approximation of the integral for gradient

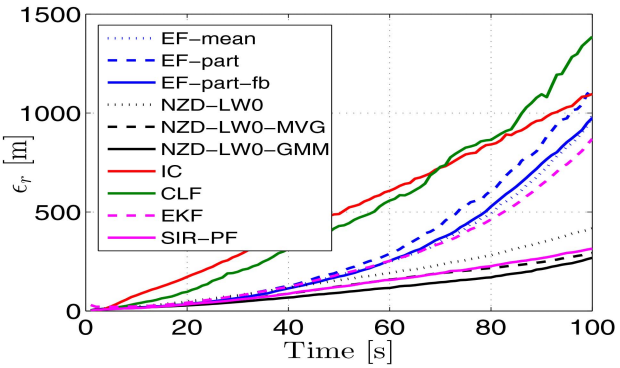

Fig. 8. Comparison for non-Gaussian noise

$\nabla \mathbf{V}(\mathbf{x}, \lambda)$. Next we see that DHF based on IC although being better than with CLF, still fares worse compared to all other filters. The filter is based on the assumption of zero-divergence, which appears to be a quite strict condition. Also, the flow might encounter singularities which can make the filter diverge. Amoung the three variants of the exact flow, the EF-part-fb is the best. This is expected as for this filter linearization is done about each particle, and also the filter is coupled to a parallel running EKF. The best amoung all DHF variants are the ones based on NZDCF, all of which use Euler integration with 30 time steps and LW0 covariance estiamtion scheme. We denote the DHF-NZDCF without redrawing by NZD-LW0, with redrawing from MVG by NZD-LW0-MVG, and the one with redrawing from GMM as NZD-LW0-GMM. For the redrawing we set the threshold ν_M equal to 1. Also for the oKDE, we set the D_{th} equal to 0.5, which on average fits 4 GMM components to the posterior distribution. We note that the NZD-LW0-GMM is the best of all the schemes, even surpassing the SIR-PF with 25000. NZD-LW0-MVG is a little worse in performance to the SIR-PF, but is still better than the EKF. Next we discuss the results for the non-Gaussian measurement noise, as plotted in figure 8.

We note that all filters, except the variants of DHF-NZDCF and SIR-PF, perform poorly. DHF-IC and DHF-CLF fail to track the targets, with the latter being the worst in the performance. All variants of DHF-EF show a diverging error trend. This is due to the fact that EF hinges on the Gaussian assumption, which is not valid in the current case. As a part of the measurement noise is non-Gaussian, we see that these filters are unable to properly track targets. The same reasoning can be applied to EKF. NZD-LW0-GMM, NZD-LW0-MVG and SIR-PF are the first, second and the third best performer respectively. The error for all filters is generally larger when compared to the case with the Gaussian noise.

Next, we compare the execution time τ for a single update, including both the time and the measurement update steps. Matlab simulations were performed on the computer with Intel Core2 Quad with 2.66 GHz processors and 4 GB RAM. Table III shows the processing time per time step in seconds. We note that the EKF is the fastest of all methods. Next in the line are the EF based DHF, with DHF-EF-mean being the fastest. DHF with IC flow and NZD flow have quite similar processing time. We can also note that the covariance estimation (LW0) and redrawing from MVG do not incur any significant processing overhead. oKDE, on the other hand takes quite a while to compute the GMM components. The processing time is the highest for D_{th}=0.3 and it drops exponentially with increasing the threshold. Redrawing with threshold 0.5 takes almost 1.2 seconds per time step, which is 6 times the processing time of the DHF-NZD-LW0. Hence the redrawing with KDE takes significant amount of time. The particle filter with 25000 particles takes 4.5 seconds, which makes it almost 4 times slower than the DHF-NZD-LW0-GMM. Finally, the slowest method is the DHF-CLF taking almost 8.5 second per time step. We note that the processing time for the model with non-Gaussian noise is slightly higher in general for most of the schemes.

TABLE III
Comparison of processing time for different filters

Method	Processing time τ [s] (Gaussian)	Processing time τ [s] (Non-Gaussian)
EKF	0.0004	0.0004
EF-mean	0.004	0.005
EF-part	0.10	0.10
EF-part-fb	0.105	0.105
IC	0.19	0.20
CLF	8.34	8.57
NZD	0.195	0.20
NZD-LW0	0.202	0.205
NZD-LW0-MVG	0.205	0.21
NZD-LW0-GMM (D_{th}=0.3)	1.77	1.83
NZD-LW0-GMM (D_{th}=0.4)	1.33	1.36
NZD-LW0-GMM (D_{th}=0.5)	1.19	1.21
NZD-LW0-GMM (D_{th}=0.6)	1.12	1.13
NZD-LW0-GMM (D_{th}=0.7)	1.09	1.11
SIR-PF (N_p=25000)	4.34	4.65

VII. DISCUSSION

Euler based numerical integration scheme is quite simple, but together with a clever pseudo-time discretization, can perform quite well. It is the most time efficient scheme. We analyzed different shrinkage estimation schemes. Some of them are tailor made for specific scenarios. The most general one is shrinkage towards identity matrix where no prior structure of the target matrix is assumed. It is a distribution free scheme and is shown to have outperformed other shrinkage estimators used in our analysis. Finally, we studied the effect of redrawing on the quality of the filter estimates. We choose two redrawing schemes: a

single MVG based re-drawing, and redrawing from a GMM estimated via the oKDE. The estimated density is then used to redraw particles which are considered too off the main cluster. The re-drawing algorithm uses the Mahalanobis distance of particles to calculate the assemblage Υ. When Υ falls below a certain threshold determined by the redrawing intensity ν_M, particles deemed too wayward are redrawn. We show that the redrawing, when combined with the skhrinkage estimation reduces the error even further. Redrawing from a GMM gives better estimation accuracy than from the MVG.

VIII. CONCLUSION

DHF filters, even though not new in the literature, are still not fully explored in detail. They lack the in-depth theoretical and numerical analysis that the other contemporary filters have gone through. Especially, the implementational details are very application specific. In this paper we have tried to point out the key factors affecting the performance of a generic DHF. Highlighted factors have been studied individually in detail, with several possible methods suggested for each of them. This include different schemes for pseudo-time discretization, numerical integration, prior covariance estimation and the redrawing. We have compared their performance in a challenging non-linear multi-target scenario, under both Gaussian and non-Gaussian measurement noises. Eulers method with exponentially spaced pseudo-time points, provides a nice trade off between the performance and the complexity. DHF with shrinkage estimation methods is shown to have outperformed the one with the sample covariace matrix or with the EKF based estimate. Finally, it is shown that a NZDCF based DHF with the shrinkage estimation and proper redrawing, can outperform a bootstrap particle filter with comparable performance within less execution time.

ACKNOWLEDGMENT

We acknowledge the support by the EU's Seventh Framework Programme under grant agreement no 607400 (TRAX-Training network on tRAcking in compleX sensor systems) http://www.trax.utwente.nl/.

APPENDIX

A. Assemblage Υ

Let $\mathbf{D}$ be the vector containing the Mahalanobis distances of the particles. We assume that the particles can be divided into L distinct sub-clusters, each cluster has the same distance to the estimated mean. In that case $\mathbf{D} = [d_1, d_2, \ldots d_L]$. This could either mean that the particles lie on hyper-balls in $\mathbb{R}^d$ with radii d_i concentric around the estimated mean, or each cluster is small enough, and far apart from others, such that it can be approximated by individual hyper-balls. Let the ith cluster has N_i number of particles such that $\sum_{i=1}^{L} N_i = N_p$.

Let the vector $\boldsymbol{\Phi}$ contain the inverse of Mahalanobis distances

$$\boldsymbol{\Phi} = \left[\left(\frac{1}{d_1}\right)_{\times N_1}, \left(\frac{1}{d_2}\right)_{\times N_2}, \ldots, \left(\frac{1}{d_L}\right)_{\times N_L} \right]$$

The sum of the vector $\boldsymbol{\Phi}$ is given by,

$$\sum_{i=1}^{L} \Phi_i = \frac{N_1}{d_1} + \frac{N_2}{d_2} + \cdots + \frac{N_L}{d_L}$$

$$= \frac{\sum_{i=1}^{L} \left(\prod_{j=1, j\neq i}^{L} d_j \right) N_i}{\prod_{j=1}^{L} d_j}$$

Therefore the normalized vector $\tilde{\boldsymbol{\Phi}}$ is given by

$$\tilde{\boldsymbol{\Phi}} = \left[\left(\frac{1}{d_1}\right)_{\times N_1}, \left(\frac{1}{d_2}\right)_{\times N_2}, \ldots, \left(\frac{1}{d_L}\right)_{\times N_L} \right] \times \frac{\prod_{j=1}^{L} d_j}{\sum_{i=1}^{L} \left(\prod_{j=1, j\neq i}^{L} d_j \right) N_i}$$

Next the sum of squares of the above vector is evaluated,

$$\begin{aligned}
\sum_{i=1}^{L} \tilde{\Phi}_i^2 &= \left[\frac{N_1}{(d_1)^2} + \frac{N_2}{(d_2)^2} + \cdots + \frac{N_L}{(d_L)^2} \right] \\
&\quad \times \left(\frac{\prod_{j=1}^{L} d_j}{\sum_{i=1}^{L} \left(\prod_{j=1, j\neq i}^{L} d_j \right) N_i} \right)^2 \\
&= \frac{N_1 \prod_{j=1, j\neq 1}^{L} (d_j)^2}{\sum_{i=1}^{L} \left(\prod_{j=1, j\neq i}^{L} (d_j)^2 \right) N_i} \cdots \\
&\quad \cdots + \frac{N_2 \left(\prod_{j=1, j\neq 2}^{L} d_j \right)^2}{\left(\sum_{i=1}^{L} \left(\prod_{j=1, j\neq i}^{L} d_j \right) N_i \right)^2} + \cdots \\
&\quad \cdots + \frac{N_L \left(\prod_{j=1, j\neq L}^{L} d_j \right)^2}{\left(\sum_{i=1}^{L} \left(\prod_{j=1, j\neq i}^{L} d_j \right) N_i \right)^2}
\end{aligned}$$

$$\sum_{i=1}^{L} \tilde{\Phi}_i^2 = \sum_{i=1}^{L} N_i \left(\frac{\prod_{j=1, j\neq i}^{L} d_j}{\sum_{k=1}^{L} \left(\prod_{j=1, j\neq k}^{L} d_j \right) N_k} \right)^2$$

and finally the assemblage Υ is given by,

$$\Upsilon = \frac{1}{\sum_{i=1}^{L} \tilde{\Phi}_i^2} = \frac{\left(\sum_{k=1}^{L} N_k \left(\prod_{j=1, j\neq k}^{L} d_j \right) \right)^2}{\sum_{i=1}^{L} N_i \left(\prod_{j=1, j\neq i}^{L} d_j \right)^2}$$

Below, we consider few special cases for the assemblage.

1) Number of clusters equals N_p: Each particle is considered as single clusters, hence each clusters has one particle with distinct distance d_i. assemblage in that case is given by,

$$\Upsilon = \frac{\left(\sum_{k=1}^{N_p}\left(\prod_{j=1,j\neq k}^{N_p} d_j\right)\right)^2}{\sum_{i=1}^{N_p}\left(\prod_{j=1,j\neq i}^{N_p} d_j\right)^2}$$

2) All particles equidistant: If $d_i \approx d$

$$\Upsilon = \frac{\left(\sum_{k=1}^{N_p} d^{N_p-1}\right)^2}{\sum_{k=1}^{N_p}(d^{N_p-1})^2} = \frac{(N_p d^{N_p-1})^2}{N_p d^{2(N_p-1)}} = \frac{N_p^2 d^{2(N_p-1)}}{N_p d^{2(N_p-1)}}$$

which leads to,

$$\Upsilon = N_p$$

3) Two dominant clusters: Now suppose that there are two main sub-cluster i.e. L=2.

$$\Upsilon = \frac{\left(\sum_{k=1}^{2} N_k\left(\prod_{j=1,j\neq k}^{2} d_j\right)\right)^2}{\sum_{i=1}^{2} N_i\left(\prod_{j=1,j\neq i}^{2} d_j\right)^2} = \frac{(d_2N_1 + d_1N_2)^2}{d_2^2N_1 + d_1^2N_2}$$

Now assume that $d_1 >> d_2$. In that case we can say in the limiting sense,

$$\lim_{d_1\to\infty} \Upsilon = \lim_{d_1\to\infty} \frac{(d_2N_1 + d_1N_2)^2}{d_2^2N_1 + d_1^2N_2}$$

$$= \frac{N_2^2}{N_2} = N_2$$

Likewise for $d_2 >> d_1$, $\lim_{d_2\to\infty} \Upsilon = N_1$.

REFERENCES

[1] F. Gustafsson
Statistical sensor fusion, 2nd ed.
Studentlitteratur AB, May 21, 2013.

[2] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp
"A tutorial on Particle Filters for Online Non-linear/Non-gaussian Bayesian Tracking,"
in *IEEE transaction on signal processing*, vol. 50, no. 2. IEEE, February 2002, pp. 174–188.

[3] N. Gordon, D. Salmond, and A. Smith
"Novel approach to nonlinear/non-Gaussian bayesian state estimation,"
in *IEE Proceedings on Radar and Signal Processing.*, Apr 1993, pp. 107–113.

[4] M. Pitt and N. Shephard
"Filtering via Simulation: Auxiliary Particle Filters,"
in *Journal of the American Statistical Association*, vol. 94, no. 446, 1999.

[5] C. Musso, N. Oudjane, and F. Le Gland
"Improving Regularised Particle Filters,"
in *Sequential Monte Carlo Methods in Practice*, ser. Statistics for Engineering and Information Science, A. Doucet, N. de Freitas, and N. Gordon, Eds. Springer New York, 2001, pp. 247–271.

[6] U. Hanebeck and O. Feiermann
"Progressive Bayesian estimation for nonlinear discrete-time systems: the filter step for scalar measurements and multidimensional states,"
in *IEEE Conference on Decision and Control*, vol. 5. IEEE, 9–12 Dec. 2003, pp. 5366–5371.

[7] J. Hagmar, M. Jirstrand, L. Svensson, and M. Morelande
"Optimal Parameterization of Posterior Densities Using Homotopy,"
in *Proceedings of the 14th International Conference on Information Fusion (FUSION)*. 5–8 July: IEEE, 2011, pp. 1–8.

[8] F. Daum and J. Huang
"Curse of Dimensionality and Particle Filters,"
in *IEEE Aerospace conference*. IEEE, 8–15 March 2003, pp. 1970–1993.

[9] F. Daum and J. Huang
"Nonlinear filters with log-homotopy,"
in *SPIE Proceedings*, September 2007.

[10] F. Daum and J. Huang
"Nonlinear filters with particle flow induced by log-homotopy,"
in *SPIE Proceedings*, May 2009.

[11] F. Daum, J. Huang, and A. Noushin
"Coulomb's law particle flow for nonlinear filters,"
vol. 8137, 2011, pp. 81 370E–81 370E-15.

[12] F. Daum and J. Huang
"Zero curvature particle flow for nonlinear filters,"
in *SPIE Proceedings*, 2013.

[13] F. Daum and J. Huang
"Particle flow with non-zero diffusion for non-linear filters,"
in *SPIE Proceedings*, 2013.

[14] M. A. Khan, M. Ulmke, B. Demissie, F. Govaers, and W. Koch
"Combining Log-Homotopy Flow with Tensor decomposition based solution for Fokker-Planck equation,"
in *2016 19th International Conference on Information Fusion (FUSION)*, July 2016, pp. 2229–2236.

[15] F. Daum and J. Huang
"Nonlinear filters with particle flow,"
in *SPIE Proceedings*, September 2009.

[16] S. Choi, P. Willet, F. Daum, and J. Huang
"Discussion and Application of Homotopy filter,"
in *SPIE Proceedings*, 2011.

[17] F. Daum and J. Huang
"Numerical experiments on nonlinear filters with exact particle flow induced by log-homotopy,"
in *Proceeding SPIE*, April 2010.

[18] L. Chen and R. Mehra
"A study of nonlinear filters with particle flow induced by log-homotopy,"
in *SPIE Proceedings*, 2010.

[19] F. Daum and J. Huang
"Friendly rebuttal to Chen and Mehra: incompressible particle flow for nonlinear filters,"
vol. 8392, 2012, pp. 839 210–839 210-6. [Online]. Available: http://dx.doi.org/10.1117/12.915510.

[20] F. Daum and J. Huang
"Generalized particle flow for nonlinear filters,"
in *SPIE Proceeding*, April 2010.

[21] T. Ding and M. Coates
"Implementation of Daum-Huang Exact Flow Particle Filter,"
in *IEEE Statistical Signal Processing Workshop (SSP)*, 2012.

[22] K. Bell and L. Stone
"Implementation of the homotopy particle filter in the JPDA and MAP-PF multi-target tracking algorithms,"
in *Information Fusion (FUSION), 2014 17th International Conference on*, July 2014, pp. 1–8.

[23] S. Datta Gupta, J. Y. Yu, M. Mallick, M. Coates, and M. Morelande
"Comparison of angle-only filtering algorithms in 3d using EKF, UKF, PF, PFF, and ensemble KF,"
in *Information Fusion (Fusion), 2015 18th International Conference on*, July 2015, pp. 1649–1656.

[24] C. Kreucher, K. Bell, and D. Sobota
"A comparison of tracking algorithms for supermaneuverable targets,"
in *Information Fusion (Fusion), 2015 18th International Conference on*, July 2015, pp. 534–541.

[25] N. Moshtagh and M. Chan
"Multisensor fusion using homotopy particle filter,"
in *Information Fusion (Fusion), 2015 18th International Conference on*, July 2015, pp. 1641–1648.

[26] P. Bunch and S. Godsill
"The Progressive Proposal Particle Filter: Better Approximations to the Optimal Importance Density,"
Tech. Rep., 2014.

[27] M. Khan and M. Ulmke
"Non-linear and non-Gaussian state estimation using log-homotopy based Particle Flow Filters,"
in *Sensor Data Fusion: Trends, Solutions, Applications (SDF), 2014*, Oct 2014, pp. 1–6.

[28] C. W. Gear
"The automatic integration of stiff ordinary differential equations,"
in *Information processing 68 (Proc. IFIP Congress, Edinburgh, 1968)*, vol. 1, 1969, pp. 187–193.

[29] J. Schäfer and J. Strimmer
"A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics,"
2005.

[30] J. Schäfer and J. Strimmer
"An empirical Bayes approach to inferring large-scale gene association networks,"
Bioinformatics, vol. 21, pp. 754–764, 2005.

[31] C. Stein
"Inadmissibility of the usual estimator for the mean of a multivariate normal distribution,"
in *Proceedings of the Third Berkeley Symposium on Mathematical and Statistical Probability.*, vol. 1, no. 446, 1956, pp. 197–206.

[32] O. Ledoit and M. Wolf
"Well-conditioned estimator for large-dimensional covariance matrices,"
Journal of Multivariate Analysis, vol. 88, no. 2, pp. 365–411, 2004.

[33] O. Ledoit and M. Wolf
"Honey, I shrunk the sample covariance matrix,"
The Journal of Portfolio Management, vol. 30, no. 4, pp. 110–119, 2004.

[34] O. Ledoit and M. Wolf
"Improved estimation of the covariance matrix of stock returns with an application to portfolio selection,"
Journal of Empirical Finance, vol. 10, no. 5, pp. 603–621, 2003.

[35] L. Haff
"Emprirical Bayes Estimation of the Multivariate Normal Covariance Matrix,"
Ann. Statist., vol. 8, no. 3, pp. 586–597, 1980.

[36] C. Stein
"Estimation of a covariance matrix, Rietz Lecture,"
39th Annual Meeting IMS, Atlanta, GA, Tech. Rep., 1975.

[37] S. Lin and M. Perlman
"A Monte Carlo Comparison of four estimators of a Covariance Matrix,"
Tech. Rep. 44, 1984.

[38] C. Stein
"Series of lectures given at the University of Washington,"
Seattle, Tech. Rep., 1982.

[39] O. Ledoit and M. Wolf
"Nonlinear shrinkage estimation of large-dimensional covariance matrices,"
The Annals of Statistics, vol. 40, no. 2, pp. 1024–1060, April 2012.

[40] C. M. Bishop
Pattern Recognition and Machine Learning.
Springer, 2006.

[41] M. Kristan, A. Leonardis, and D. Skočaj
"Multivariate Online Kernel Density Estimation with Gaussian Kernels,"
Pattern Recognition, vol. 44, pp. 2630–2642, 2011.

[42] M. A. Khan and M. Ulmke
"Improvements in the Implementation of Log-Homotopy based Particle Flow Filters,"
in *Information Fusion (Fusion), 2015 18th International Conference on*, July 2015, pp. 74–81.

[43] "Online Kernel Density Estimation with Gaussian Kernels and Online Discriminative Kernel Density Estimation with Gaussian Kernels,"
http://www.vicos.si/File:Maggot_v3.4.zip.

[44] F. Daum, J. Huang, and A. Noushin
"Numerical experiments for Coulomb's law particle flow for nonlinear filters,"
vol. 8137, 2011.

Muhammad Altamash Ahmed Khan received his B.E. in electrical engineering from the National University of Sciences and Technology (NUST), Islamabad, Pakistan, in 2006 and the M.Sc. in wireless systems from KTH Royal Institute of Technology, Stockholm, Sweden in 2011. From 2011 to 2013, he worked as Research Engineer with the Lab for Automatic Control and at the Lab for Communication Networks, KTH. During that period he worked on the statistical characterization of image/video feature extraction, implementation and testing of motion control for networked robots, and implementing the routing and MAC protocols for wireless sensor networks. Since April 2014, he has been working as a Marie Curie Fellow at FKIE Fraunhofer, Wachtberg Germany. His research interests include non-linear state estimation methods, multi-target tracking, and wireless sensors networks.

Martin Ulmke is a theoretical physicist and received his Dipl.-Phy. (M.S.) and Dr. rer. nat. (Ph.D.) degrees from the Aachen Technical University (RWTH), Germany, in 1991 and 1995, respectively. From 1995 to 1998 he was a research associate in condensed matter theory at the University of California in Davis and at the University of Augsburg, Germany. He was a systems engineer with MTU Aero Engines, Munich, from 1998 to 2001. Since 2001 he is part of the scientific staff in the Department of Sensor Data and Information Fusion at the Fraunhofer FKIE where he is head of Research Group Wide Area Surveillance.

Wolfgang Koch studied physics and mathematics at Aachen Technical University (RWTH), Germany, where he earned a "Dr. rer. nat." degree in theoretical physics. He is Head of the Department Sensor Data and Information Fusion at Fraunhofer FKIE, a German defence research lab. At Bonn University, he is teaching Applied Computer Science as a "Privatdozent." Dr. Koch serves the IEEE AESS as an Associate Editor-in-Chief and Technical Editor of Target Tracking and Multisensor Systems for the IEEE Transactions on Aerospace and Electronic Systems. In 2013, he served as President of the International Society of Information Fusion (ISIF). In 2016, he was General Co-Chair of the IEEE ISF International Conference on Information Fusion, Heidelberg, Germany.