18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

Multisensor Fusion Using Homotopy Particle Filter

Nima Moshtagh
Advanced Technology Center
Lockheed Martin Space Systems Company
Sunnyvale, CA. USA
nima.moshtagh@lmco.com

Moses W. Chan
Advanced Technology Center
Lockheed Martin Space Systems Company
Sunnyvale, CA. USA
moses.w.chan@lmco.com

Abstract - Homotopy particle filters (HPFs), recently developed by Daum and Huang [1], present an alternative nonlinear filtering approach to sampling-based particle filters. Homotopy filters perform information update using the flow of particles to regions with high measurement likelihood. The particle flows in HPFs are solutions to a Fokker-Plank equation governing the dynamics of the posterior density function. The resulting partial differential equation is highly under-determined and has many solutions. In this work we study the nonzero-diffusion flow, and show its advantage in multisensor fusion. The nonzero-diffusion flow was chosen because it has the form of an information filter (inverse-covariance filter), and this feature makes it suitable for integration of measurement contributions from different sensors. The effectiveness of the HPF with nonzero diffusion flow as a fusion mechanism was evaluated for tracking a moving target using multiple range and bearing sensors. It is shown that the HPF requires several orders of magnitude fewer particles than a sampling-based particle filter, without loss in state estimator performance.

Keywords: Homotopy Particle Filter, Flow Control, Target Tracking, Multisensor Fusion

1 Introduction

Similar to a sampling-based particle filter, a homotopy particle filter (HPF) represents the posterior distribution using a set of particles. However, instead of updating the weights of particles using the likelihood function, HPF controls the flow of the particles during the filter's information update step. The motion of the particles is determined by a solution to a partial differential equation (PDE). As we will see in Section IV, the PDE is highly under-determined, and many solutions (particle flows) can be found. Daum and Huang introduced this new approach to Bayesian filtering, and have developed a number of interesting solutions over the past few years, including exact flow [1], incompressible flow [2], and nonzero diffusion flow [3] to name a few.

Applications of the HPF to tracking problems have been studied recently [4], [5] and [6]. Choi et al. [4] evaluated the performance of homotopy filters with incompressible flow and exact flow in tracking a planar target with (linear) position measurements. Bell and Stone [5] used a homotopy filter with exact flow for a multi-target tracking problem with a linear observation model. Ding and Coates [6] implemented a homotopy filter with exact flow to track multiple targets using acoustic amplitude sensors. In all the above implementations, HPF algorithms rely on an extended/unscented Kalman filter (EKF/UKF) that is executed in parallel. Even though the reliance on EKF is not intrinsic to the exact flow filter, the EKF was used to provide covariance matrices needed in evaluating the flow. Ding and Coates [6] illustrated that the performance of the HPF can degrade when the EKF/UKF fail.

In this work we chose to use an HPF with *nonzero diffusion flow* because (a) it can be implemented without reliance on an EKF to run in parallel, (b) the update expression of the nonzero diffusion flow has the form of an *information* (*inverse-covariance) filter* which lends itself to multi-sensor fusion quite well. The HPF with nonzero diffusion was used as a fusion mechanism to track a moving target using multiple sensors with nonlinear measurement models (range or bearing). Our implementation of HPF with nonzero diffusion does linearize the measurement models but it does so for each particle rather than the point estimate (as in EKF). Thus, the HPF can achieve very good estimation accuracies with several orders of magnitude fewer particles compared to a sampling-based particle filter. Simulation results show the effectiveness of the nonzero diffusion HPF under varying number of particles, and simulation parameters.

Sections II and III provide background information about Bayesian filtering and sampling-based particle filtering. Section IV explains the theory behind the HPFs, and derives the PDE that provides the particles flow. Section V summarizes the derivation of nonzero-diffusion flow following the work in [2]. Section VI shows how HPF with nonzero diffusion flow can be used to fuse measurements from multiple sensors. Simulation results are presented in Section VII, where we evaluate the effectiveness of HPF with nonzero diffusion flow in tracking of a moving target.

Approved for Public Release 15-MDA-8218 (27 April 15)
DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

2 Bayesian Filter

Let x_t represent the state of a system at time t. Consider the following dynamical system modeling the evolution of the system over time:

$$x_t = f_t(x_{t-1}) + v_{t-1}\,, \tag{1}$$

where $f_t(\cdot)$ is a (possibly nonlinear) d-dimensional transition function, and v_t is the process noise. We assume the Markov property holds for the system, i.e. $p(x_t|x_{t-1}, \dots x_1) = p(x_t|x_{t-1})$. The measurements taken from the system are modelled as

$$z_t = h(x_t, n_t) \tag{2}$$

where n_t is a measurement noise. The measurement z_t is conditionally independent from the past states given the

$$w_t^i \propto p\big(z_t|x_t^i\big). \tag{3}$$

current state, i.e. $(z_t|z_{t-1}, \dots z_1, x_t, \dots x_1) = p(z_t|x_t)$, where $p(z_t|x_t)$ is known as the *likelihood function*. The objective of state estimation is to recursively estimate the state x_t from all available measurements $Z^{1:t} = \{z_1, z_2, \dots z_t\}$. More formally, we would like to estimate the conditional probability density function $p\left(x_t\middle|Z^{1:t}\right)$, which is known as the *posterior*. In Bayesian filtering the posterior is computed in a two-step recursion. In the *prediction* step the current posterior $p\left(x_{t-1}\middle|Z^{1:t-1}\right)$ is propagated to the next time step using the transition function

$$p(x_t|Z^{1:t-1}) = \int p(x_t|x_{t-1}).\, p(x_{t-1}|Z^{1:t-1})dx_{t-1}\,. \tag{4}$$

When measurement z_t arrives, the *update* step is executed. First the likelihood $p(z_t|x_t)$ is computed, and then it is used to update the posterior

$$p(x_t|Z^{1:t}) = \frac{p(x_t|Z^{1:t-1})p(z_t|x_t)}{p(z_t|Z^{1:t-1})}, \tag{5}$$

where $p(z_t|Z^{1:t-1})$ is the normalization constant. The pair of recursive equations (4) and (5) provides the optimal Bayesian solution for the posterior. However, the closed-form solution only exists for some special cases as in *Kalman filtering* for linear Gaussian systems. Estimation methods such as *particle filtering* approximate the optimal Bayesian solution for nonlinear systems.

3 Particle Filters

Particle filters provide approximate solutions to a Bayesian filter by representing the posterior as a set of random samples and their associated weights $\{x^i, w^i\}_{i=1}^N$. Particles are drawn from a proposal density according to their weights (*importance sampling*). Formally, the posterior is represented by a discrete approximation

$$p\big(x_t|Z^{1:t}\big) \approx \sum_{i=1}^{N} w_t^i \delta\big(x_t - x_t^i\big),$$

where $\delta(\cdot)$ is the Dirac delta function. As in Bayesian filtering the posterior is computed in a recursion. In the propagation step the transition function $p(x_t|x_{t-1})$ is used to propagate the particles and compute the predicted density (4). When measurement z_t becomes available, the likelihood function $p(z_t|x_t^i)$ is computed for all particles and the weights of the particles are updated. In the end, particles are resampled according to their weights to avoid sample impoverishment [7]. This recursive algorithm is known as *Sampling Importance Resampling* (SIR) particle filter. In SIR particle filter, the weights of the particles are proportional to the likelihood function. Table below summarizes the measurement update step of the SIR particle filter.

	Particle Filter - SIR (Measurement Update)
1	**For** $x^i,\ i = 1, \dots, N$
2	*Likelihood*: Given z_t Compute $p\big(z_t\big|x_t^i\big)$
3	*Update weight*: $w_t^i = w_{t-1}^i\, p\big(z_t\big|x_t^i\big)$.
4	**End**
5	*Normalize* weights $w_t^i \leftarrow w_t^i / \sum_k w_t^k$
6	*Resample*: draw new particles based on the weights.

Table 1: Measurement Update for PF-SIR

The problem with essentially all particle filters is the "particle degeneracy" problem. To understand the problem better let us consider the un-normalized conditional probability density function

$$p(x) = g(x)l(x)\,,$$

where $l(x)$ is the likelihood, and $g(x)$ is the predicted density (see Equation (5)). $g(x)$ is given by the set of particles, and the likelihood $l(x)$ is known analytically. The difficulty arises from the multiplication of these two functions. If not enough particles are used, the posterior will have many particles with (nearly) zero weights, and very few non-zero particles. This phenomenon is known as *particle degeneracy* [7]. The problem is exacerbated when measurements are very accurate. Also the performance of sampling-based particle filters deteriorates significantly for problems with high dimensional state vectors. The number of particles that are needed to obtain the same level of performance grows exponentially with state dimension [8].

4 Homotopy Particle Filters

To mitigate the problems with sampling-based particle filters (e.g. particle degeneracy, exponential growth of number of particles, etc.), Daum and Huang [1] proposed an alternative approach to nonlinear filtering known as the *particle flow* or *homotopy particle filter*. In the particle flow

approach particles are essentially moved to regions in the state space where the posterior has larger values. The particles flow as the *homotopy* variable varies continuously from 0 to 1.

The flow in the un-normalized posterior is generated using a homotopy

$$p(x,\lambda) = g(x)l(x)^{\lambda}$$

as a function of parameter $\lambda \in [0\ 1]$. The flow of the posterior corresponds to the Bayes rule. At the start of the flow ($\lambda = 0$), we have $p(x,\lambda) = g(x)$, i.e. $p(x,\lambda)$ is equal to the prior density. At the end of the flow ($\lambda = 1$), $p(x,\lambda)$ is equal to the desired posterior density $p(x,\lambda) = g(x)l(x)$.

The flow of the logarithm of the posterior with respect to is given by:

$$\log p(x,\lambda) = \log g(x) + \lambda \log\ l(x)\,. \qquad (6)$$

This represents a *line homotopy* of the logarithm of the densities. The task is to find an appropriate flow of the probability density defined by the log-homotopy (6). Suppose the flow for the Bayes rule obeys the following stochastic differential equation:

$$dx = f(x,\lambda)d\lambda + dw \qquad (7)$$

where dw is the diffusion noise (or flow nosie) with covariance matrix $Q(x)$. Note that the objective is to compute the vector field (flow) $f_\lambda = f(x,\lambda)$. Using the Fokker-Plank equation governing the dynamics of the posterior density function, Daum and Huang [1] derived the following first-order PDE in the unknown function f_λ (see APPENDIX for derivation):

$$\frac{\partial \log p}{\partial x} f_\lambda = -\log l - \mathrm{div}(f_\lambda) + \frac{1}{2p}\mathrm{div}\left[Q(x)\frac{\partial p}{\partial x}\right]. \qquad (8)$$

This PDE is highly under-determined because there is only one scalar valued equation, but the unknown function $f(x,\lambda)$, or the *flow*, is a D-dimensional vector field. Our objective is to solve equation (8), given $p(x,\lambda)$ and $l(x,\lambda)$. There are many ways to solve this equation for finding an appropriate flow. In Section V, a solution is derived by imposing mild assumptions on the diffusion parameter Q. Necessary and sufficient conditions for the existence of the particle flow are discussed in [17].

A *homotopy particle filter* is similar to a standard particle filter where the posterior is represented by a set of particles x^i. However, instead of updating particles weights using (3) a homotopy filter uses a particle flow (a solution of the flow PDE (8)) to update the particles states as the homotopy parameter varies from 0 to 1. The particles are migrated in small steps using the Euler's method:

$$x^i(\lambda_k) = x^i(\lambda_{k-1}) + \Delta_k \cdot f(x^i_{k-1}, \lambda_k)\,,$$

where the step size is $\Delta_k = \lambda_k - \lambda_{k-1}$. No particle resampling is needed in a homotopy particle filter. Table below shows the update step in a HPF with a generic particle flow.

	Homotopy Particle Filter (Measurement Update)
1	Compute particles mean μ_s and covariance P_s
2	**For** $\lambda_k \in [0, \lambda_2, \dots, \lambda_{M-1}, 1]$
3	$dx/d\lambda$ `= compute_flow(`μ_s, P_s, λ_k`)`
4	Move particles $x^i_k = x^i_{k-1} + \Delta_k(dx/d\lambda)$
5	**End**
6	Compute state estimates from the updated particles

Table 2: Measurement Update for HPF

The `compute_flow` function could implement any of the particle flows such as exact flow, incompressible flow, or nonzero diffusion flow.

5 HPF with Nonzero Diffusion Flow

In this section we follow the work in [2] in derivation of the HPF with nonzero diffusion flow. Consider the PDE (8) with unknown f_λ. Suppose the diffusion parameter Q is nonzero, and the prior density $g(x)$ and likelihood function $l(x)$ are twice differentiable. Now compute the gradient of (8) with respect to x

$$\frac{\partial \log l}{\partial x} = -f_\lambda^T \frac{\partial^2 \log p}{\partial x^2} - \frac{\partial \mathrm{div}(f_\lambda)}{\partial x} - \frac{\partial \log p}{\partial x}\frac{\partial f_\lambda}{\partial x} + \frac{\partial \mathrm{div}\left[Q(x)\frac{\partial p}{\partial x}/2p\right]}{\partial x}\,. \qquad (9)$$

The above equation is actually a system of D equations with D unknown functions f_λ. If a nonzero diffusion Q and flow f_λ exist such that the last three terms are summed up to zero, we get the simpler equation:

$$\left(\frac{\partial \log l}{\partial x}\right) = -f_\lambda^T\left(\frac{\partial^2 \log p}{\partial x^2}\right). \qquad (10)$$

Therefore, under the stated assumptions, the unique solution for f_λ is given by:

$$f_\lambda = -\left(\frac{\partial^2 \log p}{\partial x^2}\right)^{-1}\left(\frac{\partial \log l}{\partial x}\right)^T. \qquad (11)$$

Since the likelihood function l is analytically given, the derivative of $\log l$ can be computed analytically as well. To compute the Hessian of $\log p$, the definition of log-homotopy (6) must be used:

$$\frac{\partial^2 \log\ p}{\partial x^2} = \frac{\partial^2 \log\ g}{\partial x^2} + \lambda\ \frac{\partial^2 \log\ l}{\partial x^2}\ . \qquad (12)$$

The Hessian of $\log l$ is computed in closed-form. But computing the Hessian of $\log g$ is more difficult. An approach using the k-nearest neighbor algorithm was proposed in [9]. Hessian of $\log p$ is also known as the *observed Fisher information matrix*.

If a Gaussian approximation to g is used, the Hessian of $\log g$ can be approximated by the sample covariance matrix P_s of the prior computed from the set of particles at $\lambda = 0$. For applications where the nonlinear measurement has additive Gaussian noise, the information matrix (Hessian of $\log p$ evaluated at x^i) becomes

$$I(x^i, \lambda) = \frac{\partial^2 \log\, p}{\partial x^2}\Big]_{x^i} = -P_s^{-1} - \lambda H(x^i)^T R^{-1} H(x^i)\,.$$

Thus using (11) the nonzero-diffusion flow of each particle becomes

$$f_\lambda(x^i) = -I(x^i, \lambda)^{-1} H(x^i)^T R^{-1}(z - h(x^i)), \qquad (13)$$

where $H(x^i)$ is the linearized measurement matrix evaluated at x^i; R is the measurement noise covariance. Note that this flow is similar to the *Extended Information filter* [10] but for each particle rather than the conditional mean.

Note: In numerical experiments the nonzero-diffusion flow requires much finer integration steps compared with the exact flow. This will add to the computational requirements of this flow.

6 Multisensor Fusion and Tracking

Given the similarity of the nonzero diffusion flow to an information filter, the multi-sensor estimation problem becomes considerably simple when the sensors are independent. As it is shown next, the flow of particles for the group-sensor filter is derived from the linear combination of the information from individual sensors.

Let us consider S sensors, each observing a common state according to

$$z_s(t) = h_s\big(x(t)\big) + n_s(t), \quad s = 1, \dots, S \qquad (14)$$

where the noise $n_s(t)$ are assumed to be white and uncorrelated in both time and among the sensors:

$$\mathrm{E}\{n_s(t)\} = 0, \quad \mathrm{E}\big\{n_s(t) n_p(k)\big\} = \delta_{sp}\delta_{tk} R_s\ . \qquad (15)$$

Suppose the sensors make synchronized observations. We form the group measurement vector $\boldsymbol{z}(t)$, group (linearized) measurement matrix $\boldsymbol{H}(t)$, and group noise vector $\boldsymbol{v}(t)$:

$$\boldsymbol{z}(t) = \begin{bmatrix} z_1(t) \\ \vdots \\ z_s(t) \end{bmatrix}, \quad \boldsymbol{H}(t) = \begin{bmatrix} H_1(t) \\ \vdots \\ H_s(t) \end{bmatrix}, \quad \boldsymbol{v}(t) = \begin{bmatrix} v_1(t) \\ \vdots \\ v_s(t) \end{bmatrix}\ .$$

The measurement covariance for the group noise vector is

$$\boldsymbol{R}(t) = \mathrm{E}\{\boldsymbol{v}(t)\boldsymbol{v}(t)^{\boldsymbol{T}}\} = \mathrm{blockdiag}(R_1(t), \dots, R_s(t))\,.$$

To obtain the flow expression for the multi-sensor estimation problem, we start by writing the flow and the information update equations for the group-sensor system:

$$f_\lambda(x) = -I^{-1}\boldsymbol{H}^T\boldsymbol{R}^{-1}\big(\boldsymbol{z} - \boldsymbol{h}(x)\big)\,, \qquad (16)$$

and

$$I = -P_s^{-1} - \lambda \boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H}\,. \qquad (17)$$

Using the above definitions, we have:

$$\boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H} = \begin{bmatrix} H_1(t) \\ \vdots \\ H_s(t) \end{bmatrix}^T \begin{bmatrix} R_1^{-1}(t) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R_s^{-1}(t) \end{bmatrix} \begin{bmatrix} H_1(t) \\ \vdots \\ H_s(t) \end{bmatrix} = \sum_{s=1}^{S} H_s^T R_s^{-1} H_s \qquad (18)$$

and similarly

$$\boldsymbol{H}^T\boldsymbol{R}^{-1}\big(\boldsymbol{z} - \boldsymbol{h}(x)\big) = \sum_{s=1}^{S} H_s^T R_s^{-1}(z_s - h_s(x))\,. \qquad (19)$$

Substituting equations (18) and (19) into equations (16) and (17), the nonzero diffusion flow for the group-sensor system becomes

$$f_\lambda(x^i) = -I(x^i, \lambda)^{-1} \sum_{s=1}^{S} H_s^T(x^i) R_s^{-1}\left(z_s - h_s(x^i)\right), \qquad (20)$$

with the information update equation given by

$$I(x^i, \lambda) = -P_s^{-1} - \lambda \sum_{s=1}^{S} H_s^T(x^i) R_s^{-1} H_s(x^i). \qquad (21)$$

Thus the multi-sensor fusion algorithm with a nonzero diffusion flow would use equation (21) to compute the information matrix and then use equation (20) to compute the flow of the HPF, as summarized in the table below.

	Nonzero Diffusion Flow (`compute_flow`)
1	**Inputs**: • Homotopy parameter λ_k • Particles mean μ_s (at $\lambda_1 = 0$) • Particles covariance P_s (at $\lambda_1 = 0$)
2	**For** $x^i,\ i = 1, \dots, N$
3	Compute information update $I(x^i, \lambda_k)$ using (21)
4	Compute the multi-sensor flow $f_\lambda(x^i)$ using (20)
5	**End**
6	**Output**: particle flow $dx/d\lambda = f_\lambda$ for all particles.

Table 3: Nonzero Diffusion Flow

7 Simulation Studies

We compared the performances of an HPF with nonzero-diffusion flow (HPF-NZD) and an SIR particle filter (PF-SIR) in a tracking problem involving a planar target. Target's motion is observed using multiple sensors with states $X_s, s = 1, \ldots, S$. The measurement models are given by equation (14) where for a bearing sensor we have

$$h_s(X, X_s) = \tan^{-1}\left(\frac{x - x_s}{y - y_s}\right), \quad \text{and} \quad n_s \sim \mathcal{N}(0, R_b) \qquad (22)$$

and for a range sensor we have

$$h_s(X, X_s) = \sqrt{(x - x_s)^2 + (y - y_s)^2}, \text{ and } n_s \sim \mathcal{N}(0, R_r) \ . \qquad (23)$$

Note that measurements are nonlinear functions of target state with additive Gaussian noise. The bearing is measured clockwise relative to the horizontal line. The sensors take measurements of the target relative to the sensor location, and then measurements are passed to a fusion center where it runs PF-SIR and HPF-NZD algorithms. For both algorithms the initial particles are drawn from a Uniform distribution in the position space, and a zero-mean Normal distribution for the velocity and acceleration spaces. As measurements arrive, the HPF is implemented using the measurement update algorithm in Table 2 with particle flow given in Table 3. The step sizes of the homotopy parameter λ are important for convergence of the HPF. We divided the [0 1] range to 15 intervals with the first 5 intervals being between 0 and 0.1.

Next we present the results of our study for two different scenarios. The first scenario (Section VII.A) considers tracking an accelerating target. The motion model (used in the filters) is linear but the measurement models are nonlinear (equations (22) and (23)). In the second scenario (Section VII.B), we use a nonlinear motion model suitable for tracking maneuvering targets.

Each scenario is run 100 times for PF-SIR and HPF-NZD with different random initial conditions for the particles. State estimation errors are computed as the difference between the true target state and the estimated state. The magnitudes of position and velocity errors are defined as

$$\varepsilon_{pos}(t) = \sqrt{\varepsilon_1^2 + \varepsilon_2^2}, \qquad \varepsilon_{vel}(t) = \sqrt{\varepsilon_3^2 + \varepsilon_4^2}\ ,$$

where ε_i denotes the estimation error for state dimension i. Errors are averaged over all 100 runs. Also, the percentages of convergence[2] are computed as the ratio of the converged runs to total number of runs.

[2] In our study we say the state estimator converged when the position estimation errors remain less than 2 meters after a short initial period (10 time steps).

7.1 Linear Motion & Nonlinear Measurement Models

In this scenario, the target moves across the two-dimensional plane (see Figure 1) and multiple sensors are taking measurements at constant intervals dt. Target's state vector is given by its position, velocity, and acceleration $X_t = [x\ y\ v_x\ v_y\ a_x\ a_y]_t^T$. Target's motion is modeled by the *nearly-constant-acceleration* model:

$$\dot{X}_t = AX_t + Bw_t \qquad (24)$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \otimes I_2\ , \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \otimes I_2\ ,$$

and w_t is the process noise, $\otimes$ is the Kronecker product, and I_2 is the 2×2 identity matrix.

Figure 2 shows the state estimation results using a PF-SIR and a HPF-NZD. As the number of particles N varies from 10 to 1000, the estimated position, velocity, and acceleration errors for the HPF-NZD remain relatively constant. However PF-SIR is not able to track the target for those numbers of particles. Each point in Figure 2 is the average of the accumulated errors over 100 Monte Carlo runs (The accumulated position (velocity) error is the sum of position (velocity) errors over the duration of the simulation). Most of the error is due to the fact that PF-SIR does not track the target, as it can be seen in Figure 3. PF-SIR achieves acceptable tracking performance with N=10000 particles.

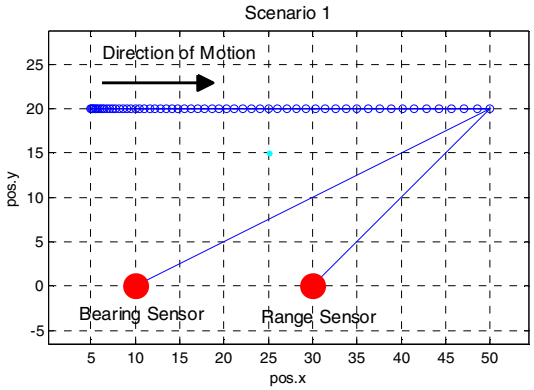

Figure 1: Scenario 1. Tracking an accelerating target using range and bearing sensors located at [30,0] and [10,0] respectively.

Figure 4 compares the velocity error magnitudes of the HPF-NZD with N=100 and PF-SIR with N=10000 particles. HPF-NZD achieve the same performance as PF-SIR with two orders of magnitude fewer particles. The average error is computed over 100 Monte Carlo runs. Some statistics of the 100 runs are shown in Figure 5. The 50% and 97%

graphs show the mean and covariance (3σ) of the velocity estimation error.

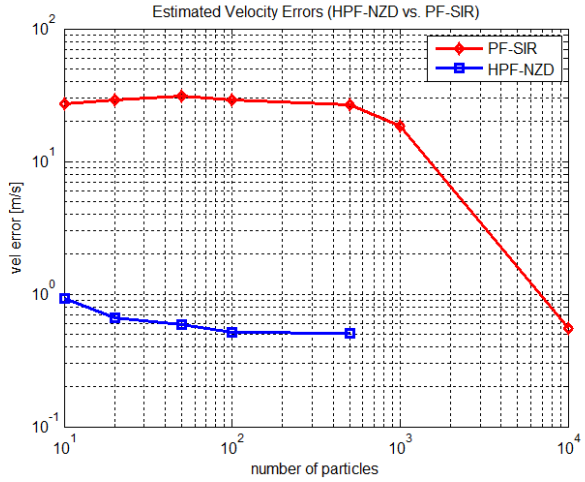

Figure 2: Estimation Error vs. Number of Particles

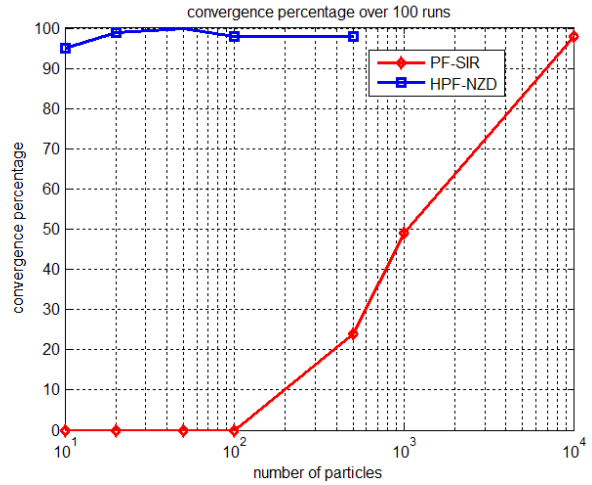

Figure 3: Comparison of the convergence percentages.

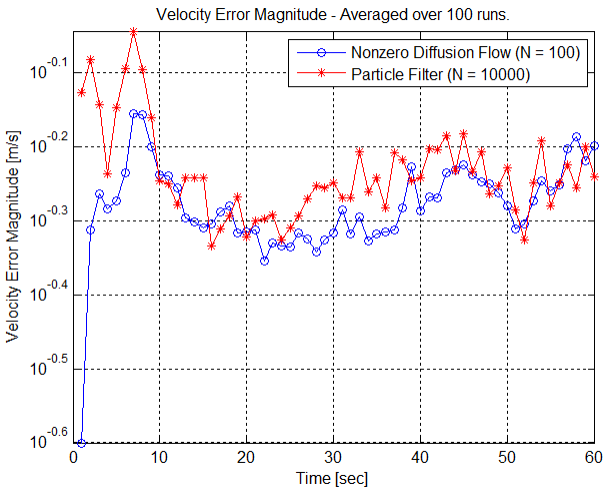

Figure 4: Comparison of velocity errors averaged over 100 runs.

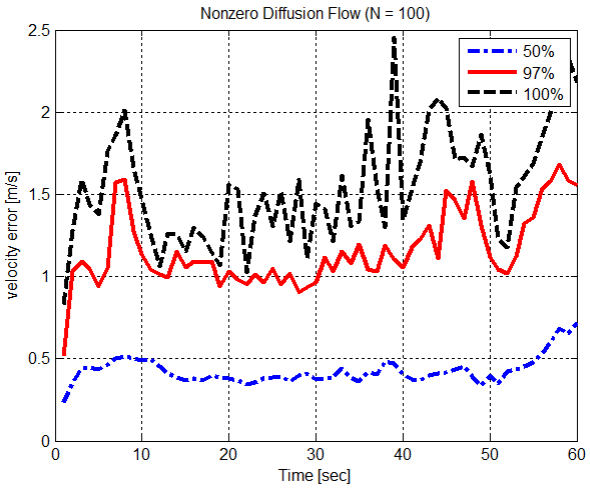

Figure 5: Velocity Error Statistics of 100 Runs for HPF-NZD with N=50 particles.

7.2 Nonlinear Motion & Nonlinear Measurement Models

In this scenario target is moving in a circular pattern and a pair of range and bearing sensors are taking measurements at constant intervals dt. Target's state vector is given by its position [x y], speed v, heading angle φ, and turning rate, $X_t = [x \quad y \quad \varphi \quad v \quad \omega]_t^T$. Target's dynamics is modeled by the *coordinate-turn model with polar velocity* [11]

$$\dot{X}_t = f(X_t) + w_t \tag{25}$$

where

$$f(X_t) = \begin{bmatrix} v(t)\cos(\varphi(t)) \\ v(t)\sin(\varphi(t)) \\ \omega(t) \\ 0 \\ 0 \end{bmatrix}, \quad w_t = \begin{bmatrix} 0 \\ 0 \\ 0 \\ w_v(t) \\ w_\omega(t) \end{bmatrix},$$

and w_v and w_ω are the process noise for target speed and turning rate. Figure 6 shows the circular trajectory of the target and sensors' locations.

Figure 7 shows that as the number of particles increases the accumulated position error decreases for PF-SIR but HPF-NZD is quite robust to changes in the number of particles. This is actually directly related to the increase in the convergence percentage (Figure 8). For HPF-NZD all 100 runs converged while PF-SIR convergence improves gradually as more particles are used. Eventually, for N=50000 particles, PF-SIR achieves similar performance as HPF-NZD (see also Figure 9). Some statistics associated with the 100 runs of the HPF-NZD with N=100 particles are shown in Figure 10. It shows that HPF-NZD can track the target with small errors.

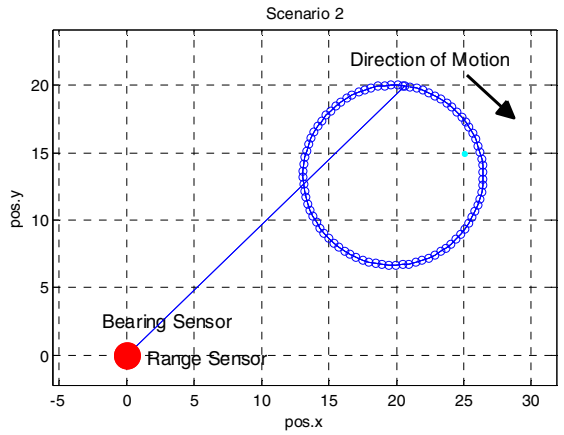

Figure 6: Scenario 2. Tracking a target moving in a circular pattern using bearing and range sensors located at [0,0].

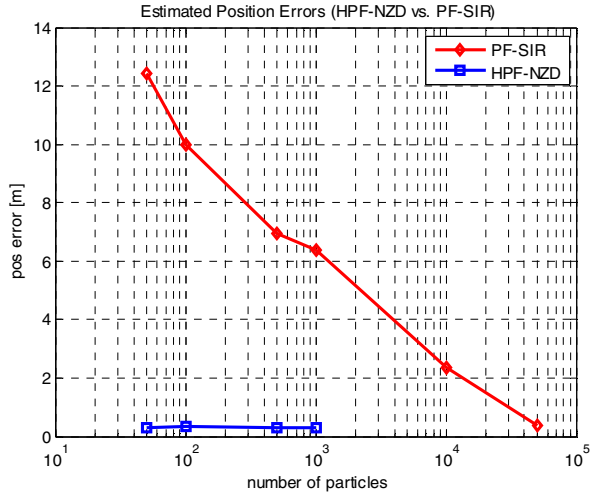

Figure 7: Position Error vs. Number of Particles

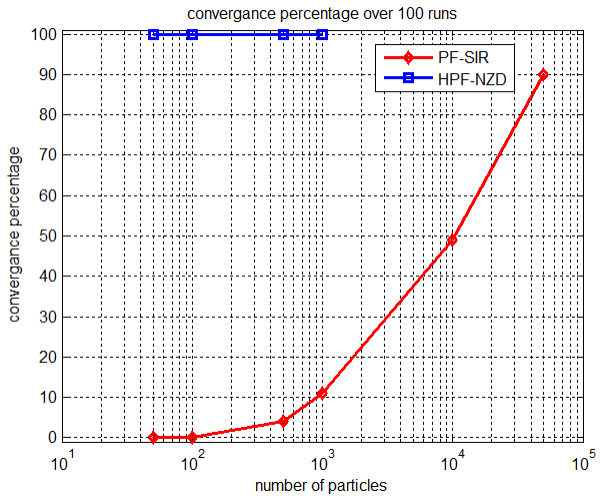

Figure 8: Convergence Percentage vs. Number of Particles

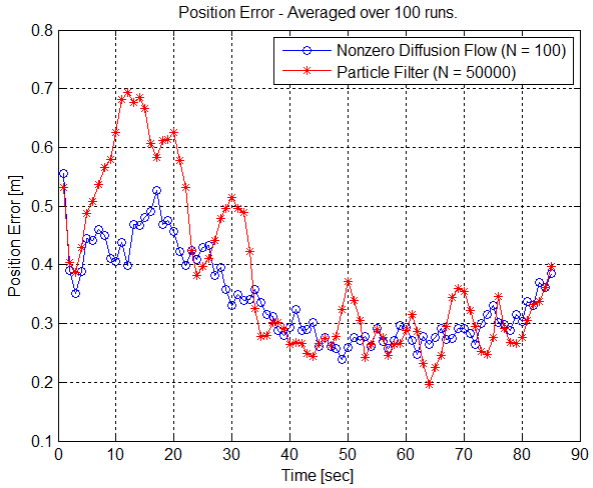

Figure 9: Position Estimation Errors over Time.

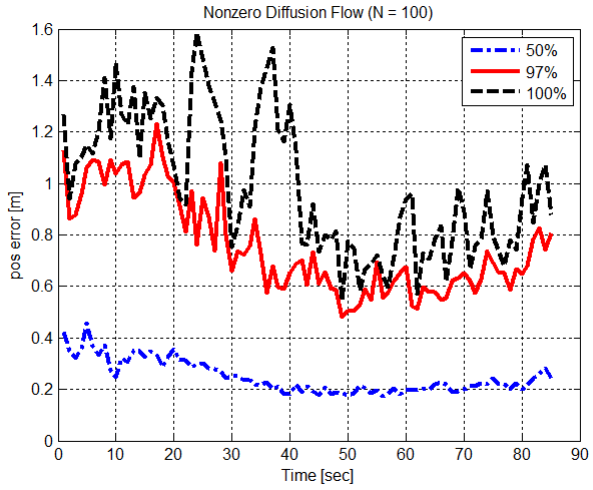

Figure 10: Position Error Statistics over 100 runs for HPF-NZD with N=100 particles (for scenario 2).

8 Summary and Conclusions

We showed how to use the homotopy particle filter with nonzero diffusion flow (HPF-NZD) as a fusion mechanism to fuse information from multiple sensors. We chose to work with the nonzero-diffusion flow because it has the form of an information filter (inverse-covariance filter), and that feature makes it suitable for integration of measurement contributions as they arrive at the fusion center. We ran a number of studies with increasing degrees of difficulty. We started with an easy tracking problem (linear motion model + linear position measurements) to more difficult tracking problems (nonlinear motion model + multiple nonlinear measurements). Our extensive comparative studies showed that HPF with nonzero diffusion flow outperforms SIR particle filter. As the dimension of the state space became larger, PF-SIR needed orders of magnitude more particles to achieve the same level of performance as HPF-NZD. However, the computational cost of HPF-NZD ended up being larger than PF-SIR. This was due to the need to

compute the Hessian of the log-likelihood and information matrix inversion in the flow equation (20) for every particle.

The application of the HPF to target tracking with focal plane measurements is the subject of ongoing work. Particle filter implementations of track-before-detect algorithms [14], [15] have shown great potential in tracking low-SNR targets. However, a large number of particles are needed to achieve good results. This is true especially when measurements from multiple focal planes are fused [16] to generate 3D state estimates. In future works we will investigate the application of HPF to such multisensor fusion problems.

Appendix

Consider the log-homotopy equation (6)

$$\log p(x,\lambda) = \log g(x) + \lambda \log l(x)\,.$$

Differentiating the log-homotopy w.r.t. parameter λ we have

$$\frac{\partial p}{\partial \lambda} = p.\log l\,, \tag{26}$$

where the assumption is $p(x,\lambda)$ is nowhere vanishing and differentiable. On the other hand, we can compute the function $f(x,\lambda)$ by using the *Fokker-Plank equation*

$$\frac{\partial p}{\partial \lambda} = Tr\left[\frac{\partial (f_\lambda . p)}{\partial x}\right] = -\mathrm{div}(f_\lambda . p) + \frac{1}{2}\mathrm{div}\left[Q(x)\frac{\partial p}{\partial x}\right] \tag{27}$$

where $\mathrm{div}(\cdot)$ is the divergence function. By equating the right-hand sides of equations (26) and (27) we obtain the following partial differential equation (PDE) governing the flow of the particles:

$$p.\log l = -\mathrm{div}(f_\lambda . p) + \frac{1}{2}\mathrm{div}\left[Q(x)\frac{\partial p}{\partial x}\right].$$

Using the *product rule* for the scalar valued function p and the vector field f_λ we expand $\mathrm{div}(f_\lambda . p)$ to get

$$\mathrm{div}(f_\lambda . p) = \mathrm{div}(f_\lambda).p + \frac{\partial p}{\partial x} f_\lambda\,.$$

Now assuming density p is nowhere vanishing, and dividing by p we obtain the first-order PDE, as given by (8).

References

[1] F. Daum and J. Huang, "Particle flow for nonlinear filters with log-homotopy," in *Proceedings of SPIE Signal Processing, Sensor Fusion, and Target Recognition XVIII*, p. 733603, 2009.

[2] F. Daum and J. Huang, "Particle flow with nonzero diffusion for nonlinear filters," in *Proceedings of SPIE: Signal processing, sensor fusion and target tracking XXII*, p. 87450P, 2013.

[3] F. Daum and J. Huang, "Particle flow for nonlinear filters, Bayesian decisions and transport," in *16th International Conference on Information Fusion*, pp. 1072 – 1079, 2013.

[4] S. Choi, P. Willett, F. Daum and J. Huang, "Discussion and applications of homotopy filter," in *Proceedings of SPIE, Signal Processing, Sensor Fusion, and Target Recognition XX*, p. 805021, 2011.

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

[6] T. Ding and M. J. Coates, "Implementation of the Daum-Huang exact-flow particle filter," in *Statistical Signal Processing Workshop*, pp. 257 – 260, 2012.

[7] S. Godsill and T. Clapp, "Improvement strategies for particle filters," in *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, pp. 139 – 158, 2001.

[8] F. Daum and F. Haung, "Particle Degeneracy: root cause and solution," in *Proceedings of SPIE 8050, Signal Processing, Sensor Fusion, and Target Recognition XX*, p. 80500W, 2011.

[9] F. Daum, J. Huang, A. Noushin and M. Krichman, "Gradient estimation for particle flow induced by log-homotopy for nonlinear filters," in *Proceedings of SPIE 7336, Signal Processing, Sensor Fusion, and Target Recognition XVIII*, p. 733602, 2009.

[10] H. Durrant-Whyte and T. C. Henderson, "Multisensor data fusion," in *Springer Handbook of Robotics*, Springer, pp. 585 – 610, 2008.

[11] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 39, no. 4, pp. 1333 – 1364, 2003.

[12] N. Moshtagh and M. W. Chan, "Boosting target tracking using particle filter with flow control," in *Proc. SPIE 8744,* p. 87440I, 2013.

[13] N. Moshtagh, P. M. Romberg and M. W. Chan, "Tracking low SNR targets using particle filter with flow control," in *Proc. SPIE 9092*, p. 90920A, 2014.

[14] N. Moshtagh, P. M. Romberg and M. W. Chan, "Multisensor fusion for 3D target tracking using track-before-detect particle filter", in *Proc. SPIE 9474 XXIV*, 947405, 2015.

[15] F. Daum, J. Huang, "Proof that particle flow corresponds to Bayes' rule: necessary and sufficient conditions," in *Proc. SPIE 9474,* p. 94740I, 2015.