MULTISENSOR FUSION FOR TARGET TRACKING USING SEQUENTIAL MONTE CARLO METHODS

Mahesh Vemula and Petar M. Djurić

Department of Electrical and Computer Engineering
Stony Brook University
Stony Brook, NY, 11794-2350
vema, djuric @ece.sunysb.edu

ABSTRACT

In this paper, we consider the problems of centralized and distributed multisensor filtering from a Bayesian perspective. We present sequential Monte Carlo algorithms for obtaining complete posterior distributions from individual sensor measurements and from individual sensor posterior distributions, respectively. In the latter case, the individual posterior distributions are approximated as Gaussian distributions, where the information being communicated by the sensors are the statistics of the distributions. The posterior distributions obtained by a centralized algorithm are computed either by the fusing of the likelihoods or by combining the moments of the individual sensor posterior distributions. The proposed algorithms are applied to two problems of target tracking (a) using bearings only measurements and (b) using multimodal sensor data. For the problems, we provide the root mean square errors, and for problem (a), we compare them with the posterior Cramér-Rao lower bounds.

1. INTRODUCTION

Recent advances in fabrication and MEMS technology allow for the production of low cost and reliable miniature sensors which can be deployed in large numbers to form ubiquitous networks for the purpose of sensing, computing, and communication. These sensors which are used to monitor time-varying physical phenomena are of various modalities including acoustic, image, seismic, thermal, pressure, and light. Some of the important challenges in this field are methods for optimally combining data from disparate sources and these methods are broadly classified as multisensor data fusion algorithms. Thus "multisensor data fusion refers to the acquisition, processing and synergistic combination of information gathered by various knowledge sources and sensors to provide a better understanding of the phenomenon under consideration" [1]. The purpose of multisensor data fusion is to improve the estimate of the state of the time-varying phenomenon and to decrease the uncertainty in its estimation. Multisensor data fusion has found widespread use in military and non-military applications. Some military applications are in automated target recognition, battlefield surveillance, target trajectory estimation, and threat recognition systems [2]. Non-military applications include robotics, image processing, monitoring of traffic, weather and biomedical applications.

This work has been supported under Award CCR-0082607.

An important problem in multisensor data fusion is the multisensor filtering problem. The sensors collect measurements about the time-varying processes, such as temperature, pressure, humidity, or trajectory of a target and combine these data to estimate the states of these dynamic processes. Mathematically, we model the discrete-time system of interest as

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}) + \mathbf{w}_t \\ \mathbf{y}_t^n &= h^n(\mathbf{x}_t) + \mathbf{v}_t^n \end{aligned} \tag{1}$$

where $\mathbf{x}_t$ represents the dynamically evolving time-varying unobservable state, $\mathbf{y}_t^n$ is the sensed information by the n^{th} sensor about the state $\mathbf{x}_t$, $\mathbf{w}_t$ is state noise process, and $\mathbf{v}_t^n$ the measurement noise process. If $\mathbf{x}_{0:t}$ denotes the trajectory of the state from time instant 0 to time t, and $\mathbf{y}_{1:t}^{1:N}$ are the data observed by the sensors $n = 1, 2, \cdots, N$ up to time t, all the information about $\mathbf{x}_{0:t}$ is in the joint posterior distribution

$$p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{1:N}).$$

The aim of the multisensor filtering algorithms is to obtain this distribution or some of their statistics. In situations when the evolving dynamic process is linear, Markovian, Gaussian, and the measurements are linear functions of the state, optimal recursive solutions of the state estimate can be obtained by using the Kalman filter. When either of these conditions of linearity or gaussianity are no longer valid, approximations like local linearization and the extended Kalman Filter are used to obtain the posterior distribution [3][4].

Such approximations, however, often lead to poor estimates, instability, and large uncertainity. To counteract some of these problems in filtering theory, a new class of Monte Carlo algorithms, also known as particle filters, have been introduced. They do not suffer from these problems and treat the non-linearity and non-gaussianity with equal ease. In this paper we address issues of using particle filters for obtaining the joint posterior distribution in centralized and decentralized sensor fusion architectures. In the former, there is a central unit which computes the joint posterior distribution while in the latter, the sensors communicate among themselves and collaborate to obtain the joint posterior distribution. Thus, in a decentralized architecture either a selected few or all the sensors contain the joint posterior distributions.

The main contributions in this paper are as follows:

- Estimation of the joint posterior distribution $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{1:N})$ using the sensor measurements $\mathbf{y}_{1:t}^{1:N}$. We refer to this problem as one of data fusion Type-I (see Fig 1).

- Estimation of the joint posterior distribution $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{1:N})$ from the individual posterior distributions $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{n})$ $(n = 1 \cdots N)$. We call this problem as one of data fusion Type-II (see Fig 2 and 3).

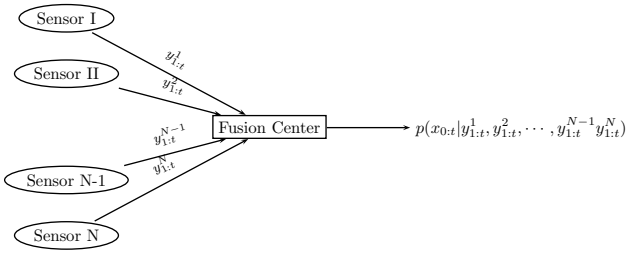

Fig. 1. Centralized sensor fusion architecture I.

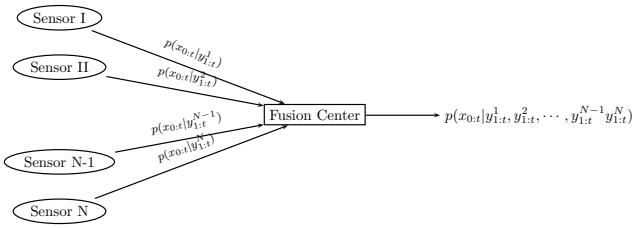

Fig. 2. Centralized sensor fusion architecture II.

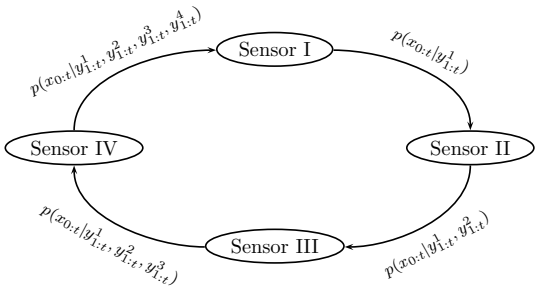

Fig. 3. Distributed sensor fusion architecture.

The motivation for the second architecture arises from scenarios where the transmission of all the data to a central unit is not feasable due to communication traffic congestions. Instead, the sensors perform data processing on board and transmit summaries of the processing periodically and with much smaller frequency than the sampling rate of the measured data.

The organization of the paper is as follows: in Section 2, a very brief review of particle filtering methods is provided. We present solutions to the data fusion problem in Section 3. Finally in Sections 4 and 5 we present some results and simulations for the bearings-only tracking problem and multimodal sensor data and demonstrate the performances of the proposed schemes. With Section 6, we conclude the paper.

2. PARTICLE FILTERING

¿From a Bayesian perspective, all information about the state is contained in the *a posteriori* probability distribution function, $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t})$. In the particle filtering framework, this posterior distribution is approximated by means of a discrete measure with a random support of M sample (particle) trajectories as

$$\hat{p}(\mathbf{x}_{0:t}, \mid \mathbf{y}_{0:t}) = \sum_{m=1}^{M} \delta \left(\mathbf{x}_{0:t} - \mathbf{x}_{0:t}^{(m)} \right) w_t^{(m)}$$

where $\mathbf{x}_{0:t}^{(m)}$ denotes a stream of particles of the unknown state, while $w_t^{(m)}$ are weights associated to the particles, and $\delta(\cdot)$ denotes the Dirac's delta function [5]. The particle filtering algorithm is composed of three main steps: (1) particle generation, (2) weight computations followed by their normalization, and (3) resampling. The particles $\mathbf{x}_t^{(m)}$ are generated from a proposal distribution function $\pi(\cdot)$ as

$$\mathbf{x}_t^{(m)} \sim \pi(\mathbf{x}_t \mid \mathbf{x}_{t-1}^{(m)}, \mathbf{y}_t).$$

Upon reception of the measurement $\mathbf{y}_t$, the weights are updated by

$$\tilde{w}_t^{(m)} = w_{t-1}^{(m)} \frac{p(\mathbf{y}_t \mid \mathbf{x}_{0:t}^{(m)}, \mathbf{y}_{1:t-1}) p(\mathbf{x}_t^{(m)} \mid \mathbf{x}_{t-1}^{(m)})}{\pi(\mathbf{x}_t^{(m)} \mid \mathbf{x}_{t-1}^{(m)}, \mathbf{y}_t)}.$$

In the resampling step, particles with negligible weights are eliminated and those with large weights are replicated.

A variant of the particle filtering methods is the Gaussian particle filter [6]. The central idea of this work is the representation of the posterior and predictive distributions with Gaussian kernel using the generated particles. Its performance has been shown to be better than the Kalman filter and the Extended Kalman filter for non-linear and non-Gaussian state-space models.

3. BAYESIAN DATA FUSION

3.1. Data Fusion Type I

The posterior equation for multisensor data fusion can be derived as

$$p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{1:N}) \quad \propto \quad p(\mathbf{x}_{0:t-1} \mid \mathbf{y}_{1:t-1}^{1:N}) \prod_{n=1}^{N} p(\mathbf{y}_t^n \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \tag{2}$$

assuming conditional independence among the sensors. This recursive expression can be implemented by a straightforward application of the particle filtering methods. The method proceeds as follows:

1. Initialization: Particles are drawn from an initial distribution $\pi(\mathbf{x}_0)$ and the weights are all set to $\frac{1}{M}$.
2. New particle generation: At time instant t, the particles are drawn from a proposal function $\pi(\mathbf{x}_t \mid \mathbf{x}_{t-1}^{(m)}, \mathbf{y}_t^{1:N})$. A typical proposal function is the prior distribution $p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$.
3. Weight computation: Each particle is assigned a weight according to

$$w_t^{(m)} \propto w_{t-1}^{(m)} \prod_{n=1}^{N} p(\mathbf{y}_t^n \mid \mathbf{x}_t^{(m)}). \tag{3}$$

Thus, one can recursively estimate the joint posterior distribution.

3.2. Data Fusion Type II

The objective consists of estimating the joint posterior distribution $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{1:N})$ from the individual posterior distributions, $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{n})$, where $n = 1 \cdots N$. With this data fusion type, an important issue is the transformation of the sample (particle) based non-parametric distribution representations into parametric representations. In this paper we consider the transformation of particle distributions into parametric forms of the Gaussian kind. Other representations including ones based on Gaussians mixtures will be dealt elsewhere. When one has prior knowledge that the posterior distributions are unimodal then such approximations by a single Gaussian distribution are sufficiently accurate. However, when the posterior distributions are not unimodal, the Gaussian approximations usually yield poor results. In the remainder of the paper, we assume that the posterior distributions are unimodal.

It can be shown that

$$\begin{aligned} & p(\mathbf{x}_{0:t} \mid \mathbf{y}_{1:t}^{1:N}) \\ & \propto p(\mathbf{x}_{0:t-1} \mid \mathbf{y}_{1:t-1}^{1:N}) \prod_{n=1}^{N} \frac{p(\mathbf{x}_t \mid \mathbf{y}_{1:t}^{n})}{p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}^{n})} \, p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \end{aligned} \quad (4)$$

which is the optimal recursive fusion equation. However, only in situations where the state space model is linear and Gaussian, analytically tractable solutions can be obtained. To the authors' knowledge, for non-linear state-space models there are no methods for solving (4). In this paper, we provide a suboptimal solution to (4) using the Gaussian particle filtering methodology. In the Gaussian particle filtering framework, the individual posteriors and predictive distributions are represented by Gaussian kernels, i.e.,

$$\begin{aligned} p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}^{n}) &\simeq \mathcal{N}(\tilde{\boldsymbol{\mu}}_t^n, \tilde{\boldsymbol{\Sigma}}_t^n) \\ p(\mathbf{x}_t \mid \mathbf{y}_{1:t}^{n}) &\simeq \mathcal{N}(\hat{\boldsymbol{\mu}}_t^n, \hat{\boldsymbol{\Sigma}}_t^n) \\ p(\mathbf{x}_t \mid \mathbf{y}_{1:t}^{1:N}) &\simeq \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t). \end{aligned}$$

Therefore, from (4) we have

$$p(\mathbf{x}_t \mid \mathbf{y}_{1:t}^{1:N}) \propto \frac{\mathcal{N}(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t)}{\mathcal{N}(\tilde{\boldsymbol{\mu}}_t, \tilde{\boldsymbol{\Sigma}}_t)} \, p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \, p(\mathbf{x}_{0:t-1} \mid \mathbf{y}_{1:t-1}^{1:N})$$

with

$$\begin{aligned} \hat{\boldsymbol{\Sigma}}_t^{-1} &= \hat{\boldsymbol{\Sigma}}_t^{1^{-1}} + \hat{\boldsymbol{\Sigma}}_t^{2^{-1}} + \cdots + \hat{\boldsymbol{\Sigma}}_t^{N^{-1}} \\ \hat{\boldsymbol{\mu}}_t &= \hat{\boldsymbol{\Sigma}}_t \left(\hat{\boldsymbol{\Sigma}}_t^{1^{-1}} \hat{\boldsymbol{\mu}}_t^1 + \hat{\boldsymbol{\Sigma}}_t^{2^{-1}} \hat{\boldsymbol{\mu}}_t^2 + \cdots + \hat{\boldsymbol{\Sigma}}_t^{N^{-1}} \hat{\boldsymbol{\mu}}_t^N \right) \\ \tilde{\boldsymbol{\Sigma}}_t^{-1} &= \tilde{\boldsymbol{\Sigma}}_t^{1^{-1}} + \tilde{\boldsymbol{\Sigma}}_t^{2^{-1}} + \cdots + \tilde{\boldsymbol{\Sigma}}_t^{N^{-1}} \\ \tilde{\boldsymbol{\mu}}_t &= \tilde{\boldsymbol{\Sigma}}_t \left(\tilde{\boldsymbol{\Sigma}}_t^{1^{-1}} \tilde{\boldsymbol{\mu}}_t^1 + \tilde{\boldsymbol{\Sigma}}_t^{2^{-1}} \tilde{\boldsymbol{\mu}}_t^2 + \cdots + \tilde{\boldsymbol{\Sigma}}_t^{N^{-1}} \tilde{\boldsymbol{\mu}}_t^N \right). \end{aligned} \quad (5)$$

At each sensor node, a Gaussian particle filter which approximates the individual posterior distributions with a Gaussian kernel is implemented. The sensor transmits the moments to the fusion center. We approximate the joint posterior distribution with a Gaussian distribution function at the fusion center by implementing another Gaussian particle filter. The outline of the algorithm is as follows:

1. Initialization:
   $\mathbf{x}_0^{(m)} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$
2. Collection of posterior moments:
   At time t, the moments of the posterior distributions from all the sensors are collected. The n^{th} sensor transmits the moments of the distribution functions $p(\mathbf{x}_t \mid \mathbf{y}_{1:t}^n)$ and $p(\mathbf{x}_{t+1} \mid \mathbf{y}_{1:t}^n)$. The fusion center computes $\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t, \tilde{\boldsymbol{\mu}}_t, \tilde{\boldsymbol{\Sigma}}_t$ according to (5).
3. Particle generation:
   (a) Draw samples from $\mathcal{N}(\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1})$ and denote them as $\mathbf{x}_{t-1}^{(m)}$.
   (b) Sample from $p(\mathbf{x}_t \mid \mathbf{x}_{t-1} = \mathbf{x}_{t-1}^{(m)})$ and denote these samples as $\mathbf{x}_t^{(m)}$.
4. Weight calculation: $\tilde{w}_t^{(m)} \propto \frac{\mathcal{N}(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t)}{\mathcal{N}(\tilde{\boldsymbol{\mu}}_t, \tilde{\boldsymbol{\Sigma}}_t)}$
5. Weight normalization: $w_t^{(m)} = \frac{\tilde{w}_t^{(m)}}{\sum_{m=1}^{M} \tilde{w}_t^{(m)}}$
6. Estimation of the mean and covariance matrix:
   $$\begin{aligned} \boldsymbol{\mu}_t &= \sum_{m=1}^{M} w_t^{(m)} \mathbf{x}_t^{(m)} \\ \boldsymbol{\Sigma}_t &= \sum_{m=1}^{M} w_t^{(m)} (\mathbf{x}_t^{(m)} - \boldsymbol{\mu}_t)(\mathbf{x}_t^{(m)} - \boldsymbol{\mu}_t)^\top \end{aligned}$$

This algorithm is for the centralized fusion architectures II. However, if the sensors can communicate with its neighbors in a round robin manner, the algorithm with very minor modifications is also suitable for distributed fusion architectures as in Figure 3.

4. THE BEARINGS-ONLY TRACKING PROBLEM.

We applied the proposed ideas to solving the bearings-only tracking problem, where the target trajectory is estimated using only angle measurements and some prior information of the dynamics of the target. In this situation, the prior information we have about the target is that its velocity is subject to an unknown acceleration (modeled as process noise). Therefore the target trajectory was given by the following equation

$$\mathbf{x}_t = \mathbf{G}_x \mathbf{x}_{t-1} + \boldsymbol{\Gamma}_u \mathbf{u}_t \quad (6)$$

where $\mathbf{x}_t = [x_t, \, y_t, \, \dot{x_t}, \, \dot{y_t}]^\top \in \mathbb{R}^4$ indicates the position and the velocity of the target,[1] $\mathbf{G}_x$ and $\boldsymbol{\Gamma}_u$ are known transition matrices given by

$$\mathbf{G_x} = \begin{pmatrix} 1 & 0 & Ts & 0 \\ 0 & 1 & 0 & Ts \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ and } \boldsymbol{\Gamma}_u = \begin{pmatrix} \frac{Ts^2}{2} & 0 \\ 0 & \frac{Ts^2}{2} \\ Ts & 0 \\ 0 & Ts \end{pmatrix}$$

and $\mathbf{u}_t$ is a Gaussian noise process with zero mean and covariance matrix $\mathbf{C}_u$ that accounts for the acceleration of the target. The sensor sampling unit collects measurements z_t which were modeled as

$$z_t^n = \arctan\left(\frac{y_t - l_y^n}{x_t - l_x^n} \right) + w_t^n \quad (7)$$

[1] Note that the observations in this section are denoted by z_t. Here, the symbol y_t is an element of the state vector.

where $\{l_x^n, l_y^n\}$ are the coordinates of the sensor [7], and w_t^n is zero mean white Gaussian noise with variance σ_w^2. We applied the proposed methods to track the target in a multi-sensor scenario.

We considered three sensors placed randomly in the tracking field. At the time of initialization, particles were drawn from a Gaussian distribution function with a known mean $\boldsymbol{\mu}_0$ and covariance matrix $\boldsymbol{\Sigma}_0$

$$\boldsymbol{\mu}_0 = \begin{bmatrix} 1 \\ 2 \\ 0.001 \\ -0.055 \end{bmatrix} \text{ and } \quad \boldsymbol{\Sigma}_0 = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.005 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}. \tag{8}$$

Other parameters used in the simulations were the state covariance matrix $\mathbf{C}_u = q\ \mathcal{I}_2$, $\sqrt{q} = 0.001$, $\sigma_w = 0.1$, and the sampling period $T_s = 1$. We considered the simulation of the following three algorithms:

- The fusion center has complete knowledge of the sensors' measurements and the estimation is performed based on the algorithm explained in Section 3.1 using standard particle filtering (SPF).
- The fusion center has complete knowledge of the sensors' measurements and the estimation is performed based on the algorithm explained in Section 3.1 using Gaussian particle filtering (GPF).
- The fusion center has knowledge of the moments of the individual posterior distributions of the sensors and the sequential estimation is performed based on the algorithm explained in Section 3.2 using Gaussian particle filtering (denoted as Moment GPF-MGPF).

Figure 4, shows the mean square error obtained with these three methods. The posterior Cramér-Rao lower bounds (PCRBs) are also plotted and provide a benchmark in analyzing the performance of the algorithms. It can be seen that the performances of the proposed SPF and GPF algorithms in terms of the mean square error are very similar and that they approach the PCRBs. The performance of the MGPF is worse than the other two algorithms.

5. TARGET TRACKING USING MULTIMODAL SENSORS.

In this second experiment, we implemented the proposed ideas for target tarcking using groups of multimodal sensors. Each group consisted of a set of sensors, with each equipped with the ability to sense the relative angle between the sensor and the target and the strength of the signal emitted by the target. These groups then use this sensor data to track the target. The statistics of the filtering distribution, namely the first and second moments, were transmitted to the fusion center. The dynamics of the target are as given by Equation (6). The angle measurement equation is given by (7) and the measurement equation of the received signal strength is given by the following equation:

$$s_t^n = \Psi_0 - 10\alpha \log 10(|\mathbf{l}^n - \mathbf{x}_t|) + v_t^n \tag{9}$$

where s_t^n is the measured signal strength, Ψ_0 is a known constant, α is a path-loss coefficient, v_t^n represents zero mean white Gaussian noise with variance σ_v^2, and $\mathbf{l}^n$ is the location of the n^{th} sensor, i.e., $\mathbf{l}^n = [l_x^n\ l_y^n]^\top$.

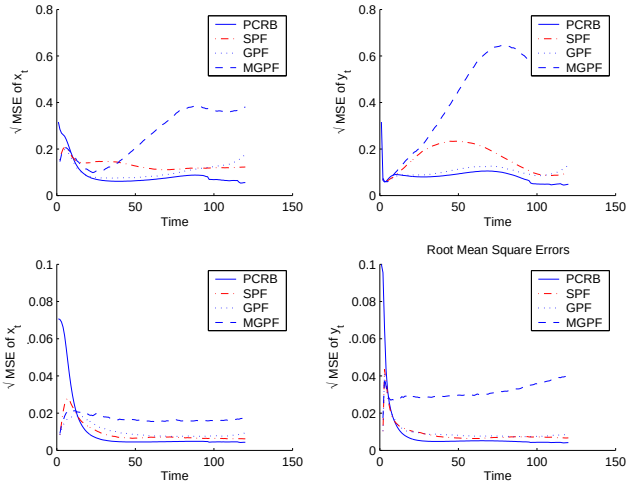

Fig. 4. Root Mean Square Errors for the bearings-only tracking problem. The applied filters are the standard particle filter (SPF), the Gaussian particle filter (GPF), both based on the algorithm from Section 3.1, and the moment-based Gaussian particle filter (MGPF) that uses the algorithm from Section 3.2. The solid lines are the posterior Cramér-Rao bounds (PCRBs).

In our simulations we considered three groups of sensors with each group consisting of three sensors and deployed randomly in the sensor field. This represents a scenario wherein the sensor network is divided into smaller groups and each group individually carries out the tasks of tracking. The groups may then transmit the statistics of the filtered density to a fusion center or collaboratively process it with its neighbors in a round-robin manner. The groups of sensors are denoted by group 1, group 2, and group 3. Each group implements a particle filter for the purpose of individually tracking the target for which the initial particles were drawn from a Gaussian distribution with parameters

$$\boldsymbol{\mu}_0 = \begin{bmatrix} 2002 \\ 2002.5 \\ 5 \\ 5 \end{bmatrix} \text{ and } \quad \boldsymbol{\Sigma}_0 = \begin{bmatrix} 25 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}. \tag{10}$$

The other parameters used in the simulations were $\Psi_0 = -40$, $\alpha = 3.5$, $T_s = 1$s, and $\sigma_v = 3dB$.

In Figure 5, the mean square errors (MSE's) in estimating the location and the velocity by individual sensor groups and the MSE obtained by the proposed fusion algorithm are plotted. It can be seen that the MSE after fusion is less than the individual MSE's. In Figure 6 we plot a target trajectory and the individual sensor-group and group-fusion estimates while in Figure 7 we plot the trace of the covariance matrix of the filtered distribution for this scenario. The trace of the covariance matrix may be regarded as a measure of the uncertainty in estimating unknown parameters, and thus it can be observed that after fusion the total uncertainty in estimating the targets trajectory decreases.

6. CONCLUSIONS

In this paper the problem of fusing information from sensors has been addressed. In general, two types of information are consid-

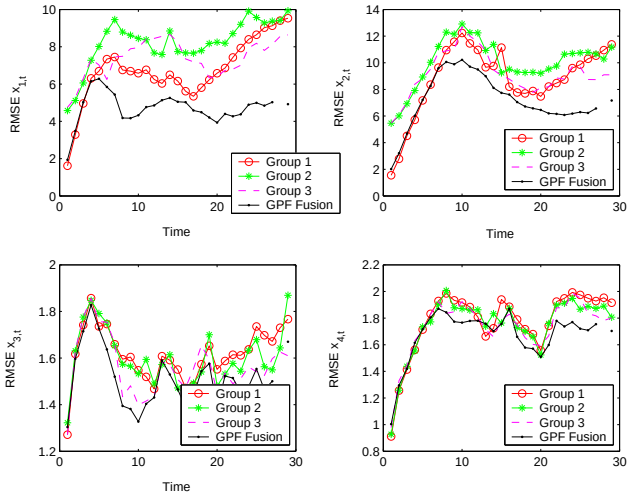

Fig. 5. Root mean square errors.

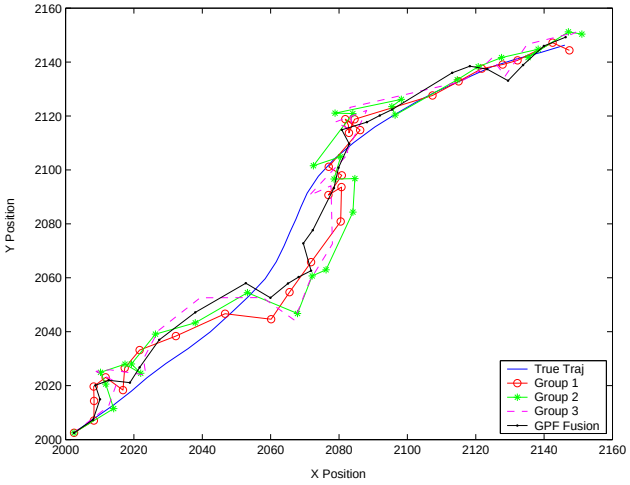

Fig. 6. Target trajectory and its estimates by individual groups . The individual sensor estimates are also fused.

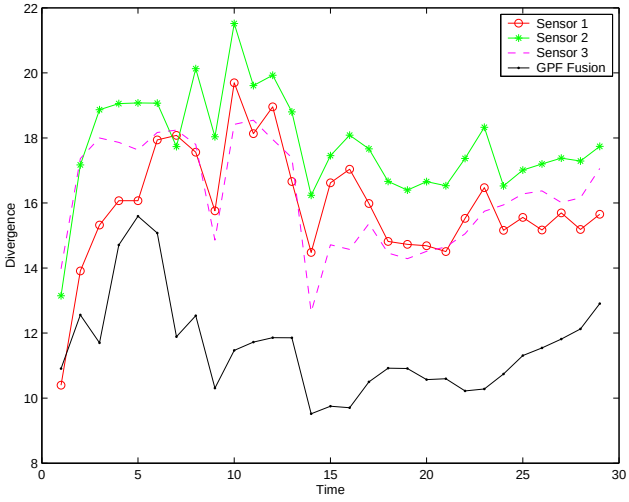

Fig. 7. Trace of the covariance matrix of the filtering density.

ered. One is information provided by raw data and the other by summaries of the estimated unknowns. The former is used for networks that allow for high communication throughput and do not require sophisticated signal processing by the sensors. The latter are used for scenarios where local signal processing takes place at the sensor. The obtained estimates expressed by Gaussian posteriors are sent to a fusion center, and there they are combined to obtain final estimates of the tracked unknowns. The two schemes are tested and compared on the bearings-only tracking problem and the problem of target tracking with multimodal sensor data.

7. REFERENCES

[1] P. K. Varshney, "Scanning the special issue on data fusion," in *Proceedings of the IEEE*, January 1997, vol. 85, pp. 3–5.

[2] D. Hall and J. Llinas, "Introduction to multisensor data fusion," in *Proceedings of the IEEE*, January 1997, vol. 85, pp. 6–23.

[3] C. J. Harris and J. B. Gao, "Some remarks on Kalman filters for the multisensor fusion," *Information Fusion*, vol. 3, no. 191-201, Sep 2002.

[4] K. C. Chang, C. Y. Chong, and Y. Bar-Shalom, *Distributed Estimation in Distributed Sensor Networks in Stochastic Large Scale Systems*, M. Dekker, 1992.

[5] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, 2001.

[6] J. H. Kotecha and P. M. Djurić, "Gaussian particle filtering," *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2592–2601, October 2003.

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