Acoustic Multi Target Tracking using Direction-of-Arrival Batches

Volkan Cevher, *Member, IEEE,* Rajbabu Velmurugan, *Student Member, IEEE*, James H. McClellan, *Fellow, IEEE*

Abstract

In this paper, we propose a particle filter acoustic direction-of-arrival (DOA) tracker to track multiple maneuvering targets using a state space approach. The particle filter determines its state vector using a batch of DOA estimates. The filter likelihood treats the observations as an image, using template models derived from the state update equation, and also incorporates the possibility of missing data as well as spurious DOA observations. The particle filter handles multiple targets, using a partitioned state-vector approach. The particle filter solution is compared with three other methods: the extended Kalman filter, Laplacian filter, and another particle filter that uses the acoustic microphone outputs directly. We discuss the advantages and disadvantages of these methods for our problem. In addition, we also demonstrate an autonomous system for multiple target DOA tracking with automatic target initialization and deletion. The initialization system uses a *track-before-detect* approach and employs the matching pursuit idea to initialize multiple targets. Computer simulations are presented to show the performances of the algorithms.

I. Introduction

A challenging signal processing problem arises when one demands automated tracking of the direction-of-arrival (DOA) angles of multiple targets using an acoustic array in the presence of noise or interferers [1]–[5]. In the literature, DOA tracking problems are usually formulated using state-space models [6]–[8], with an observation equation that relates the state vector (i.e., target DOA's and possibly motion states) to the acoustic microphone outputs, and a state equation that constrains the dynamic nature of the state vector. The performance of the tracking algorithms

V. Cevher (volkan@umiacs.umd.edu) is with the Center for Automation Research, University of Maryland, College Park, MD 20742. R. Velmurugan and J. H. McClellan ({rajbabu, jim.mcclellan}@ece.gatech.edu) are with the Center for Signal and Image Processing, School of ECE, Georgia Institute of Technology, Atlanta, GA 30332.

Prepared through collaborative participation in the Advanced Sensors Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-02-0008.

using state-spaces relies heavily on how accurate the models represent the observed natural phenomena. Hence, in most cases, it is important to use nonlinear and non-Gaussian state-space models despite their computational complexity [9].

The presence of multiple targets increases the tracking complexity because a mechanism for data association is needed, in effect, to sort the received data for each target. In the literature, the association problem is handled in several different ways: (i) probabilistic data association methods estimate the states by summing over all the association hypothesis weighted by the probabilities obtained by the likelihood [10]–[13], (ii) smoothness assumptions on the target (motion) states allow a natural ordering of the data [14], (iii) computationally costly ML/EM methods use the likelihood function to search for a global maximum, or (iv) nearest neighbor methods provide easy heuristics to perform measurement updates. Most of these methods use the mean and covariance approximation of the sufficient statistics for the state, which may be estimated with a Kalman filter; however, for nonlinear state-spaces with general noise assumptions, Monté-Carlo methods should be used to adequately capture the dynamic, possibly multi-modal, statistics.

In a particle filter, where the observations arrive in sequence, the state probability density function is represented by discrete state samples (particles) distributed according to the underlying distribution (as explained by the state-space) either directly or by proper weighting [8], [9]. Hence, the filter can approximate any statistics of the distribution arbitrarily accurately by increasing the number of particles with proven convergence results. In the particle filtering framework, the data association problem is undertaken implicitly by the state-space model interaction. However, the particle filter suffers from the curse of dimensionality problem, as the number of targets increases [15]. To increase the efficiency of the algorithm, various methods are proposed, such as the partitioning approach [2], [16], or other Bayesian approaches [17].

In this paper, we present a particle filter algorithm to track the DOA's of multiple maneuvering targets, using an acoustic node that contains an array of microphones with known positions. Each particle in the filter is created by concatenating partitions, i.e., the state vector for each target. For example, the partition of the kth target has a state vector that consists of the DOA $\theta_k(t)$, the heading direction $\phi_k(t)$, and the logarithm of velocity over range $Q_k(t) = \log\left(v_k/r_k(t)\right)$ of the kth target. The total number of targets (or partitions) K is determined by a mode hungry Metropolis-Hastings block. Hence, given K targets, a particle has K partitions where each target, and hence each partition, is assumed to be independent. Target motions are modeled as locally linear, i.e., each target has a constant velocity within an estimation period of duration T.

The particle filter uses multiple DOA's to determine the state vector, based on an image template matching idea. We denote the collection of M DOA's a batch, where M is the batch size. In our problem, a DOA image is first formed when a batch of DOA observations are

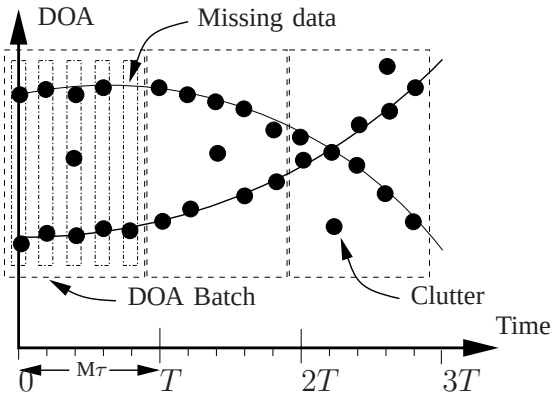


Fig. 1: Observation model uses a batch of DOA measurements. Note that the DOA measurements are not necessarily ordered. However, the image based observation approach provides a natural ordering when targets are being tracked by the particle filter.

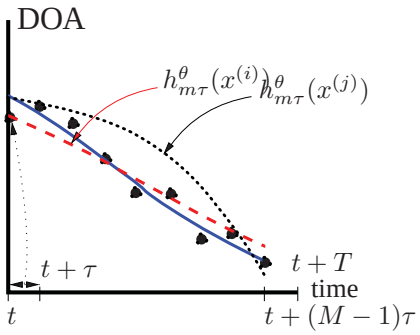


Fig. 2: Template matching idea is illustrated. The solid line represents the true DOA track. Black dots represent the noisy DOA estimates. The dashed line and the dotted line represent the DOA tracks for the two proposed particles i and j. These tracks are calculated using the state update function h. Visually, the ith particle is a better match than the jth particle; hence, its likelihood is higher.

received from a beamformer that processes the received acoustic data at M τ-second intervals (Fig. 1). Then, image templates for target tracks are created using the state update function and the target partition state vectors (Fig. 2). By determining the best matching template (e.g., most probable target track), the target state-vectors are estimated. Because the observations are treated as an image, the data association and DOA ordering problems are naturally alleviated. Moreover, by assuming that the DOA observations are approximately normally distributed around the true target DOA tracks, with constant DOA miss-probability and clutter density, a robust particle filter tracker is formulated.

The particle filter importance function proposes particles for each target independently to increase the efficiency of the algorithm. To derive the proposal function for each target, we

use Laplace's method to approximate the posterior of the corresponding partition by a Gaussian around its mode [8], [18], [19]. We calculate the partition modes using a robust Newton-Raphson search method that imposes smoothness constraints on the target motion. We also present a slower but more robust method for determining the partition posteriors, based on the mode hungry Metropolis-Hastings algorithm [20]. Details of the partition posteriors are given in Sect. IV.

For the initialization of the particle filter, we describe a mode hungry Metropolis-Hastings (MHMH) sampling algorithm. For comparison purposes, we also present an approximate Bayesian filter based on the particle filter's proposal function, and an extended Kalman filter (EKF) [10]. The EKF is derived using a sliding window implementation, because the Kalman filter initial data association becomes prohibitive, while handling the complex observation model based on DOA batches in the case of multiple targets, using the sufficient statistics. In addition, we qualitatively compare the computational complexity of the filters: the particle filter solution outperforms the approximate Bayesian solution with no significant increase in the computational requirements.

The paper is organized as follows. Section II explains the mechanics of the automated tracking system for multiple target DOA tracking. Sections III, IV, and V elaborate on the individual elements of the tracking system. The computational complexity of the algorithms is given in Sect. VI. Alternative tracking approaches are discussed in Sect. VII. Computer simulations are shown in Sect. VIII to demonstrate the algorithm performances.

II. Tracker Design

The tracker mechanics in this paper is constructed such that the tracker (i) compensates DOA estimation biases due to rapid target motion [3], (ii) results in higher resolution DOA estimates than just beamforming [2]–[4], (iii) is robust against changes in target signal characteristics, and (iv) automatically determines the number of targets. The tracker system consists of three blocks as illustrated in Fig. 3. Details of the individual blocks are given in the following sections. Below, we discuss the technicalities that lead to this design.

Note that the main objective of our tracker is to report multiple target DOA's at some period T, after observing the acoustic data at the node microphones. Quite often, target DOA's can change more than a few degrees during an estimation period, e.g., due to rapid target motion. Hence, if we were to just use conventional snapshot DOA estimation methods (e.g., MUSIC, MVDR, etc.) for tracking the targets, the bearing estimates become biased, because the received data is not stationary [21]. This is intuitive, because these methods estimate an average of the target angular spread during their estimation periods [3].

In general, locally linear motion models eliminate this bias by simultaneously estimating the bearing and the target motion parameters for the estimation period of T. Conceptually, this

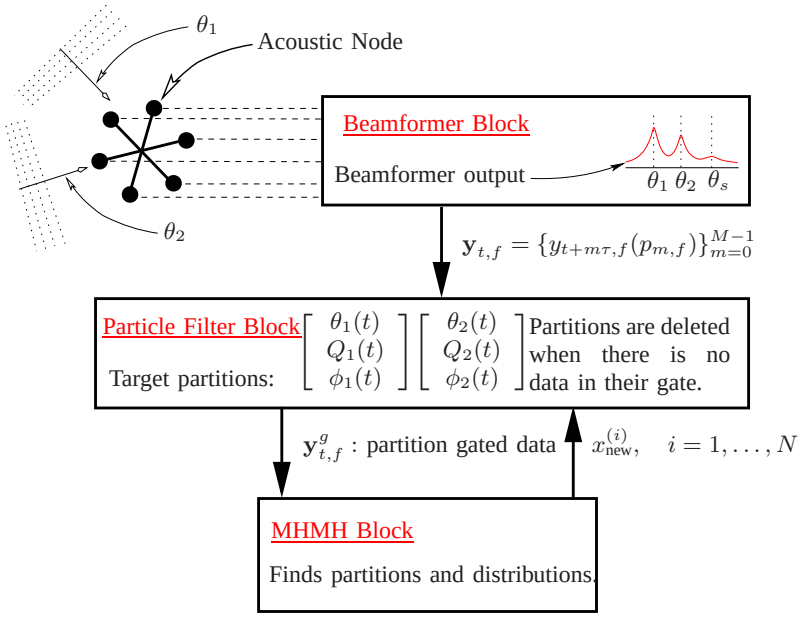


Fig. 3: Tracker mechanics demonstrated. The beamformer block monitors the received acoustic signals to adapt to their frequency characteristics for better bearing estimation. It outputs a batch of DOA's that form a sufficient statistics for the particle filter block. The particle filter estimates the tracking posterior and can make various inferences (i.e., mean and mode estimates). It acquires the new partition information from the mode hungry Metropolis Hastings (MHMH) block. It can also delete a partition, if there is no data in the respective partition gate (explained in the text). The MHMH block determines the new partitions and their distributions from the residual DOA's coming from the particle filter (i.e, the gating operation).

is equivalent to aligning the received acoustic data with the motion parameters so that the data becomes stationary for bearing estimation purposes. But, this alignment process relies heavily on the observation model and is computationally costly because of the high volume of the acoustic data used for estimation. In this paper, we propose to use a beamformer block to buffer the variability in the observed acoustic signals and to create a compressed set of invariant statistics for our particle filter block.

Therefore, the beamformer block (Fig. 3) processes the acoustic data at shorter time intervals of $\tau = T/M$ (e.g., $M = 10$), where the targets are assumed to be relatively stationary. This reduces the number of acoustic data samples available for processing, resulting in a sequence of noisier DOA estimates. However, these noisier DOA estimates can be smoothed in the particle filter block, because a motion structure is imposed on the batch of DOA's. Moreover, for the state vector defined in the introduction, a batch of DOA's when $M > 3$ is sufficient for observability of the state [3], [7]. Since the filter is built on the compressed statistics which is also sufficient to observe the state vector, it achieves a significant reduction in computation.

When the received signal characteristics change, the particle filter tracker formulation is not affected because the beamformer block can absorb the variabilities in the acoustic signals. In the literature, various trackers use similar state-space formulations as in this paper [2]–[4]. The trackers presented in [2], [3] directly employ the classical narrow-band observation model, where targets exhibit constant narrow-band frequency characteristics [21], [22]. The tracker in [4] tries to adapt to varying time-frequency characteristics of the target signals, assuming that the varying frequencies are narrow-band. The tracker in [5] also incorporates an amplitude model for the signals. Because their probability density equations explicitly use an observation equation, these trackers are hardwired to their observation model. Hence, a complete re-work of the filter equations would be required to track targets with wideband signal characteristics (e.g., [5]- [23]).

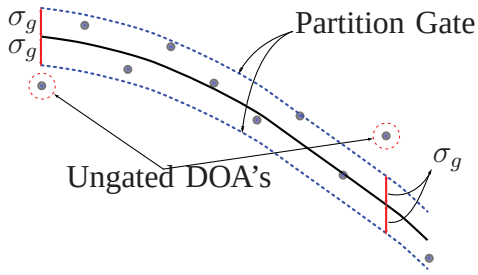


Fig. 4: Gating operation is illustrated. Solid line is the true DOA track. The DOA observations are shown with dots. Given a parition gate size σ_g, the DOA's are gated out if they are more than σ_g away from the solid line.

The third block in Fig. 3 addresses a fundamental issue for trackers: initialization. It is an accelerated Metropolis-Hastings algorithm modified specifically for unimodal distributions. It takes the ungated DOA's from the particle filter as inputs (Fig. 4), or if there are no partitions in the particle filter at initialization, it takes the data from the beamformer block directly. It generates a particle distribution for each potential new target *one at a time*. That is, it converges on the strongest mode in the, possibly, multi-modal target posterior and sifts out the corresponding DOA data. It then iterates to find other modes until stopping criteria are met. This block creates new partitions for the particle filter, which can also delete its own partitions when conditions described in Sect. V are met. Conceptually, this initialization idea is equivalent to the *track-before-detect* approach used in the radar community [24], [25].

III. Beamformer Block

Beamforming is the name given to a wide variety of array processing algorithms that focus an array's signal processing capabilities in a particular direction [21]. Beamformers use the collected acoustic data to determine target DOA's and are called narrow-band beamformers if they use the

classical narrow-band array observation model [21], [22]. Beamformers are wideband if they are designed for target signals with broadband frequency characteristics [26]; others are designed for signals with time-varying narrow-band frequency characteristics [27], [28].

The beamformer block chooses a beamfomer for processing the acoustic data depending on the local characteristics of the acoustic signals. That is, given the observed acoustic signal and its time-frequency distribution, we choose an optimal beamformer to calculate target DOA's. For example, we can choose multiple beamformers if the received acoustic signal shows both narrow-band and wideband characteristics. The output of the beamformer $\mathbf{y}_{t,f} = \{y_{t+m\tau,f}(p)\}_{m=0}^{M-1}$ is a DOA data cube containing the $P_{m,f}$-highest DOA peaks of the beamformer pattern (Fig. 1 and Fig. 5). In general, the number of DOA peaks $P_{m,f}$ at batch index m and each frequency index f should be greater than or equal to the number of targets K so that we can also detect new targets while tracking the existing targets. In the simulations section, we fix $P_{m,f} = P$ but the derivations below explicitly show the dependance on m and f. Note that the input of the particle filter has the same structure regardless of the target signal characteristics.

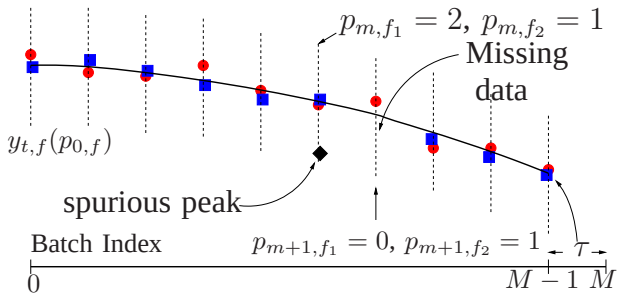


Fig. 5: The circles and squares denote the DOA estimates at two different frequencies f_1 and f_2, calculated using the acoustic data received during a period of length τ. In this example, the maximum number of beamformer peaks P is 2. Given the observations $\mathbf{y}_{t,f}$, the objective of the particle filter is to determine the state $x_k(t)$ which completely parameterizes the solid curve.

Finally, the choice of the parameter τ for beamforming is determined by various physical constraints, including (i) target frequency spread, (ii) target speed, and (iii) a target's affinity to maneuver. For reasonable beamforming, at least two cycles of the narrow-band target signals must be observed. This stipulates that $\tau > 2/F_{min}$, where F_{min} is the minimum beamforming frequency for the target. Moreover, to keep the worst case beamforming bias[1] bounded for each DOA by an angle threshold denoted by D, we approximately have $\tau < \frac{2D}{\exp\{Q\}}$, where $\exp\{Q\}$ is

[1]The bias is calculated by taking the angular average of the target track. Hence, this bias also depends on the heading direction. The worst case bias happens when the target heading and DOA sum up to π. Moreover, it is also possible to analytically find an expected bias by assuming uniform heading direction, using a similar analysis done in [3].

the target velocity over range ratio. Lastly, the target motion should satisfy the constant velocity assumption during the output period T. For slow moving ground targets, $T = 1$s is a reasonable choice. Note that at least three DOA estimates are necessary to determine the state vector. To improve the robustness of the tracker, we use $M > 5$ to decrease the probability that the state is not observable due to missing DOA's. Hence, the parameter τ is bounded by the following

$$\frac{2}{F_{min}} < \tau < \min\left\{\frac{2D}{\exp\{Q\}}, \frac{T}{M}\right\}. \tag{1}$$

IV. Particle Filter

In this section, the details of the particle filter block in Fig. 3 are discussed. In Sect. VII, we provide alternative algorithms to replace this block based on the Kalman filter and Laplace's method and compare their performance in terms of computational requirements.

A. State Equation

The particle filter state vector $\mathbf{x}_t = \left[\ x_1^T(t),\ x_2^T(t),\ \cdots\ , x_K^T(t)\ \right]^T$ consists of the concatenation of partitions $x_k(t)$ for each target, indexed by k, where K is the number of targets at time t.[2] Each partition has the corresponding target motion parameters $x_k(t) \triangleq [\ \theta_k(t)\ ,\ Q_k(t)\ ,\ \phi_k(t)\ ]^T$, as defined in the introduction. The angle parameters $\theta_k(t)$ and $\phi_k(t)$ are measured counterclockwise with respect to the x-axis.

The state update equation can be derived from the geometry imposed by the locally constant velocity model. The resulting state update equation is nonlinear:

$$x_k(t+T) = h_T\left(x_k(t)\right) + u_k(t), \tag{2}$$

where $u_k(t) \sim \mathcal{N}\left(0, \Sigma_u\right)$ with $\Sigma_u = \mathrm{diag}\{\sigma^2_{\theta,k}, \sigma^2_{Q,k}, \sigma^2_{\phi,k}\}$ and

$$h_T(x_k(t)) = \begin{bmatrix} \tan^{-1}\left\{\frac{\sin\theta_k(t) + T\exp Q_k(t)\sin\phi_k(t)}{\cos\theta_k(t) + T\exp Q_k(t)\cos\phi_k(t)}\right\} \\ Q_k(t) - \frac{1}{2}\log\left\{1 + 2T\exp Q_k(t)\cos(\theta_k(t) - \phi_k(t)) + T^2\exp(2Q_k(t))\right\} \\ \phi_k(t) \end{bmatrix}. \tag{3}$$

The analytical derivations of (3) can be found in [3], [4], and reference [4] also discusses state update equations based on a constant acceleration assumption.

[2]Explicit time dependence is not shown in the formulations. The parameter K is determined by the MHMH block (Sect. V).

B. Observation Equation

The observations $\mathbf{y}_{t,f} = \{y_{t+m\tau,f}(p)\}_{m=0}^{M-1}$ consist of all the batch DOA estimates from the beamformer block indexed by m. Hence, the acoustic data of length T is segmented into M segments of length τ. The batch of DOA's, $\mathbf{y}_{t,f}$, is assumed to form an approximately normally distributed cloud around the true target DOA tracks (Fig. 1). In addition, only one DOA is present for each target at each f or the target is missed. Multiple DOA measurements imply the presence of clutter or other targets. We also assume that there is a constant detection probability for each target denoted by κ^f, where dependence on f is allowed. If the targets are also simultaneously identified, an additional partition dependency is added, i.e., κ_k^f.[3]

The particle filter observation model also includes a clutter model because beamformers can produce spurious DOA peaks as output (e.g., the sidelobes in the power vs. angle patterns) [21]. To derive the clutter model, we assume that the spurious DOA peaks are random with uniform *spatial* distribution on the angle space, and are temporally as well as spatially independent. In this case, the probability distribution for the number of spurious peaks is best approximated by the Poisson distribution with a spatial density [10], [29]. Moreover, the probability density function (pdf) of the spurious peaks is the uniform distribution on $[0, 2\pi)$. However, since the number of peaks in the beamformer output can be user defined (P), and since the beamformer power vs. angle pattern has smoothness properties, we use the following pdf for the spurious peaks:

$$p(\theta|\theta \text{ is spurious}) = \frac{\gamma}{2\pi}, \tag{4}$$

where $\gamma \gg 1$ is a constant that depends on the maximum number of beamformer peaks P, the beamformer itself (i.e., the smoothness of the beamformer's steered response), and the number of targets K. Equation (4) implies that the natural space of the clutter is reduced by a factor of γ because of the characteristics of our specific system.

We now derive the data-likelihood function using the joint probabilistic data association arguments found in [10]. Similar arguments for active contour tracking relevant to this paper are found in [30]. Consider the output of one batch period $\mathbf{y}_{m,f} = y_{t+m\tau,f}(p)$, where $p = 0, 1, \ldots, P_{m,f}$ for each f and m. The DOA's $\mathbf{y}_{m,f}$ may belong to none, or some combination, or all of the targets in the particle filter partitions. Hence, we first define a notation to represent possible combinations between the data and the particle filter partitions to effectively derive the observation density.

[3]Recognition/identification may have an impact on choice of the acoustic frequencies as discussed earlier. Hence, some targets may have a lower detection probability at the recognized target frequencies. The partition dependence of the detection probability takes care of this issue.

Define a set $\mathcal{I}_n$ that consists of n-unordered combination of all K-partitions of the particle filter state vector: $\mathcal{I}_n \in \{{}_K\mathcal{C}_n\}$, where ${}_K\mathcal{C}_n$ is number of ways of picking n-unordered outcomes from K possibilities. Each element of $\mathcal{I}_n$ has n numbers, and there are a total of ${}_K\mathcal{C}_n$ elements. For example, when $K = 3$ and $n = 2$, then $\mathcal{I}_2 = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$, each element referring to subset of the individual partitions of the particle state vector. We refer to the individual elements of this set using the notation $\mathcal{I}_n(j)$, where $j = 1, \ldots, {}_K\mathcal{C}_n$. Hence, $\mathcal{I}_2(2) = \{1, 3\}$. Then, denote ${}_n\mathbf{x}_t(j) \in \{x_i(t)|i \in \mathcal{I}_n(j), x_i(t) \in \mathbf{x}_t\}$ as a single realization from the set $\mathcal{I}_n$. Using the same example, ${}_2\mathbf{x}_t(3) = \left[\ x_2^T(t),\ x_3^T(t)\ \right]^T = \left[\ \hat{x}_1^T(t),\ \hat{x}_2^T(t)\ \right]^T$. Hence, the set ${}_n\hat{\mathbf{x}}_t(j)$ contains the same elements of the set ${}_n\mathbf{x}_t(j)$, re-indexed sequentially from $1, \ldots, n$.

We denote $\pi_{n,j}(\mathbf{y}_{m,f}) = p(\mathbf{y}_{m,f}|{}_n\mathbf{x}_t(j))$ as the probability density function of the data, where only n-DOA's belong to the targets defined by the partitions of ${}_n\mathbf{x}_t(j)$. Hence, when $n = 0$, all data is due to clutter:

$$\pi_{0,1}(\mathbf{y}_{m,f}) = \left(\frac{\gamma}{2\pi}\right)^{P_{m,f}} \tag{5}$$

The probability density $\pi_{n,j}(\mathbf{y}_{m,f})$ can be calculated by noting that (i) there are $P_{m,f}!/(P_{m,f}-n)!$ ordered ways of choosing DOA's to associate with the n-subset partitions, and (ii) the remaining $(P_{m,f} - n)$-DOA's are explained by the clutter. Therefore,

$$\pi_{n,j}(\mathbf{y}_{m,f}) = \frac{(P_{m,f} - n)!\left(\gamma/2\pi\right)^{P_{m,f}-n}}{P_{m,f}!} \sum_{p_1 \neq p_2 \neq \ldots \neq p_n}^{P_{m,f}} \prod_{i=1}^{n} \psi_{t,m,f}\left(p_i \middle| \hat{x}_i\right), \tag{6}$$

where the function ψ is derived from the assumption that the associated target DOA's form a Gaussian distribution around the true target DOA tracks:

$$\psi_{t,m,f}\left(p_i \middle| x_i\right) = \frac{1}{\sqrt{2\pi}\sigma_\theta^2(m,f)} \exp\left\{-\frac{\left(h_{m\tau}^\theta(x_i(t)) - y_{t+m\tau,f}(p_i)\right)^2}{2\sigma_\theta^2(m,f)}\right\}, \tag{7}$$

where the superscript θ on the state update function h refers only to the DOA component of the state update and $\sigma_\theta^2(m, f)$ can be supplied by the beamformer block.

Note that the DOA distribution (7) is not a proper circular distribution for an angle space. For angle spaces, the von Mises distribution is used as a natural distribution [31]. The von Mises distribution has a concentration parameter with a corresponding circular variance. It can be shown that for small $\sigma_\theta^2 << 1$ (high concentration), the von Mises distribution tends to the Gaussian distribution in (7) [32]. Because the von Mises distribution has numerical issues for small DOA variances, the Gaussian approximation (7) is used in this paper. Hence, special care must be taken in the implementation to handle angle wrapping issues.

The Gaussian in (6) $\psi(\cdot|\cdot)$ are directly multiplied, because the partitions are assumed to be

independent. To elaborate, consider $n = 2$ and $j = 3$ from the example of $\mathcal{I}_2$ above:

$$\begin{aligned}\pi_{2,3}(\mathbf{y}_{m,f}) &\propto \sum_{p_1=1}^{P_{m,f}} \sum_{p_2=1, p_1 \neq p_2}^{P_{m,f}} \psi_{t,m,f}\left(p_1 \middle| \hat{x}_1\right) \psi_{t,m,f}\left(p_2 \middle| \hat{x}_2\right) \\ &\propto \sum_{p_1=1}^{P_{m,f}} \sum_{p_2=1, p_1 \neq p_2}^{P_{m,f}} \psi_{t,m,f}\left(p_1 \middle| x_2\right) \psi_{t,m,f}\left(p_2 \middle| x_3\right).\end{aligned} \tag{8}$$

Hence, the density $\pi_{2,3}(\mathbf{y}_{m,f})$ is a Gaussian mixture that peaks when the updated DOA components of the partitions 2 and 3 ($h^{\theta}_{m\tau}(\cdot)$) are simultaneously close to the observed data. Note that Eqn. (8) guarantees that no measurement is assigned to multiple targets simultaneously.

Given the densities $\pi_{n,j}$, the observation density function can be constructed as a combination of all the target association hypotheses. Hence, by adding mixtures that consist of the data permutations and the partition combinations, we derive the observation density:

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{f=1}^{F} \prod_{m=0}^{M-1} \sum_{n=0}^{K} \frac{\kappa^f_{n,K}}{{}_K\mathcal{C}_n} \sum_{j=1}^{{}_K\mathcal{C}_n} \pi_{n,j}(\mathbf{y}_{m,f}). \tag{9}$$

In (9), the parameters $\kappa^f_{n,K}$ ($\sum_n \kappa^f_{n,K} = 1$) are the elements of a detection (or confusion) matrix. For example, when $K = 2$, $\kappa^f_{0,2}$ is the probability that no target DOA is in the beamformer output, whereas $\kappa^f_{1,2}$ ($\kappa^f_{2,2}$) means that 1 (2) target DOA('s) are present in the beamformer output at each f. These fixed values have to be provided by the user. However, they should be changed adaptively to improve robustness of the particle filter output. For example, when two partitions k_1 and k_2 have close DOA tracks and are about to cross, it is possible that the beamformer's Rayleigh resolution is not enough to output two DOA's for both targets. Then, we change the confusion matrix to indicate the possibility that one of the targets will likely be missed.

C. Particle Filter Proposal Function

In our problem of DOA-only multiple target tracking, the proposal function poses difficult challenges because (i) the state vector dimension is proportional to the number of targets K, hence the number of particles to represent posterior can increase significantly as K increases (the curse of dimensionality), (ii) in many cases, the targets maneuver, hence full posterior approximations are required for robust tracking, and (iii) for full posterior approximations, robustly determining the DOA-only data-likelihood is rather hard. We will address each of these challenges in this section. Note that once the proposal function is formulated, the rest of the particle filter structure is well-defined: weighting and resampling.

1) Partitioned Sampling: A partitioned sampling approach is used to reduce the curse of dimensionality in the particle filter. The basic idea is as follows. Suppose that we (wrongfully) factor the tracking posterior density as

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-T}) \propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-T}) &= \prod_{k=1}^{K} p(\mathbf{y}_t|x_k(t)) \prod_{k=1}^{K} p(x_k(t)|x_k(t-T)) \\ &= \prod_{k=1}^{K} p(\mathbf{y}_t|x_k(t))p(x_k(t)|x_k(t-T)) \propto \prod_{k=1}^{K} q_k(x_k(t)|\mathbf{y}_t, x_k(t-T)). \end{aligned} \tag{10}$$

In this case, the target posterior is conveniently a product of the partition posteriors (i.e., individual target posteriors) $q_k(\cdot|\cdot)$. We can then generate samples for each partition according to its posterior (i.e., $x_k^{(i)} \sim q_k(x_k(t)|\mathbf{y}_t, x_k(t-T))$) and merge them to represent $\mathbf{x}_t$. It can be proved that the resulting particle distribution is the same as when we generate $\mathbf{x}_t$ directly from the full posterior $p(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-T})$. However, in the partitioned sampling case, the computational complexity of the state vector generation is linear with respect to the number of targets K as opposed to exponential when the state vector is sampled from the full posterior.

In our problem, we can approximately factor out the tracking posterior to exploit the computational advantage of the partitioned sampling. Note that in our case, the target dynamics can already be factored out because we assume the targets are moving independently.[4] Unfortunately, the observation density does not factor out, because the observed DOA data cannot be immediately associated with any of the partitions. However, for a given partition, if we assume that the data is only due to that partition and clutter (hence, the DOA data corresponding to other partitions are treated as clutter), we can do the following approximate factorization on the observation likelihood (9):

$$p(\mathbf{y}_t|\mathbf{x}_t) \approx \prod_{k=1}^{K} p(\mathbf{y}_t|x_k(t)) = \prod_{k=1}^{K} \prod_{f=1}^{F} \prod_{m=0}^{M-1} \left\{ \kappa_{0,1}^f \left(\frac{\gamma}{2\pi}\right)^{P_{m,f}} + \kappa_{1,1}^f \left(\frac{\gamma}{2\pi}\right)^{P_{m,f}-1} \sum_{p=1}^{P_{m,f}} \frac{\psi_{t,m,f}\left(p\middle|x_k\right)}{P_{m,f}} \right\}. \tag{11}$$

Hence, for our problem, an approximate partition posterior is the following

$$q_k(x_k(t)|\mathbf{y}_t, x_k(t-T)) \propto p(\mathbf{y}_t|x_k(t))p(x_k(t)|x_k(t-T)), \tag{12}$$

where $p(\mathbf{y}_t|x_k(t))$ is given in (11) and $p(x_k(t)|x_k(t-T)) = \mathcal{N}\left(h_T(x_k(t-T)), \Sigma_u\right)$ with $\Sigma_u =$

[4]When the targets are moving closely in tandem, there is a possibility that the beamformer block may not resolve them. Hence, they can be treated as a single target. In other cases, the independence assumption still works. However, if high resolution observations (e.g, top down video images of the target plane) are available, it is better to also model the interactions of targets. A good example using Monté-Carlo Markov chain methods can be found in [33].

$\mathrm{diag}\{\sigma_\theta^2, \sigma_Q^2, \sigma_\phi^2\}$. Note that (11) will not be used as the data-likelihood of the particle filter. The above approximate factorization of the data-likelihood is to make use of the partitioned sampling strategy to propose particles. To calculate the particle filter weights, the full posterior uses the observation density (9).

2) A Gaussian Approximation for Partition Posteriors: In this section, we derive a Gaussian approximation to (12) to use as a proposal function for the particle filter. Hence, we use the current observed data to propose the filter's particle support, also incorporating target maneuvers if there are any. This approximation can be done in two ways: (i) Laplace's method or (ii) the mode hungry Monte-Carlo method. The first method is an order of magnitude faster than the second, but it is not as robust. The filter monitors its tracking and can decide to switch between methods to find the partition posteriors.

By default, the filter uses Laplace's method to approximate $p(\mathbf{y}_t|x_k(t))$ in (12) and thereby derive the partition proposal functions of the particle filter, denoted as $g_k(x_k(t)|\mathbf{y}_t, x_k(t-T))$. Laplace's method is an analytical approximation of probability density functions based on a Gaussian approximation of the density around its mode, where the inverse Hessian of the logarithm of the density is used as a covariance approximation [19]. It can provide adequate approximations to posteriors that are as accurate and sometimes more accurate than the approximations based on third-order expansions of the density functions [18]. The computational advantage of this approach is rather attractive because it only requires first- and second-order derivatives. The condition for the accurate approximation is that the posterior be a unimodal density or be dominated by a single mode. Hence, it is appropriate for approximating the partition posteriors of the particle filter.

Laplace's approximation requires the calculation of the data statistics. The Laplacian approximation is described in detail in [34]. For this paper, it is implemented with the Newton-Raphson recursion with backtracking for computational efficiency. Because of the constrained nature of the algorithm's modified cost function, this method is sometimes susceptible to "shadow tracking" and can diverge under certain conditions (refer to Fig. 6). Shadow tracking refers to the scaled target tracks in x-y that can lead to the same DOA track as illustrated in Fig. 6. Shadow tracks may cause the Laplacian approximation deviate from the truth because of the motion smoothness constraints used to calculate it.

Any divergence in the Newton-Raphson mode calculation can be detected by monitoring the number of DOA measurements that fall into the gate of the mode estimate (Fig. 4). If there are less than a threshold number of DOA's (typically M/2) within a σ_g neighborhood of the Newton-Raphson mode's DOA track, then the particle filter uses the MHMH method for determining the mode. After the MHMH iterations are over, if the MHMH corrected mode fails to have the

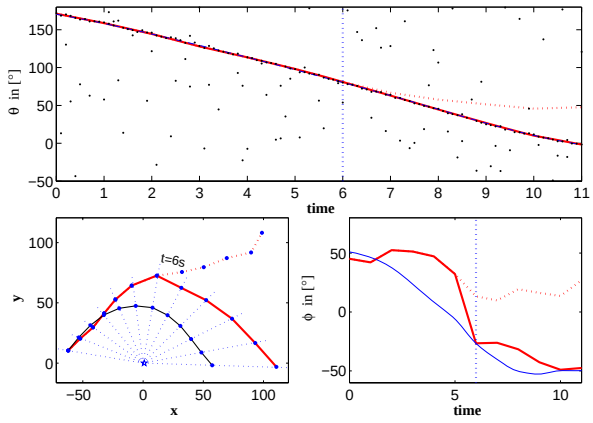


Fig. 6: The particle filter is susceptible to shadow tracking, when only the Newton-Raphson recursion is used to determine the data-likelihood function. (*Top*) DOA tracking results with (solid line) and without (dotted line) MHMH correction. At $t = 6$s, the MHMH automatically corrects the heading bias. (*Bottom Left*) True track (inside), the shadow track (middle) and the diverged track (outside). The dotted line is without the MHMH correction. (*Bottom Right*) Filter heading estimates. Note the MHMH correction at time $t = 6$s.

required number of DOA's within its gate, the partition is declared dead. Details of the MHMH method are omitted due to lack of space. The final expression for the partition proposal functions to be used in the particle filter is given by

$$g_k(x_k(t)|\mathbf{y}_t, x_k(t-T)) \sim \mathcal{N}\left(\mu_g(k), \Sigma_g(k)\right) \tag{13}$$

where the Gaussian density parameters are

$$\begin{aligned} \Sigma_g(k) &= \left(\Sigma_y^{-1}(k) + \Sigma_u^{-1}\right)^{-1} \\ \mu_g(k) &= \Sigma_g(k)\left(\Sigma_y^{-1}(k) x_{k,\text{mode}} + \Sigma_u^{-1} h_T(x_k(t-T))\right), \end{aligned} \tag{14}$$

where $x_{k,\text{mode}}$ is the mode of $p(\mathbf{y}_t|x_k(t))$, and $\Sigma_y^{-1}(k)$ is the Hessian of $p(\mathbf{y}_t|x_k(t))$ at $x_{k,\text{mode}}$, calculated by either one of the methods in this section.

D. Algorithm Details

Pseudo-code of the particle filter algorithm is given in Table I. The filter implementation employs an efficient resampling strategy, named "deterministic resampling", first outlined by Kitagawa [35]. This resampling strategy is preferred because of (i) the efficient sorting of the particles and (ii) the number of random number generations. The deterministic resampling strategy also has known convergence properties [35]. Faster resampling schemes without conver-

gence proofs are also available [36] and these could make a difference in the filter computation, especially when $K = 1$. Effects of the resampling stage can be seen at the computational analysis done in Sect. VI.

TABLE I: **Particle Filter Tracker Pseudo-Code**

Given the observed data $\mathbf{y}_{t,f} = \{y_{t+m\tau,f}(p)\}_{m=0}^{M-1}$ in $[t, t+T)$, do

1. For $i = 1, 2, \ldots, N$
   - For $k = 1, 2, \ldots, K$
     sample $x_k^{(i)}(t) \sim g_k(x_k^{(i)}(t)|\mathbf{y}_t, x_k^{(i)}(t-T))$, given by Eqn. (13).
   - Form $\mathbf{x}_t^{(i)} = \left[x_1^{(i)}(t),\ x_2^{(i)}(t), \ldots,\ x_K^{(i)}(t) \right]^T$.
2. Calculate the weights

$$w^{*(i)}_t = w^{(i)}_{t-T} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-T})}{\prod_k g_k(x_k^{(i)}(t)|\mathbf{y}_t, x_k^{(i)}(t-T))},$$

   where $p(\mathbf{y}_t|\mathbf{x}_t^{(i)})$ is fully joint observation density, given by Eqn. (9).
3. Normalize the weights:

$$w_t^{(i)} = \frac{w^{*(i)}_t}{\sum_i w^{*(i)}_t}.$$

4. Make estimation: $E\{f(\mathbf{x}_t)\} = \sum_{i=1}^N w_t^{(i)} f(\mathbf{x}_t^{(i)})$.
5. Resample the particles:
   - Heapsort the particles in a ascending order according to their weights: $\mathbf{x}^{(i)}_t \rightarrow \tilde{\mathbf{x}}_t^{(i)}$.
   - Generate $\omega \sim \mathcal{U}[0, 1)$.
   - For $j = 1, 2, \ldots, N$
     a. $u^{(j)} = \frac{j-\omega}{N}$,
     b. Find i, satisfying $\sum_{l=1}^{i-1} \tilde{w}_t^{(i)} < u^{(j)} \leq \sum_{l=1}^{i} \tilde{w}_t^{(i)}$,
     c. Set $\mathbf{x}_t^{(j)} = \tilde{\mathbf{x}}_t^{(i)}$.
6. Associate the DOA estimates with partitions using a nearest neighbor approach.
   - Calculate the DOA track for each partition of $\mathbf{x}$, $\mathbf{x} = \sum_{i=1}^N w_t^{(i)} \mathbf{x}_t^{(i)}$, using the state update equation.
   - Associate the nearest DOA estimates within a threshold angle distance, with each partition.
   - Report all the unassociated DOA estimates to the MHMH block.
7. Delete partitions with the number of associated DOA estimates less than a threshold number (e.g., M/2).
8. Randomly merge the new partitions coming from the MHMH block, if there are any.

Finally, the partitions are managed by the specific interaction between the particle filter and the MHMH block. New partitions are introduced into the particle filter, using the distribution supplied by the MHMH block. First, the particle filter block deletes the DOA's from the observed data that belong to the partitions that it is currently tracking (gating) after estimation (Fig. 4). Gating here is a simple distance tresholding operation and does not incur significant computation. Then, the remaining DOA's are used by the MHMH block to determine any new partition distributions. For the particle filter to stop tracking a given target, a deletion operation is necessary. The particle filter can delete partitions at either the proposal stage or after estimation. Lastly, note that our

implementation of the particle filter does not make partition associations such as partition *split* or *merge*. We leave these decisions to a higher level fusion algorithm in the sensor network.

V. The MHMH block

The objective of the MHMH block[5] is to find any new targets in the observed data and determine their probability density distributions. The MHMH block uses the matching pursuit idea by Mallat [37] by consecutively applying the MHMH algorithm on the ungated DOA's until stopping conditions described below are satisfied. Each iteration of the MHMH block results in a new target state vector distribution, whereas the total number of MHMH block iterations gives the number of new targets.

The MHMH block employs the mode hungry Metropolis-Hastings sampling algorithm to determine the distribution for the target that has the highest mode in the multi-target observation density. The MHMH sampling algorithm [20] is an accelerated version of the Metropolis-Hastings algorithm [38]–[41] that generates samples around the modes of a target density. The pseudocode for the MHMH sampling algorithm is given in Table II (reproduced from [20]). For our system, we use a random walk for the candidate generating function $q(\cdot)$ in Table II with the walk noise variances set to $\sigma_\theta = 0.5°$, $\sigma_Q = 0.01\text{s}^{-1}$, and $\sigma_\phi = 4°$. Cross sampling is applied every $L_{jump} = 2$ iterations on subpartitions of size M_l, which is typically half the number of MHMH particles N_{MHMH}. For our problem, the sampling algorithm is iterated a total of 150 iterations. Finally, the target density $\pi(\cdot)$ is obtained by setting $K = 1$ on the observation density (9):

$$\pi(x_i) \propto \prod_{f=1}^{F} \prod_{m=0}^{M-1} \left\{ \kappa_{0,1}^{f} \left(\frac{\gamma}{2\pi} \right)^{P_{m,f}} + \kappa_{1,1}^{f} \left(\frac{\gamma}{2\pi} \right)^{P_{m,f}-1} \sum_{p=1}^{P_{m,f}} \frac{\psi_{t,m,f}\left(p \middle| x_i \right)}{P_{m,f}} \right\}. \tag{15}$$

The pseudocode of the MHMH block is given in Table III. The algorithm creates a set of particles to input the MHMH sampling algorithm. This initial set consists of some of the residual DOA's and a grid of Q-ϕ values. After the MHMH sampling is over, the mode of these particles is monitored by two conditions. The first condition checks if there are sufficient DOA's in the mode's DOA gate. This ensures that the mode is relatively accurate when used by the particle filter. The second condition makes sure that mode belongs to a physical target as opposed to some alignment of the clutter. If these conditions are not satisfied, the MHMH block exits and no new partition is introduced. Otherwise, we resample N particles with replacement from the

[5]The MHMH block is not to be confused with the MHMH correction of the proposal stage.

TABLE II: MHMH Sampling Algorithm

1. At iteration l, decide if cross-jumping is needed for $x^{(l)}$, i.e., every $L_{jump} = 2$ iterations.
2. If not, then for each particle x_i, $i = 1, 2, \ldots, N_{MHMH}$, use the Metropolis-Hastings scheme:
   - generate a candidate y_i using $q(x_i, y_i)$,
   - calculate the acceptance ratio
   $$\alpha(x_i, y_i) = \min\left\{1, \frac{\pi(y_i)q(y_i, x_i)}{\pi(x_i)q(x_i, y_i)}\right\},$$
   - sample $u \sim \mathcal{U}(0, 1)$,
   - if $u \leq \alpha(x_i, y_i)$, set $x_i^{(l+1)} = y_i$, else, $x_i^{(l+1)} = x_i^{(l)}$.
3. Use the Mode-Hungry scheme:
   - determine a subpartition of size $M_l < N_{MHMH}$, (e.g.,$M_l = N_{MHMH}/2$),
   - order the current particles according to their probabilities in descending order: $x_i \rightarrow x_j^*$, where x^* is the ordered particle set,
   - generate candidates $y^*(1)$ for $x^*(1) = \{x_j^* | j : j = 1, 2, \ldots, N_{MHMH} - M_l\}$, using $q(\cdot, \cdot)$,
   - calculate the acceptance ratio $\alpha(x^*(1), y^*(1))$, and set $x_j^{(l+1)}$ to $x_j^*(1)$ or $y_j^*(1)$, accordingly for $j = 1, 2, \ldots, N_{MHMH} - M_l$,
   - distribute M_l candidates $y^*(2)$ from $x^*(1)$ uniformly,
   - set $x_j^{(l+1)}$ to $y^*(2)$ for $j = N_{MHMH} - M_l + 1, \ldots, N_{MHMH}$.

MHMH output and declare a new partition in the particle filter. The DOA's belonging to the new mode are gated out, and the procedure is repeated.

TABLE III: MHMH Block Pseudo-Code

1. Let $\mathbf{y}_{t,f}^g$ be the residual set of DOA observations from the particle filter. If there are no partitions in the particle filter, then $\mathbf{y}_{t,f}^g = \mathbf{y}_{t,f}$. Define N_θ as the number DOA's in $\mathbf{y}_{t,f}^g$.
2. Set $K_{\text{new}} = 0$. Create a uniform grid for Q and ϕ. Choose N_Q starting values for Q based on the physical motion constraints, and choose N_ϕ heading directions, uniformly spaced in $(-\pi, \pi]$.
3. `while` $N_\theta > 5$,
   - $\{\theta_i\}_{i=1}^{N_\theta} = \{y_{t+m\tau,f}^g(p)\}_{m=0}^4$.
   - Replicate the Q-ϕ grid N_θ times and combine it with each θ_i. Generate the initial set of particles: $\{x^{(j)}\}_{j=1}^{N_\theta N_Q N_\phi}$.
   - Use MHMH (Table II) to identify one mode (corresponding to one of the targets) of the distribution.
   - If the DOA gate of the highest probability particle has less than 5 DOA's, `break`.
   - If Q value of the highest probability particle is greater than $Q_{th} = 0$, `break`.
   - Resample N particles with replacement from the MHMH output. $K_{\text{new}} = K_{\text{new}} + 1$. Report the resampled set to the particle filter.
   - Delete the DOA's within 3° of the mode DOA track (gating) to obtain the residual $\mathbf{y}_{t,f}^g$.

VI. Computational Complexity

This section describes the computational requirements of the particle filter and MHMH blocks described in Sect. IV and Sect. V, respectively. The preprocessing beamforming block has several established efficient implementation methods based on Q-R matrix decomposition [21] and is not considered here. This section uses the notations in Table IV to derive the implementation complexities.

TABLE IV: Notations for Computational Complexity Analysis

Notation	Description	Typical values
N	Number of particles	100
N_s	State vector size	3
K	Number of targets	3
M	Number of batch samples for DOA estimation	10 (> 5)
P	Number of DOA peaks in the beamformer pattern	3 ($\geq K$)
F	Narrow-band beamformers at different center frequencies	1
N_{iter}	Number of iterations in Newton search	15

A. Computational Analysis of the Particle Filter Block

The choice of our proposal function drives the complexity and execution time of the tracking algorithm. As expected complexity increases with the number of targets and the number of particles used. The analytical complexity analysis and simulation results suggest that the overall complexity of the algorithm is of $O(NPKM)$. This is also apparent from the linear increase in the total computational time in Fig.7. The computational time with respect to the number of targets is approximately quadratic. Moreover, the complexity of the particle proposal stage (Step 1 in Table I) and the weight evaluation stage (Steps 2 and 3 in Table I) are comparable. For small number of particles, up to $N = 500$, the execution time of the Newton's iteration stage is considerably more compared to other stages. Note that this stage does not depend on the number of particles used, but depends on the number of targets. The number of iterations in the Newton search is typically between 10 and 20 because of the adaptive step-size and the stopping conditions.

B. Computational Analysis of the MHMH Block

The computational requirements of the MHMH block are shown in Table VI. The main complexity of the MHMH algorithm consists of generating the particles and identifing the data mode. The MHMH complexity analysis is similar to that of the particle filter block and is based on the algorithms in Tables II and III. The algorithm complexity is directly proportional to the

TABLE V: Computational Complexity Analysis - Particle Filter Block

Description (Equation)	**Number of Operations**				**Complexity**
	Adds	*Mult.*	*Trans.*	*Other*	
Propose particles (Table I - Step 1)					
State $\mathbf{x_t}$ update $\mathbf{x_t}\|\mathbf{x_{t-T}}, \mathbf{y_t}$ (3)	$6NK$	$12NK$	$8NK$	–	$O(NK)$
Evaluate mode $\mathbf{x_{mode}}$	$12NM$	$6NM$	$4NM$	$3NM$	$O(NM)$
Newton's iteration G, H	$K^2N_s^2M$	K^2M	$10K$	KPM^a	$O(MK^2)$
Evaluate μ_g and Σ_g (14)	$NK^2N_s^2$	$2NK^2N_s^2$	–	–	$O(NK^2)$
Sample particles	$NK^2N_s^2$	$NK^2N_s^2$	–	NKN_s^b	$O(NK^2)$
Update particle weights (Table I - Steps 2,3)					
Evaluate state likelihood, proposal (13)	$NK^2N_s^2$	$NK^2N_s^2$	$2NKN_s$	NKN_s^a	$O(NK^2)$
Evaluate data-likelihood (11)	$3NKPM$	$10NKM$	$NKPM$	$NKPM^a$	$O(NKPM)$
Evaluate weights	$3N$	$2N$	N	–	$O(N)$
State estimation (Table I - Step 4)	NKN_s	NKN_s	–	–	$O(NK)$
Resampling (Table I - Step 5)	$2N$	–	–	$2N^c$	$O(N)$
Overall					$O(NPKM)$

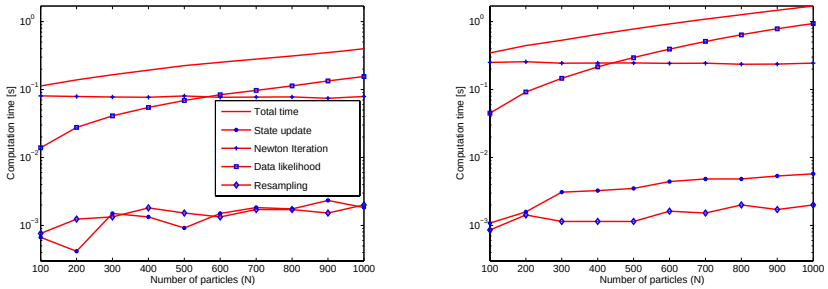
[a] Modulo operations, [b] Random number generation, [c] Compare operations.

burn-in iterations, denoted by L, that the algorithm is run for convergence. Typically, the number of burn-in iterations is in the range 150 to 400, proportional to the number N_{MHMH} of particles in the initial set. During each iteration, significant time is spent in evaluating the target density $\pi(\cdot)$, shown in (15). As can be seen from Table VI, this depends on both on P and M.

TABLE VI: Computational Complexity Analysis - MHMH Block

Description (Equation)	**Number of Operations**				**Complexity**
	Adds	*Mult.*	*Trans.*	*Other*	
Generate candidate y_i (Table II Step 2)	LNN_s	LNN_s^2	–	LN^b	$O(LN)$
Calculate acceptance ratio (Table II - Step 2)					
Evaluate $\pi(\cdot)$	$3LNPM$	$10LNM$	$LNPM$	$LNPM^a$	$O(LNPM)$
Other	LN	–	LN	$2LN^c$	$O(LN)$
Mode-Hungry scheme (Table III)	–	–	–	–	$O(LN \log N^d)$

[a] Modulo operations, [b] Random number generation, [c] Compare operations, [d] Sort operations.



(a) One target (b) Three targets

Fig. 7: Execution times for various particle filter stages.

VII. Other Tracker Solutions

This section discusses alternative DOA tracking algorithms to replace the particle filter in Fig. 3, namely the extended Kalman filter and the Laplacian filter. These algorithms incur less computational cost than the particle filter. Laplacian filter solution can even be implemented in a parallel fashion. However, the extended Kalman solution is difficult to extend to multiple targets. In addition, the Laplacian filter performance is sensitive to its initialization.

A. Extended Kalman Filter

The EKF solution is an M-scan multihypothesis tracking (MHT) filter. As opposed to standard Kalman filter based approaches, the observation equation (9) requires M snapshots of data to perform state estimates. Hence, the usage of MHT here differs from traditional approaches.[6] Compared to the particle filter approach that updates the state every T seconds, the MHT approach estimates states every τ seconds, based on the sliding M-DOA snapshots. This is similar to a sliding window approach [10], but assumes that the number of targets is fixed and known since it is estimated by the MHMH block.

In the EKF approach explicit measurement-to-target associations have to be made. We use the JPDA approach to generate a set of hypotheses, relating measurements-to-targets, during each snapshot [10]–[13], [42]. Similar hypotheses are generated for each of the M DOA's, starting from time t. The best hypothesis at each snapshot is chosen, based on the probabilities obtained from the JPDA. These M-best hypotheses are then used in associating the measurements-to-targets at each snapshot. If a measurement-to-target association for a specific target cannot be

[6]In traditional MHT, using the M-scan associations from one snapshot $t+(m-1)\tau$ are carried over to the next snapshot $t+m\tau$. However, associations here during each snapshot $t+m\tau$ depend only on the predicted measurement, based on state x_t, and do not depend on associations at $t+(m-1)\tau$

found, it is assigned a probability of zero. For each target, a modified innovation term is obtained as

$$\nu_k(t) = \{\beta_m(t) \otimes \left(y_{t+m\tau}(p_k) - h^{\theta}_{m\tau}\hat{x}_k(t|t-\tau)\right)\}_{m=0}^{M-1}, \tag{16}$$

where $y_{t+m\tau}(p_k)$ is the measurement associated with the target k at time instant $t+m\tau$ and $\beta_m(t)$ is the probability corresponding to the hypothesis at the mth snapshot obtained using the JPDA. The EKF covariance update is

$$\mathbf{P}_k(t|t) = P_{cc}\left(\mathbf{P}_k(t|t-\tau)\right) + (1-P_{cc})\left(\mathbf{P}_k(t|t-\tau) - \mathbf{K}_k(t)\mathbf{M}_k(t)\mathbf{K}_k^T(t)\right) + \widetilde{\mathbf{P}}, \tag{17}$$

where $\widetilde{\mathbf{P}} = \mathbf{K}_k(t)\nu_k(t)\nu_k^T(t)\mathbf{K}_k^T(t)$, $\mathbf{K}_k(t)$ is the Kalman gain, P_{cc} is the probability that the associations are correct, $\mathbf{P}_k(t|t-\tau)$ is the state prediction covariance, and $\mathbf{M}_k(t)$ is the measurement covariance obtained as in the standard EKF [10].

B. Laplacian Filter

Laplacian filter is a Bayesian filter that uses the product of the partition proposal functions (13), as the tracking posterior. Hence, the posterior of the Laplacian filter is a combination of multi-target Gaussian approximations:

$$p(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-T}) = \prod_{k=1}^{K} q_k(x_k(t)|\mathbf{y}_t, x_k(t-T)). \tag{18}$$

This approximation asymptotically has an $O(M^{-2})$ estimation error [18] for a single target. However, it is difficult to analyze the asymptotic effects on the multi target posterior.

The Laplacian filter handles multiple targets using the partition approach. Each target posterior is assumed independent. Hence, the filter can be parallelized for faster implementation. In addition, because the Laplacian filter uses the Newton-Raphson recursion to calculate the data-likelihood, its complexity is similar to the particle filter for $N < 200$. However, the particle filter cannot be implemented in a parallel structure due to the presence of the cross terms in the data-likelihood. In comparison to the EKF, we need M estimates from the EKF compared to one estimate from the particle filter or Laplacian filter every T seconds. Furthermore, using M JPDA steps to make explicit target-data associations during each estimate of the EKF also increases its complexity. Considering this, the Laplacian filter complexity is less than that of the EKF implementation.

VIII. SIMULATIONS

Table VII summarizes the simulation parameters used in this section. We first give a multi-target multi-frequency tracking example. The automated system performance is demonstrated

next. Finally, we compare the tracking performance of the algorithms presented in the paper.

TABLE VII: Simulation Parameters

Number of particles, N	200	Beamformer batch period, τ	0.1s
θ state noise $\sigma_{\theta,k}$	1°	Clutter space parameter, γ	600
Q state noise $\sigma_{Q,k}$	0.05s^{-1}	Probability of target miss, $\kappa^f_{0,K}$	0.1
ϕ state noise $\sigma_{\phi,k}$	10°	Number of batch samples, M	10
Measurement noise σ_θ	1°	Number of DOA peaks, P	4 ($\geq K$)
Tracker sampling period, T	1s	Number of beamforming frequencies, F	1

A. Multi-Frequency Tracking

Figure 8 shows a challenging three targets scenario, where the targets cross in the bearing space as they are maneuvering. For this simulation, two independent layers of DOA estimates are used each with the correct mean and a variance of $(3^\circ)^2$. The particle filter does a good job of keeping the target association as well as determining the maneuvers, which are most prominent in the heading plot. In Fig. 8, the dashed line is the ground truth, whereas the solid line is the particle filter estimates. The dots and diamonds represent the two independent noisy DOA measurement sets.

There is a small bias in the filter DOA estimates between $t = 4$s and $t = 6$s where the targets are crossing. This bias is, in part, also due to the target maneuvers, which start at $t = 4$s. The filter maintains the track coherence in this difficult case by using the independent frequency observations (when only one of them is present, we observed that the filter can confuse the targets). Although its estimates deteriorate in the region where targets are crossing as well as maneuvering, it locks back on the targets after the transient region.

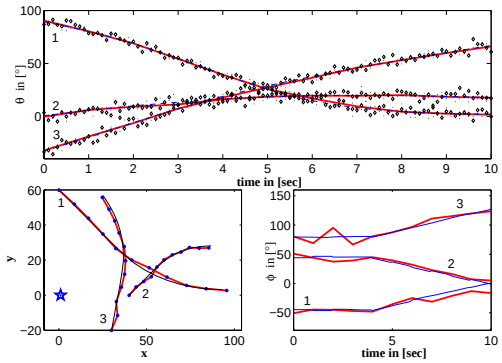
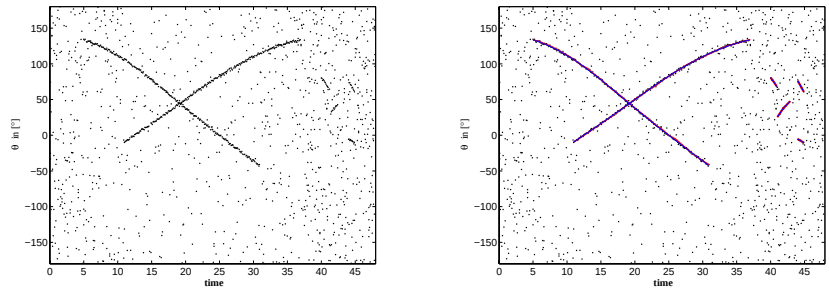


Fig. 8: DOAs, represented by diamonds and dots, are generated by independent noise and are input to the filter unsorted. Below, we also show the track and heading estimates.

B. System Simulation

This subsection simulates the automatic system, described by the block diagram in Fig. 3. Two targets are being tracked by the system, where one target appears in the beamformer output between $t = 5$s and $t = 32$s, and the other one between $t = 11$s and $t = 37$s (Fig. 9). The targets are successfully initialized by the MHMH block and then deleted by the particle filter. The track coherence is also maintained around $t = 20$s, because the particle filter also keeps the target motion parameters. In this case, there were no false target initializations by the MHMH block. The short bearing tracks between times $t = 40$s and $t = 45$s were also embedded into the data set to demonstrate the ability to initiate and kill tracks.



(a) System Input (b) System Result

Fig. 9: Two targets are successfully initialized and tracked by the automated system. The system can successfully detect short target tracks as well.

In Fig. 10, we provide a simulation to demonstrate the performance of our detection scheme. In this simulation, we randomly picked a target track for a duration of 1 second. We generated the corresponding bearing track for $M = 10$ with varying bearing errors. We then added a spurious track with $M = 10$ by generating random bearing estimates that are uniformly distributed in $[0, 2\pi)$. We repeated this for a Monte-Carlo run of size 100. During each simulation, we also randomly replaced any bearings in the input data with random clutter with probability P_c to simulate missing bearings. To determine the probability of missed detections in our initialization algorithm, we counted the number of times our initialization algorithm failed to detect the target and then divided that by the size of the Monte-Carlo run. For DOA error variances less than 2°, the detection scheme is quite successful, i.e., the probability of miss is less than 0.1. As the DOA noise increases above the gate size, the detection performance loses its monotonicity.

C. Comparisons of DOA Trackers

In Figs. 11, 12, and 13, Monte-Carlo simulations compare the estimates obtained using the particle filter, the EKF, and the Laplacian filter. The state estimation performances of these filters

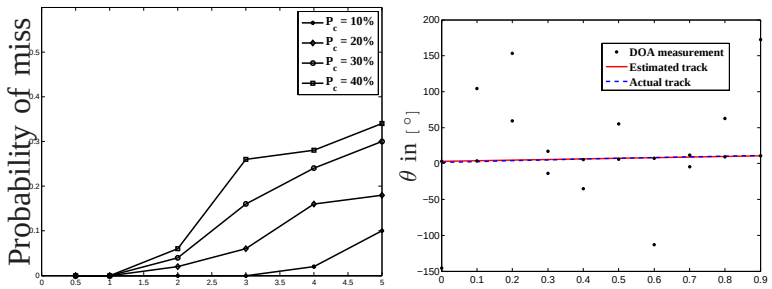


Fig. 10: (a) Probability of missed detections in the MHMH block with varying noise variance in the measurement data. A partition gate size of $\sigma_g = 3^\circ$ is used. (b) A sample realization with $\sigma_\theta = 2^\circ$ from the Monte-Carlo run that calculates the probability of miss in part (a).

are comparable. The Laplacian filter tends to have less variance than the particle filter in general, because the actual posterior is wider than the Gaussian approximation. Hence, we can consider the Laplacian filter as a mode tracker. The estimates of the particle filter can be further improved by also adding cross partition sampling as suggested in [2] or auxiliary sampling [43].

The filters' tracking performance in most cases are similar, when initialized correctly around the true target state. However, the extended Kalman filter is more sensitive to errors in initialization compared to the other two filters. The EKF may have difficulty tracking the DOA's correctly, when initialized with the mean of the particle set generated by the MHMH block. The EKF can completely lose the DOA tracks, if the initial estimates are not accurate, whereas the other two filters can still absorb the discrepancies. There is also a notable decrease in the EKF performance, when targets maneuver. The particle filter is the most robust to target maneuvers compared to the other two methods due to the diversity provided by the particles.

We also compare the particle filter and the Laplacian filter presented in this paper with the particle filter that uses the received acoustic signals directly (denoted as DPF) [4] in Fig. 14. For the comparison, we use the classical narrow-band observation model to generate the acoustic data as described in [4]. For the simulation, we use a constant frequency target at 150Hz, 8 microphone circular array with .45 wavelength separation, and a sampling frequency of 1000Hz. For the synthetic target track, a process noise is added with variances $\sigma_\theta = .5^\circ$, $\sigma_Q = 0.05$, and $\sigma_\phi = 8^\circ$. The process noise explains the abrupt changes in the heading estimates in Fig. 14 at each time period. Also, the acoustic SNR at the array is approximately 7dB in the simulation.

To use the filters presented in this paper, we applied a simple MVDR beamformer on the acoustic data at the center frequency and calculated DOA estimates. The filters in comparison use the same noise variances for the state update equation. The DPF runs three orders of magnitude

slower than the particle filter presented here and performs better because the observation model is perfectly matched by the data. The particle filter has a larger variance partly because of the clutter handling mechanism that creates a noise floor for the pdf (for a single target, the PDF is a raised Gaussian). Not surprisingly, the Laplacian filter show a similar estimation variance as the DPF in bearing estimation because it tracks the mode of the target track with a narrower posterior approximation.

IX. Conclusions

In this paper, we present an automated framework for multiple target DOA tracking based on a batch measurement model. The DOA batches are treated as images to naturally handle the data association and ordering issues. The presence of multiple targets is handled using a partition approach. The observation likelihoods are calculated jointly and are assigned by using the templates created by the state vectors and the state update equation. We present three filters for the DOA tracking problem: the particle filter, the Laplacian filter, and the extended Kalman filter. These filters offer computationally attractive alternative to filters based on acoustic sensor outputs directly. The filters are compared in terms of computational cost, sensitivity to initialization, and robustness. For the initialization of the automated DOA tracking system, we also discuss a sampling algorithm based on the matching pursuit idea, using the mode hungry Metropolis-Hastings method.

References

[1] C.K. Sword, M. Simaan, and E.W. Kamen, "Multiple target angle tracking using sensor array output," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, pp. 367–372, Mar. 1990.

[2] M. Orton and W. Fitzgerald, "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Trans. on Signal Processing*, vol. 50, no. 2, pp. 216–223, February 2002.

[3] Y. Zhou, P.C. Yip, and H. Leung, "Tracking the direction-of-arrival of multiple moving targets by passive arrays: Algorithm," *IEEE Trans. on Signal Processing*, vol. 47, no. 10, pp. 2655–2666, October 1999.

[4] V. Cevher and J. H. McClellan, "General direction-of-arrival tracking with acoustic nodes," *IEEE Trans. on Signal Processing*, vol. 53, no. 1, pp. 1–12, January 2005.

[5] J. R. Larocque, J. P. Reilly, and W. Ng, "Particle filters for tracking an unknown number of sources," *IEEE Trans. on Signal Processing*, vol. 50, no. 12, pp. 2926–2937, Dec. 2002.

[6] T. Kailath, A.H. Sayed, and B. Hassibi, *Linear Estimation*, Prentice Hall, 2000.

[7] W.L. Brogan, *Modern Control Theory*, Prentice Hall, 1991.

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

[9] J.S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, September 1998.

[10] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*, Academic-Press, 1988.

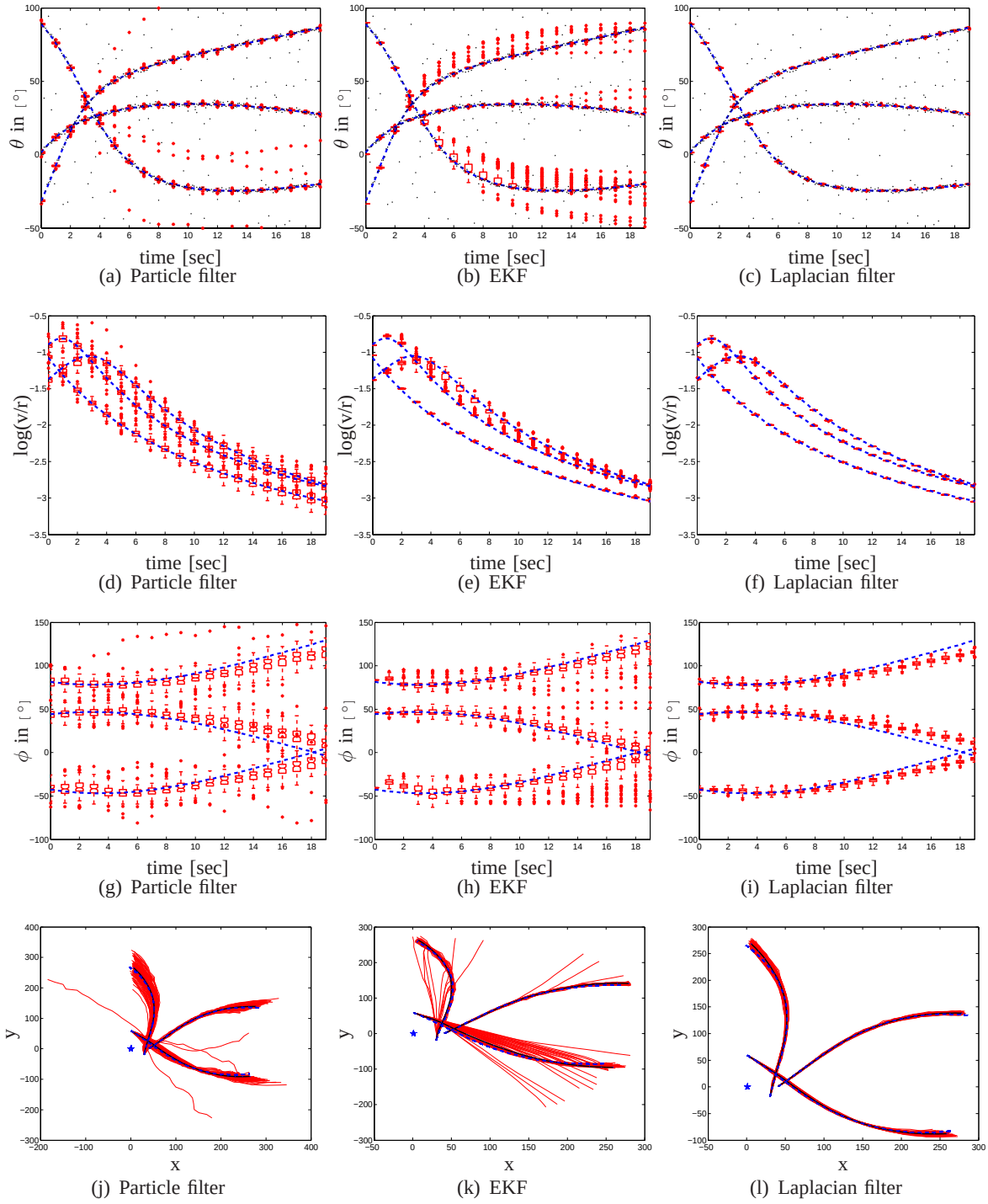


Fig. 11: Monte-Carlo run results for each filter using 100 independent noise realizations using Matlab's *boxplot* command. The ground truth is shown with the dashed lines. The last row shows the track estimates and their mean. In this example, the Laplacian performs quite well and hence there were no MHMH iterations done in the proposal stage of the particle filter. This is due to the smooth movement of the targets.

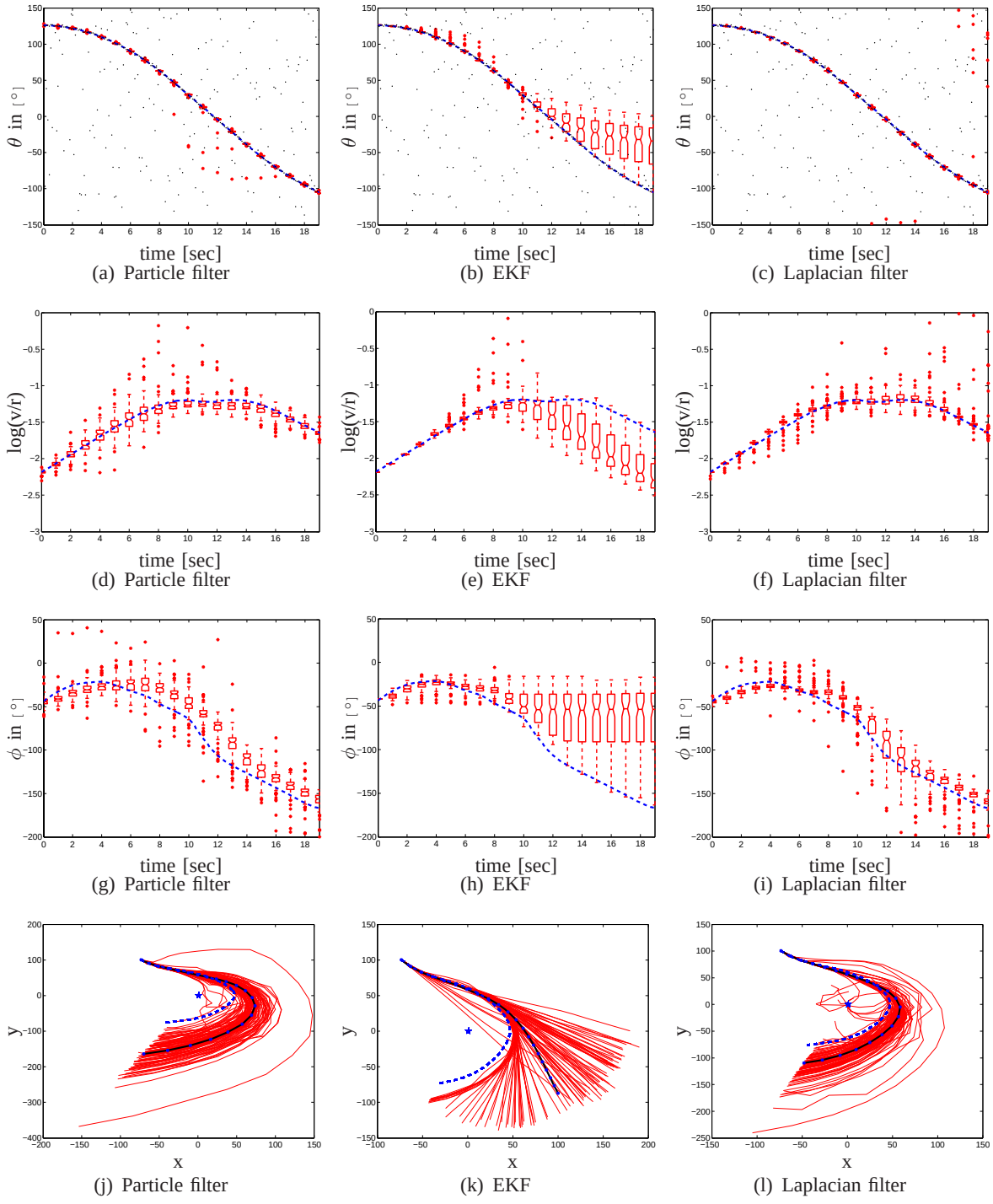


Fig. 12: Monte-Carlo run results for each filter using 100 independent noise realizations for a target that maneuvers rapidly. The last row shows track estimates and their mean. In this example, the MHMH iterations in the Laplacian calculation were necessary to pull the Laplacian towards the true track because of the abrupt maneuvers at $t = 11$s and $t = 12$s. Note that the filter track estimates shadow the true track due to the constant velocity assumption.

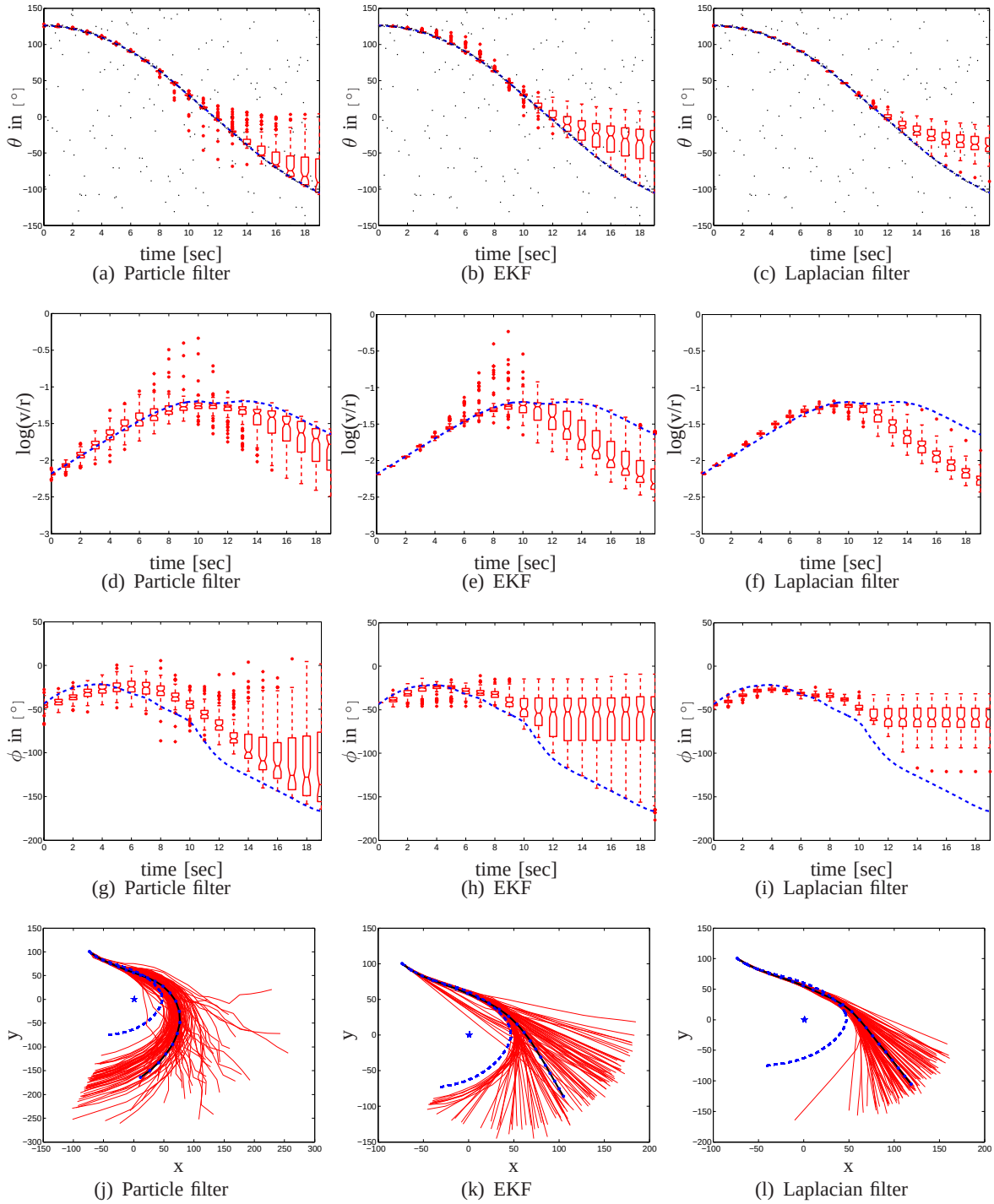


Fig. 13: Same example in Fig. 12 without the MHMH correction. In this case, the Laplacian filter cannot handle the maneuver. The particle filter can still track the target due to its particle diversity. However, the estimates of the particle filter are now worse than the estimates with the MHMH correction at the proposal stage.

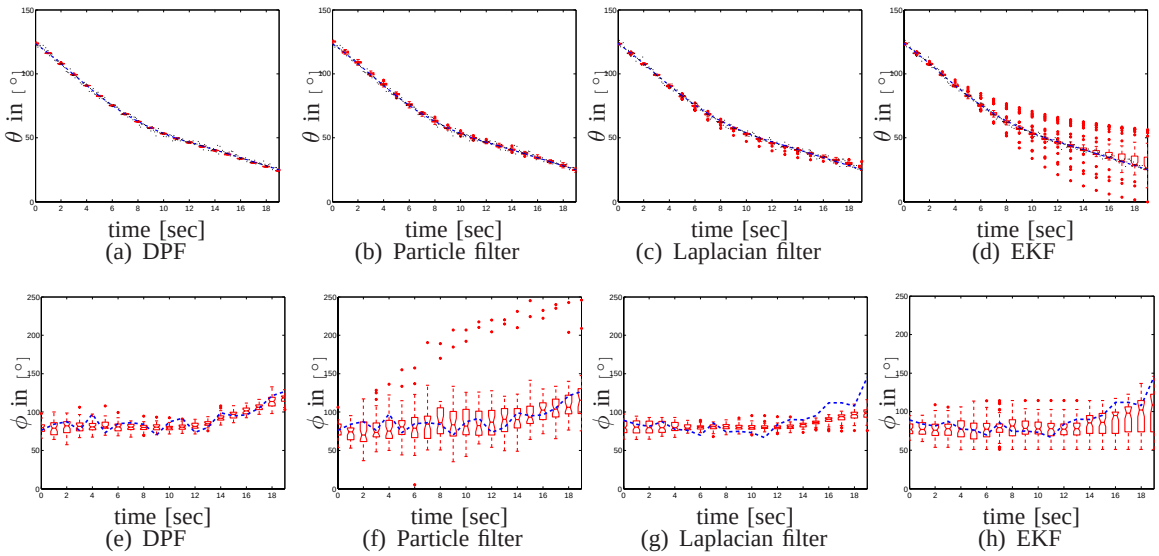


Fig. 14: Monte-Carlo comparison of the DOA filters with the DPF filter presented in [4].

[11] Y. Bar-Shalom, "Tracking methods in a multitarget environment," *IEEE Trans. Automatic Control*, vol. AC-23, pp. 618–626, Aug. 1978.

[12] K.C. Chang and Y. Bar-Shalom, "Joint probabilistic data association for multitarget tracking with possibly unresolved measurements," *IEEE Trans. Automatic Control*, vol. AC-29, pp. 585–594, July 1978.

[13] L. Ng and Y. Bar-Shalom, "Multisensor multitarget time delay vector estimation," *IEEE Trans. ASSP*, vol. ASSP-34, pp. 669–677, Aug. 1986.

[14] R. Karlsson and F. Gustafsson, "Monte Carlo data association for multiple target tracking," in *IEE Target Tracking: Algorithms and Applications*, Netherlands, 16-17 Oct. 2001.

[15] D. Crisan and A. Doucet, "A survey of convergence results on particle filtering methods for practitioners," *IEEE Trans. on Signal Processing*, vol. 50, no. 3, pp. 736–746, March 2002.

[16] J. MacCormick and M. Isard, "Partitioned sampling, articulated objects, and interface-quality hand tracking," in *Proceedings of the Europen Conference on Computer Vision*, 2000.

[17] M. Isard and J. MacCormick, "BraMBLe: A Bayesian multiple-blob tracker," in 8^{th} *International Conference on Computer Vision*, 2001.

[18] L. Tierney and J. B. Kadane, "Accurate approximations for posterior moments and marginal densities," *Journal of the American Statistical Association*, , no. 81, pp. 82–86, 1986.

[19] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, Chapman Hall/CRC, 2004.

[20] V. Cevher and J. H. McClellan, "Fast initialization of particle filters using a modified Metropolis-Hastings algorithm: Mode-Hungry approach," in *ICASSP 2004*, Montreal, CA, 17–22 May 2004.

[21] D.H. Johnson and D.E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, Prentice Hall, 1993.

[22] P. Stoica and A. Nehorai, "Music, maximum likelihood, and Cramér-Rao bound," *IEEE Trans. on ASSP*, vol. 37, no. 5, pp. 720–741, May 1989.

[23] W. Ng, J. P. Reilly, T. Kirubarajan, and R.-R. Larocque, "Wideband array signal processing using mcmc methods," *IEEE*

Trans. on Signal Processing, vol. 53, no. 2, pp. 411–426, Feb. 2005.

[24] Y. Boers and J. N. Driessen, "Multitarget particle filter track before detect application," *IEE Proc. Radar, Sonar, and Navigation*, vol. 151, no. 6, pp. 351–357, Dec. 2004.

[25] M. G. Rutten, B. Ristic, and N. J. Gordon, "A comparison of particle filters for recursive track-before-detect," in *8th International Conf. on Info. Fus.*, July 2005, vol. 1, pp. 169–175.

[26] H. Wang and M. Kaveh, "On the performance of signal-subspace processing-part II: Coherent wide-band systems," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-35, pp. 1583–1591, Nov. 1987.

[27] A.B. Gershman and M.G. Amin, "Wideband direction-of-arrival estimation of multiple chirp signals using spatial time-frequency distributions," in *Proceedings of the Tenth IEEE Workshop on Statistical Signal and Array Processing*, 29 Oct.-1 Nov. 2000, pp. 467–471.

[28] A.B. Gershman, M. Pesavento, and M.G. Amin, "Estimating the parameters of multiple weideband chirp signals in sensor arrays," *IEEE Signal Processing Letters*, vol. 7, no. 6, pp. 152–155, June 2000.

[29] A. Papoulis and S.U. Pillai, *Probability, random variables and stochastic processes*, McGraw Hill, 2002.

[30] J. MacCormick and A. Blake, "A probabilistic exclusion principle for tracking multiple objects," in 7^{th} *International Conference on Computer Vision*, 1999, pp. 572–578.

[31] M. Evans, N. Hastings, and B. Peacock, *Statistical Distributions, 3rd ed.*, Wiley, 2000.

[32] T. Edgoose, L. Allison, and D. L. Dowe, "An MML classification of protein sequences that knows about angles and sequences," in *Pacific Symp. Biocomputing 98*, Jan. 1998, pp. 585–596.

[33] Z. Khan, T. Balch, and F. Dellaert, "An MCMC-based particle filter for tracking multiple interacting targets," Tech. Rep. GIT-GVU-03-35, College of Computing, Georgia Institute of Technology, Oct. 2003.

[34] V. Cevher and J. H. McClellan, "An acoustic multiple target tracker," in *IEEE SSP 2005*, Bordeaux, FR, 17–20 July 2005.

[35] G. Kitagawa, "Monte-Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Comp. and Graph. Stat.*, vol. 5, no. 1, pp. 1–25, Mar 1996.

[36] M Bolić, P. M. Djurić, and S. Hong, "New resampling algorithms for particle filters," in *ICASSP*, 2003.

[37] S. Mallat and S. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[38] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1092, 1953.

[39] W.K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[40] S. Chib and E. Greenberg, "Understanding the Metropolis-Hastings algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995.

[41] L. Tierney, "Markov chains for exploring posterior distributions," *The Annals of Statistics*, vol. 22, no. 4, pp. 1701–1728, 1994.

[42] Samuel S.Blackman, *Multiple-Target Tracking with Radar Application*, Artech House, 1986.

[43] M.K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, pp. 590–599, 1999.