A Multi Target Bearing Tracking System using Random Sampling Consensus

Volkan Cevher[1], Faisal Shah[2], Rajbabu Velmurugan[2], and James H. McClellan[2]
[1]Center for Automation Research
University of Maryland,
College Park, MD 20742
[2]School of Electrical and Computer Engineering
Georgia Institute of Technology
777 Atlantic Drive
Atlanta, GA 30332-0250, USA
{volkan}@cfar.umd.edu, {faisal.shah, rajbabu, jim.mcclellan}@ece.gatech.edu

Abstract—In this paper, we present an acoustic direction-of-arrival (DOA) tracking system to track multiple maneuvering targets using a state space approach. The system consists of three blocks: beamformer, random sampling, and particle filter. The beamformer block processes the received acoustic data to output bearing batches as point statistics. The random sampling block determines temporal clustering of the bearings in a batch to determine region-of-interests (ROIs). Based on the *track-before-detect* approach, each ROI indicates the presence of a possible target. We describe three random sampling algorithms called RANSAC, MSAC, and NAPSAC to use in the random sampling block. The particle filter then tracks the targets via its interactions with the beamformer and the random sampling blocks. We present a computational analysis of the random sampling blocks and show tracking results with field data.

TABLE OF CONTENTS

1 INTRODUCTION 1
2 TRACKER DESIGN 2
3 BEAMFORMER BLOCK 4
4 RANDOM SAMPLING FOR REGION-OF-INTEREST PROCESSING 4
5 PARTICLE FILTER 7
6 SIMULATIONS 9
7 CONCLUSIONS 12
8 ACKNOWLEDGEMENTS 12

1. INTRODUCTION

In this paper, we discuss automated tracking of the direction-of-arrival (DOA) angles of multiple targets using an acoustic array in the presence of noise or interferers [1–6]. We present three random sampling algorithms to temporally cluster bearing observations to form region-of-interests (ROI) to be used by a particle filter tracker. The ROI processing is becoming a popular method for trackers based on the *track-before-detect* concept [7–10].

The ROI processing idea relies on the observation that temporally consistent data, e.g., bearings, indicate the presence of actual targets, when we assume that the clutter observations are uniformly distributed in the state space. Hence, during tracking, each new ROI in the observed data can be used to address a fundamental issue for target tracking algorithms in a statistical framework: initialization. Unfortunately, forming ROIs is a set covering problem, which is NP-complete [11–13].

Hence, we present three robust and efficient random sampling algorithms for ROI processing: RANSAC for random sampling consensus, MSAC for m-estimator sampling consensus, and NAPSAC for N-adjacent points sample consensus. These algorithms are commonly applied to motion segmentation problems in computer vision [14–22]. We show the application of these algorithms to our multi target bearing tracking problem. In addition, we discuss their computation complexity and analyze their statistical behavior.

We also present a particle filter algorithm to track the DOA's of multiple targets, using an acoustic node that contains an array of microphones with known positions. Each particle in the filter is created by concatenating the state vector for each target, called a partition. For example, the partition of the kth target has a state that consists of the DOA $\theta_k(t)$ and the DOA rate $\dot{\theta}_k(t)$ of the kth target. The total number of targets (or partitions) K is determined by a random sampling block based on the ROI processing. 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 within an estimation period of duration T.

In a particle filter, where the observations arrive in sequence, the state probability density function is represented by dis-

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.

crete state samples (particles) distributed according to the underlying distribution (as explained by the state space) either directly or by proper weighting [23, 24]. 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 [25]. To improve the efficiency of the algorithm, various methods are proposed, such as the partitioning approach [2, 26], or other Bayesian approaches [27].

Our particle filter uses multiple DOA's to determine the state vector, based on an image template matching idea. We denote the DOA collection as a batch. In our problem, a DOA image is first formed when a batch of DOA observations are received from a beamformer that processes the received acoustic data at τ second intervals. Then, image templates for target tracks are created using the state update function and the target partition state vectors. By determining the best matching template (e.g., 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 paper is organized as follows. Section 2 explains the mechanics of the automated tracking system for multiple target DOA tracking. Sections 3, 4, and 5 elaborate on the individual elements of the tracking system. Computer simulations are shown in Sect. 6 to demonstrate the performance of the algorithms.

2. Tracker Design

We construct the tracking system mechanics (i) to compensate DOA estimation biases due to rapid target motion [3], (ii) to result in higher resolution DOA estimates than just beamforming [2–4], (iii) to be robust against changes in target signal characteristics, and (iv) to automatically determine the number of targets. The tracker system consists of three blocks as illustrated in Fig. 1. Details of the individual blocks are given in the following sections. Below, we discuss the elements of this design.

We 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 [28]. This is intuitive, because these methods estimate an average of the target angular spread during their estimation periods [3].

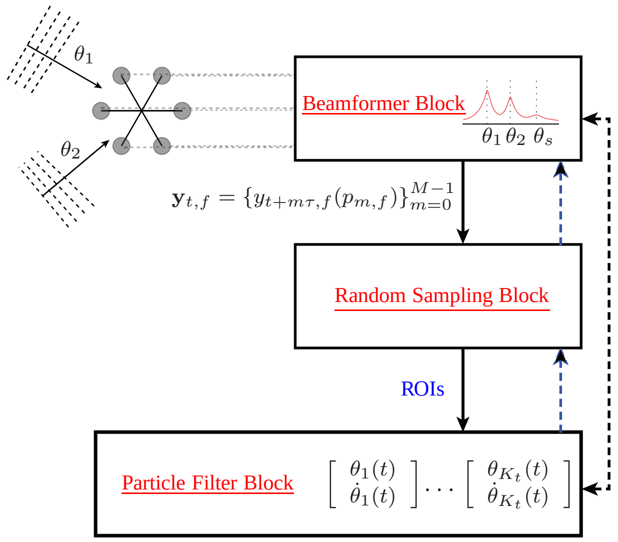

θ_1
θ_2
Beamformer Block
$\theta_1\,\theta_2\,\theta_s$
$\mathbf{y}_{t,f} = \{y_{t+m\tau,f}(p_{m,f})\}_{m=0}^{M-1}$
Random Sampling Block
ROIs
Particle Filter Block
$\begin{bmatrix} \theta_1(t) \\ \dot{\theta}_1(t) \end{bmatrix} \cdots \begin{bmatrix} \theta_{K_t}(t) \\ \dot{\theta}_{K_t}(t) \end{bmatrix}$

Figure 1. Tracker mechanics is demonstrated using three basic blocks. The beamformer block monitors the received acoustic signals to adapt to their frequency characteristics for better bearing estimation. It calculates a batch of DOA's that form point statistics for the random sampling and particle filter blocks. The random sampling block searches for temporal clusters in the bearing batch to determine region-of-interests (ROIs). Consistent ROIs are declared as targets by the particle filter block. The particle filter estimates the tracking posterior and can make various inferences (i.e., mean and mode estimates).

In general, locally linear motion models eliminate this bias by simultaneously estimating the bearing with the target motion parameters for the estimation period of T. Conceptually, this 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 the high volume of the acoustic data used for estimation. In this paper, we propose to use a beamformer block in our system to buffer the variability in the observed acoustic signals to create a compressed and invariant point statistics for our tracker block [5, 29].

Therefore, the beamformer block (Fig. 1) processes the acoustic data, sampled at F_s, at smaller intervals of $\tau = T/M$ (e.g., $M = 10$), where the targets are assumed to be relatively stationary (Fig. 2). The parameter M is called the batch size. This reduces the number of acoustic data samples available for processing, resulting in a sequence of noisier DOA estimates. However, these noisier DOA estimates are smoothed in the particle filter block, because a motion structure is imposed on the batch of DOA's.

We use the target bearing θ and the bearing rate $\dot{\theta}$ in our state vector to model the kth target ($k = 1, \ldots, K_t$, where K_t is the number of targets at time t) motion to illustrate the ideas in this paper:

$$x_k(t) = \begin{bmatrix} \theta_k(t) \\ \dot{\theta}_k(t) \end{bmatrix}, \tag{1}$$

where the DOA's are measured counterclockwise with respect to the x-axis. The state vector for the particle filter is a concatenation of these target *partitions*:

$$\mathbf{x}_t = \begin{bmatrix} x_1^T(t) & x_2^T(t) & \dots & x_{K_t}^T(t) \end{bmatrix}^T. \tag{2}$$

The state update function is locally constant velocity model for each target on its bearing:

$$x_k(t+T) = h_T\left(x_k(t)\right) + u_k(t), \tag{3}$$

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

$$h_T(x_k(t)) = \begin{bmatrix} \theta_k(t) + \dot{\theta}_k(t)T \\ \dot{\theta}_k(t) \end{bmatrix}. \tag{4}$$

In [4, 5], we discuss other state models to characterize motion. For our state vector, a batch of DOA's when $M \geq 2$ is sufficient for the state-observability [3, 30]. Since the filter is built on the compressed statistics which is also sufficient for the state vector, it achieves a significant reduction in computational requirements.

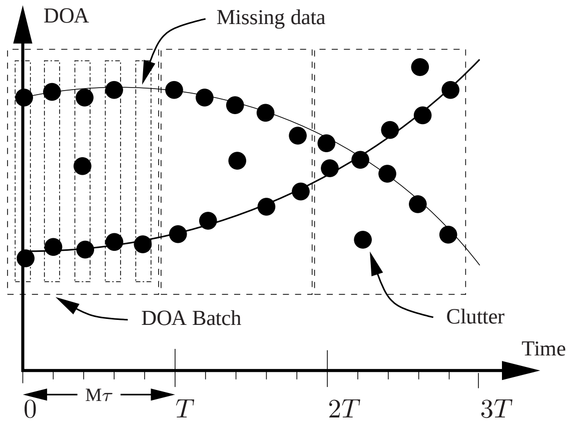


Figure 2. Observation model uses a batch of DOA measurements, generated by the beamformer block. Note that the DOA measurements are not necessarily ordered when they are output from the beamformer. However, when the batch of DOA's are treated as a bearing vs. time image, there is a natural ordering, which alleviates the multiple target tracking by the particle filter.

When the received signal characteristics change, the particle filter tracker formulation is not affected because the beamformer block absorbs 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 [28, 31]. The tracker in [4] tries to adapt to varying time-frequency characteristics of the target

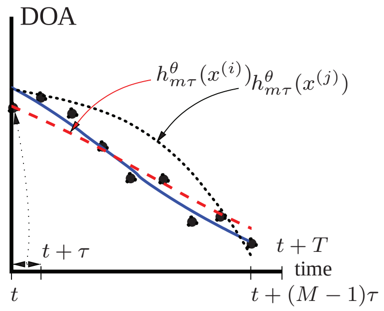


Figure 3. The random sampling and particle filter blocks use the state update function to create target track templates. In the figure, the solid line represents the true DOA track. Black dots represent the noisy DOA estimates. The dashed line and the dotted line represent the DOA tracks for the two proposed particles i and j. These tracks are calculated using the state update function h. Visually, ith particle is a better match than the jth particle; hence, its likelihood is higher.

signals, assuming that the varying frequencies are narrowband. The tracker in [6] 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 rework of the filter equations would be required to track targets with e.g., wideband signal characteristics (e.g., from [6] to [32]).

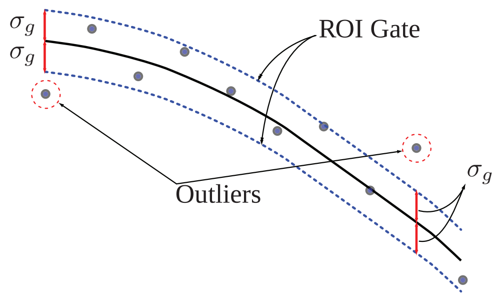


Figure 4. The random sampling block uses a gating operation around the created target track templates. Solid line is the true DOA track. The DOA observations are shown with dots. Given a gate size σ_g, most DOA's should fall into the gate of the target track to declare a ROI.

The second block in Fig. 1 addresses a fundamental issue for any target tracker: initialization and target detection. It is a random sampling block that efficiently localizes region-of-interests (ROI) in the calculated bearing batches to determine the number of targets at each time. This block takes the bearing batches from the beamformer block and determines K_t through gating operation based on the target motions, as illustrated in Figs. 3 and 4. The random sampling block determines the ROI for each target *one at a time*, similar to the matching pursuit idea [33]. That is, it first converges on a 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 (targets) for the particle filter, which can delete its own partitions. Conceptually, this initialization idea here is equivalent to *track-before-detect* approach used in the radar community [9, 10].

The final block is an independent partition particle filter tracker that uses the generated ROI's by the random sampling plot to propose particles for efficiency [2]. Our particle filter does not use a Markov random field to penalize the state update function as the targets approach each other (target interactions), since the target bearing may cross even if the targets are not close in proximity. While calculating the particle weights, the tracker uses the bearing batch data from the beamformer block directly. The filter data-likelihood employs the joint probability density assignment approach when targets are close to each other to better address their data interactions [34, 35]. Note that the filter avoids any direct data association thanks to the ROI processing by the random sampling block.

The particle filter block does not constrain the number of targets K_t that can be tracked by the filter. However, the number of targets that can be tracked is theoretically bounded by the number of microphones in the acoustic array [28]. Figure 1 also illustrates feedback paths (dashed lines) for guided sampling or beamforming. Although not fully explored in this paper, we note that when the number of targets increase (e.g., K=10), the particle filter tracker can provide guidance for the random sampling and the beamformer blocks for further computational efficiency.

3. 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 [28]. 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 [28, 31]. Beamformers are wideband if they are designed for target signals with broadband frequency characteristics [36]. There are also other beamformers that are designed for signals with time-varying narrow-band frequency characteristics [37, 38].

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 with $P_{m,f}$-highest DOA peaks of the beamformer pattern (Fig. 2). 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. 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.

Finally, the choice of the parameter τ for beamforming is determined by various physical constraints, including (i) target frequency range, (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 $\tau < \frac{2D}{\dot{\theta}_{\max}}$, where $\dot{\theta}_{\max}$ is the maximum allowed bearing rate. Lastly, the target motion should satisfy the constant velocity assumption during the output period T. For ground targets, $T = 1$s is a reasonable choice. Note that at least two 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

$$\frac{2}{F_{\min}} < \tau < \min\left\{\frac{2D}{\dot{\theta}_{\max}}, \frac{T}{M}\right\}. \tag{5}$$

4. RANDOM SAMPLING FOR REGION-OF-INTEREST PROCESSING

A region-of-interest (ROI) is defined as a cluster of any measurement sequence in time that is likely to be generated by a target (Fig. 5). The number of ROIs in a bearing batch is a strong indicator of the number of targets. In this section, we describe the application of three random sampling methods to determine the number of targets in bearing data batches, based on ROI processing. These three methods are named RANSAC for random sampling consensus [14], MSAC for m-estimator sampling consensus [19], and NAPSAC for N adjacent points sample consensus [22]. We will show that these algorithms provide more efficient ways of clustering the data to initiate ROIs, when compared to other clustering approaches applied to similar tracking problems [7, 8].

Determining the ROIs using the bearing batches as point statistics becomes a challenging problem in the face of outliers and missing bearing observations. To solve this problem efficiently, each of the random sampling procedures use a small number of bearing estimates as feasible and enlarges this set with consistent data as much as possible while iterating. These algorithms are successfully used in many computer vision problems for geometric motion segmentation, model order selection, and multiple structure data regression [14–20]. In this section, we show how to apply these algorithms for tracking multiple targets using point statistics.

In general, every possible sub set of data needs to be con-

[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].

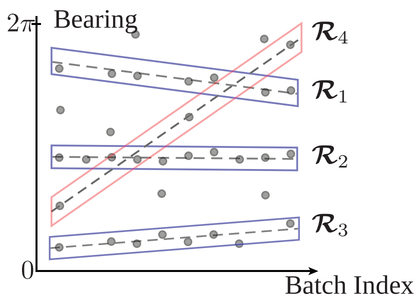


Figure 5. The circles are the DOA estimates in a batch. We illustrate four ROIs in the figure $\mathcal{R}_i$ ($i = 1, \ldots, 4$). The first three ROIs show a clustering of DOA sequences in time that are likely to be generated by targets, when we assume that the target bearings slowly change along the batch index. The fourth ROI $\mathcal{R}_4$ is not likely to be a target since it has lesser number of bearings in its gate than the other ROIs and it covers a much larger bearing range in a short amount of time.

sidered to determine a ROI. In the literature, this problem is known as the set covering problem [11–13]. If we have P peaks for each batch of size M, then $\omega = \binom{MP}{M}$ correspondences need to be considered for determining a ROI with M samples. Furthermore, if there are K targets, then a total of $\Omega = \binom{\omega}{K} \approx \omega^K$ (by Sterling's formula) correspondences also need to be considered. This calculation is further complicated, when some ROIs have missing data. For our problem:

- We first consider the minimum sufficient number of data points and obtain a feasible solution. Let L denote the minimum number of bearing samples required to determine the state vector x_k using (3). By extending the solution using only L-points, we decrease the search size to $\omega = \binom{MP}{L}$.
- We note that determining ROIs is temporal estimation problem, hence we only seek temporal correspondences. Therefore, we only need to consider $\omega = P^L \times \binom{M}{L}$.
- When the targets are separated in the angle space, then we can determine ROIs one at a time by removing the bearing points corresponding to the determined ROI without effecting the data of the other targets.
- When targets are close and they are already being tracked, the tracking information can be used to guide the ROI sampling.
- When targets are close and they are not being tracked, no specific identity is assigned to either target. Hence, an identity confusion during initialization is assumed to have no cost and can be corrected by the tracker by incorporating more temporal data.

With our assumptions, a greedy combinatorics algorithm can determine multiple targets with $\Omega^* = P^L \times \binom{M}{L} \times K$ complexity. Below, we show that we can decrease the computational complexity of a greedy combinatorial algorithm by an additional order of magnitude by allowing ourselves to miss a ROI at a fractional probability. Note that, the probability of missing the same target over a many consecutive batches exponentially goes to zero, as we gather more data. We further discuss the failure modes of the random sampling algorithms and also suggest improvements later.

RANSAC

Introduced by Fischler and Bolles in [14], RANSAC is a hypothesis generation and verification algorithm for robust estimation. It repeatedly generates solutions using a minimal set of data points and then tests the support of each solution from the complete set of correspondences. For RANSAC, the number of correspondences within the gate of the proposed solution (i.e., track template in Figs. 3, 4, and 5) defines the support of the putative correspondences.

The gate size σ_g determines the match acceptance and is usually chosen in a conservative way to minimize the number of incorrect mismatches. Note that the matching process is based only on proximity and and similarity. Hence, it is quite likely to have many mismatches in the presence of clutter, missed data, and presence of multiple targets, which are sufficient to render any least squares estimator incapacitated.

To determine our state vector in (1), we only need $L = 2$ bearing estimates since two points are sufficient to determine a line on a $2D$-plane. Given a putative solution that is determined by randomly chosen L-points ($L \ll M$), RANSAC algorithm counts the set of *inliers* that fall into the gate of the track template. Intuitively, if one of the chosen points is an *outlier*, the line will not gain much support (e.g., $\mathcal{R}_4$ in Fig. 5). However, if the chosen L-points are inliers (e.g., any two points within $\mathcal{R}_i$, $i = 1, 2, 3$), a ROI is determined from only one iteration.

Let I be the number of RANSAC iterations required to determine a ROI with probability p. Let us assume that in the data there is only one target and the rest of the data are outliers. Let ϵ denote the proportion of outliers and hence $q = 1 - \epsilon$ denote the probability that any selected point is an inlier. Then, after I selections, the probability that all the selections are outliers are given by

$$1 - p = (1 - q^L)^I. \tag{6}$$

Hence, the number of iterations required to pick an inlier set is given by

$$I = \frac{\log(1-p)}{\log\left(1 - (1-\epsilon)^L\right)}. \tag{7}$$

Table 1 determines the number of RANSAC iterations required to ensure that a ROI is picked with $p = .99$ probability. In this example, we have $M = 10$ batch samples with $P = 10$ peaks, $L = 2$, and there is only one target.

To compare the complexity of the RANSAC's operation, assume that there is no missing bearings in the target track, and hence there are 10 bearings corresponding to the target and

Table 1. RANSAC Iterations I for $p = .99$

Sample size	Proportion of Outliers ϵ								
L	50%	55%	60%	65%	70%	75%	80%	85%	90%
2	17	21	27	36	49	72	113	203	459
3	35	49	70	106	169	293	574	1363	4603
4	72	110	178	305	567	1177	2876	9095	46050
5	293	553	1123	2503	6315	18861	71954	404293	4605168

we have 90 outliers, corresponding to $\epsilon = .90$. An exhaustive search over all the bearings require $\left(\begin{array}{c}10\\2\end{array}\right) \times 10^2 = 4500$ cases to search, whereas RANSAC only considers 459 cases to determine the ROI with $p = .99$ probability. This corresponds to an order of magnitude improvement in computation. Moreover, when the targets have missing bearings, the combinatorial approaches need to further process the clusters to link them temporally [7, 8]. On the other hand, RANSAC declares a ROI even with missing bearings since the support is not affected much by a few missing bearings. The pseudo code for the RANSAC algorithm is given in Table 2.

Table 2. RANSAC Algorithm

```
1. Randomly select $L$ data points from the bearing batch data and
   instantiate a solution $x_k$ using (3).
2. Determine the number of inlier points $n$ that are within the $\sigma_g$
   gate of the solution (for visualization, refer to Fig. 4).
3. If the number of inliers are above some threshold $\alpha$, e.g.,
   $\alpha = M - 1$, re-estimate the solution $x_k$ using all the inlier points
   using (3) and terminate.
4. If the number of inliers are less than the threshold $\omega$, re-select a
   new set of $L$-points randomly and repeat above.
5. After $I$ trials, use the data set with the number of largest
   consensus inlier set, and estimate the solution $x_k$ by using (3).
```

A few remarks are in order here. Note that the number of iterations I depends on the proportion of the outliers and not the actual number of outliers. Therefore, the required computation is still manageable even if the number of outliers is large. Moreover, increasing the number of samples used to instantiate a solution x_k more than the number of the minimal set is outweighed by the severe increase in the computation and is not desired. Note that once the inlier set is extended, the actual solution is found by using all the inliers.

To further save computation, it is natural to terminate the RANSAC loop if the number of inlier points are above some threshold, denoted as ω. This threshold can be judiciously set as $\alpha = \lceil \epsilon^* M \rceil$, where $\epsilon^* > .5$ is the estimate of the worst case proportion of the outliers. The worst case scenario is that we have only one target and the rest of the data belongs to clutter. Hence, if we have $P = 8$ beamformer peaks and $M = 10$ batch size, then $\epsilon^* = 7/8$, and $\alpha = \lceil 7/8 \times 10 \rceil = 9$ can be used to stop.

RANSAC implicitly assumes that the outliers form a uniform distribution in the bearing space. This is clearly not the case when we have multiple targets. However, the presence of multiple targets actually implies that it is more likely to find one target in one application of RANSAC to the batch when we notice that we are not looking for a specific target at any given application of RANSAC. Determination of any one target suffices. When we find a ROI in the batch, we delete the bearings corresponding to the ROI and rerun RANSAC on the residual batch data. In this subsequent iteration of RANSAC, the number of outliers with respect to the number of targets is reduced and hence it is easier to find a new target, if there is any left.

A confusion issue arises when the bearing tracks of two targets are in close proximity: i.e., crossing or parallel bearing data. We illustrate two cases for two targets crossing in bearings, where RANSAC may result in two incorrect ROIs $\boldsymbol{\mathcal{R}}_1'$ and $\boldsymbol{\mathcal{R}}_2'$. Most of the times, RANSAC may resolve these cases. However, it is difficult to determine when it does not. During tracking, a guidance from the particle filter is used to resolve such cases. We further discuss how we can guide the sampling algorithms when we discuss NAPSAC. These cases are not important during initialization, since there is no need to assign an identity during initialization.

RANSAC may result in a false alarm when the clutter points are aligned according to the state update function in (3). For our state vector, this happens when the clutter points approximately form a line. Analytical derivation of the probability that random points in the bearing batch forms a line can be done. However, it is cumbersome and does not provide any additional insight. In the simulations section, we show that the false alarm probability is approximately linear with the gate size σ_g. It also increases exponentially as the number of beamformer peaks P increase and decreases exponentially as the number of batch samples M increase.

Finally, the output of RANSAC is well suited for the particle filter. The algorithm outputs multiple ROIs with detection probabilities κ, false alarm probabilities (corresponding to a spurious ROI), and the miss data estimate in each ROI. Hence, the RANSAC output can also be used in the proposal function stage of the particle filter to better sample the multi target posterior distribution.

MSAC

By using the redescending M-estimator [19, 21], a modest computational benefit can be achieved by using MSAC: m-estimator sample consensus. MSAC determines ROIs by minimizing the following cost function:

$$J_{\text{MSAC}} = \sum_{\text{all points } i} \rho(e_i^2), \text{ where } \rho(e^2) = \left\{ \begin{array}{ll} e^2, & e^2 < \sigma_g^2; \\ \sigma_g^2, & e^2 \geq \sigma_g^2. \end{array} \right. \tag{8}$$

and e is the distance of the points to the putative solution.

MSAC does not explicitly determine any inlier set until after its execution is over. Hence, its computational complexity

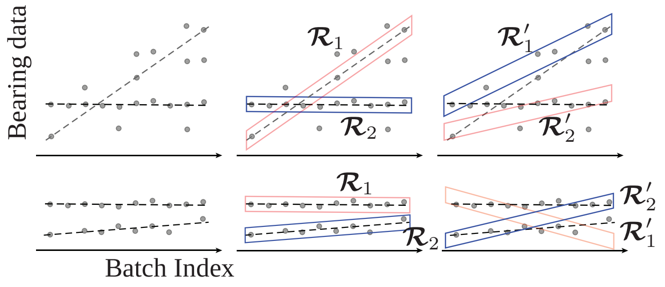


Figure 6. When the bearing tracks of two targets approach each other, RANSAC may confuse the ROIs. Most of the times, RANSAC can correctly initialize the ROIs. However, sometimes the tracks are confused. In those cases, track information from the particle filter can be used to resolve the confusion, while sampling the ROIs.

is less than RANSAC, which specifically determines an inlier set and records its count. However, where we can use an intuitive counting condition to stop RANSAC's execution, the MSAC stopping condition needs to use statistical arguments. Note that the cost function is a summation of a chi-squared distribution with M degrees-of-freedom, and a sum of squared-uniform random variables. Although it is possible to analytically determine a threshold corresponding to a given confidence level, we also need to consider missing data cases. Instead, we use the RANSAC's stopping condition, based on counting inliers, for MSAC in the simulations. The pseudo code for the MSAC algorithm is given in Table 3

Table 3. MSAC Algorithm

```
1. Randomly select $L$ data points from the bearing batch data and
   instantiate a solution $x_k$ using (3).
2. Determine the cost function $J_{\text{MSAC}}$ using (8). Also determine the
   number of inliers as in RANSAC.
3. If the number of inliers are above some threshold $\alpha$, e.g.,
   $\alpha = M - 1$, re-estimate the solution $x_k$ using all the inlier points
   using (3) and terminate.
4. If the number of inliers are less than the threshold $\omega$, re-select a
   new set of $L$-points randomly and repeat above.
5. After $I$ trials, use the data set with the number of largest
   consensus inlier set, and estimate the solution $x_k$ by using (3).
```

NAPSAC

In RANSAC or MSAC, we randomly sample L points to instantiate a solution. When the number of data points is large or when we would like to sample multiple targets *jointly*, e.g., two at a time, the probability of sampling a set of inliers decreases exponentially due to increase in the dimensionality of the sampling problem. NAPSAC algorithm specifically addresses these issues by pseudo-randomly generating its hypotheses. NAPSAC stands for N-adjacent points sample consensus [22].

When we have multiple targets in the bearing batches, the corresponding target bearings lie on manifolds that are more likely to adjacent to other inlier points in a ROI than to the outliers or pseudo-outliers, corresponding to other ROIs. Hence, if the first selected point is an inlier, the data points that are close to it have higher probabilities of being inliers. On the other hand, if an outlier point is selected, then the adjacent points are *less* likely to be inliers. However, if, in fact, there are K targets in the bearing batch, then there will be at least K regions, where sampling close to the initial random sample improves over an unbiased random selection. This advantage comes at the expense of more calculations.

Table 4. NAPSAC Algorithm

```
1. Randomly select 1-data point from the bearing batch data. Sort
   all the remaining points in increasing Eucledian distance from this
   point.
2. Use the absolute value of a Gaussian distribution with zero mean
   and a standard deviation $k\sigma_g$, where $k$ is a value between $M/4$
   and $M/2$. Select a new point in the sorted list that is closest to
   this random variable.
3. Repeat until $L$ points are chosen and instantiate a solution $x_k$
   using (3).
4. Count the number of inliers (or alternatively determine the cost
   function $J_{\text{MSAC}}$ using (8)).
5. If the number of inliers are above some threshold $\alpha$, e.g.,
   $\alpha = M - 1$, re-estimate the solution $x_k$ using all the inlier points
   using (3) and terminate.
6. If the number of inliers are less than the threshold $\omega$, re-select a
   new set of $L$-points as above.
7. After $I$ trials, use the data set with the number of largest
   consensus inlier set, and estimate the solution $x_k$ by using (3).
```

In Table 4, we list the pseudo code for NAPSAC. On line 2, we show how to prioritize samples based on their proximity to the originally chosen random point on line 1. This strategy provides a guidance based on proximity. Other generic guidance methods exist using RANSAC for computer vision problems [39]. For our specific problem, when we are looking for a new ROI corresponding to a target that is already being tracked, we can exploit the tracking information. This is achieved by weighting points according to their distances to the updated track information. This way, the points that are closer to the proposed track are chosen more often than others while determining a ROI. Even if the target maneuvers, its ROI can be easily easily determined since we are choosing the points randomly based on proximity.

5. Particle Filter

In this section, the details of the particle filter block in Fig. 1 are discussed. We first discuss the observation equation and then describe the joint probabilistic density association principle.

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. 2). 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. An additional partition dependency is also allowed, i.e., κ_k^f, since RANSAC may have a different detection probability when it detects the targets one at a time using a constant number of iterations.[2]

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) [28]. 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 [35, 40]. 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 that the beamformer power vs. angle pattern has smoothness properties, we use the following pdf for the spurious peaks:

$$p(\theta|\theta \text{ is spurious}) = \frac{\gamma}{2\pi}, \tag{9}$$

where $\gamma > 1$ 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 (9) implies that the natural space (or similarly volume) 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 [35]. Similar arguments for active contour tracking that is relevant to this paper are found in [41]. 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.

[2] In addition, recognition/identification may have an impact on choice of the acoustic frequencies. Hence, some targets may have a lower detection probability at the recognized target frequencies. The partition dependence of the detection probability can address this issue. It also allows the particle filter to guide the beamformer block for improved detection.

Define a set $\mathcal{I}_n$ that consists of n-unordered combination of all K-partitions of the particle filter state vector: $\mathcal{I}_n \in \{_K\mathbb{C}_n\}$, where $_K\mathbb{C}_n$ is the set of all n-unordered outcomes from K possibilities. Define $_K\mathcal{C}_n$ as the number of elements of the set $_K\mathbb{C}_n$. Hence, 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:

$$\pi_{0,1}(\mathbf{y}_{m,f}) = \left(\frac{\gamma}{2\pi}\right)^{P_{m,f}} \tag{10}$$

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,

$$\pi_{n,j}(\mathbf{y}_{m,f}) = \frac{(P_{m,f} - n)!\,\left(\gamma/2\pi\right)^{P_{m,f}-n}}{P_{m,f}!} \times \sum_{p_1 \neq p_2 \neq \ldots \neq p_n}^{P_{m,f}} \prod_{i=1}^{n} \psi_{t,m,f}\left(p_i \middle| \hat{x}_i\right), \tag{11}$$

where $\hat{x}_i$ is in $_n\mathbf{x}_t(j)$, and the function ψ is derived from the assumption that the associated target DOA's form a Gaussian distribution around the true target DOA tracks:

$$\psi_{t,m,f}\left(p_i \middle| x_i\right) = \frac{1}{\sqrt{2\pi}\sigma_\theta^2(m,f)} \exp\left\{ -\frac{\left(h_{m\tau}^\theta(x_i(t)) - y_{t+m\tau,f}(p_i)\right)^2}{2\sigma_\theta^2(m,f)} \right\}, \tag{12}$$

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 (12) is not a proper circular distribution for an angle space. For angle spaces, the von Mises distribution is used as a natural distribution [42]. 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 (12) [43]. Because the von Mises distribution has numerical issues for small DOA variances, the Gaussian approximation (12) is used in this paper. Hence, special care must be taken in the implementation to handle angle wrapping issues.

The Gaussian in (11) $\psi(\cdot|\cdot)$ 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:

$$\begin{aligned}\pi_{2,3}(\mathbf{y}_{m,f}) &\propto \sum_{p_1=1}^{P_{m,f}} \sum_{p_2=1,p_1\neq p_2}^{P_{m,f}} \psi_{t,m,f}\left(p_1\middle|\hat{x}_1\right)\psi_{t,m,f}\left(p_2\middle|\hat{x}_2\right)\\ &\propto \sum_{p_1=1}^{P_{m,f}} \sum_{p_2=1,p_1\neq p_2}^{P_{m,f}} \psi_{t,m,f}\left(p_1\middle|x_2\right)\psi_{t,m,f}\left(p_2\middle|x_3\right).\end{aligned} \tag{13}$$

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. Note that Eqn. (13) 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:

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{f=1}^{F}\prod_{m=0}^{M-1}\sum_{n=0}^{K}\frac{\kappa^f_{n,K}}{{}_K\mathcal{C}_n}\sum_{j=1}^{{}_K\mathcal{C}_n}\pi_{n,j}(\mathbf{y}_{m,f}). \tag{14}$$

In Eqn. 14, 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 are provided by the random sampling block. Moreover, 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, the particle filter can provide a guidance, by using the current state estimates, to the beamformer and the random sampling blocks to resolve those cases.

Particle Filter Proposal Function

To demonstrate the performance of the system, we use only the state update function as the proposal function of the particle filter. We propose each target partition independently to cope with the curse of dimensionality in sampling high dimensions. Note that once the proposal function is formulated, the rest of the particle filter structure is well-defined: weighting and resampling. In weighting stage, the approximate posterior distribution is used when targets are sufficiently apart. When they are close, we use the joint posterior.

Algorithm Details

Pseudo-code of the particle filter algorithm is given in Table 5. The filter implementation employs an efficient resampling strategy, named "deterministic resampling", first outlined by Kitagawa [44]. 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 [44]. Faster resampling schemes without convergence proofs are also available [45] and these could make a difference in the filter computation, especially when $K = 1$.

Table 5. 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 $i = 1, 2, \dots, N$
     • For $k = 1, 2, \dots, K$
     sample $x^{(i)}_k(t) \sim p(x^{(i)}_k(t)|x^{(i)}_k(t-T))$.
     • Form $\mathbf{x}^{(i)}_t = \left[\, x^{(i)}_1(t),\ x^{(i)}_2(t), \dots,\ x^{(i)}_K(t)\right]^T$.
2. Calculate the weights
     $w^{*(i)}_t = w^{(i)}_{t-T} p(\mathbf{y}_t|\mathbf{x}^{(i)}_t)$,
     where $p(\mathbf{y}_t|\mathbf{x}^{(i)}_t)$ is fully joint observation density, given by Eqn. (14).
3. Normalize the weights:
     $w^{(i)}_t = \frac{w^{*(i)}_t}{\sum_i w^{*(i)}_t}$.
4. Make estimation: $E\{f(\mathbf{x}_t)\} = \sum_{i=1}^N w^{(i)}_t f(\mathbf{x}^{(i)}_t)$.
5. Resample the particles:
     • Heapsort the particles in a ascending order according to their
     weights: $\mathbf{x}^{(i)}_t \rightarrow \tilde{\mathbf{x}}^{(i)}_t$.
     • Generate $\omega \sim \mathcal{U}[0,1)$.
     • For $j = 1, 2, \dots, N$
   a. $u^{(j)} = \frac{j-\omega}{N}$,
   b. Find $i$, satisfying $\sum_{l=1}^{i-1}\tilde{w}^{(i)}_t < u^{(j)} \le \sum_{l=1}^{i}\tilde{w}^{(i)}_t$,
   c. Set $\mathbf{x}^{(j)}_t = \tilde{\mathbf{x}}^{(i)}_t$.
```

Finally, the partitions are managed by the specific interaction between the particle filter and the random sampling block. New partitions are introduced into the particle filter, using the distribution supplied by the random sampling block. The particle filter deletes partitions at either the proposal stage or after estimation, when there are not enough bearings within the gate of the mode estimate. 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.

6. SIMULATIONS

Demonstration of the Random Sampling Block

Figures 7-13 show the performance of the random sampling algorithms under varying conditions. Each figure is created by a Monte-Carlo run of size 1000. Based on these figures, the authors subjectively rank the algorithms in the following order in usefulness: 1) RANSAC, 2) MSAC, and 3) NAPSAC. Although we were expecting better performance from NAPSAC, the assumptions of NAPSAC are not matched by our data. MSAC performs arguably better than RANSAC in terms of root-mean-square error and false alarm probability. However, its results are not as intuitive as RANSAC, which is based on the number of inliers as opposed to their m-estimator statistics. In the next subsection, we show how to extend the RANSAC idea to cope with bearing observations corresponding to multiple frequencies.

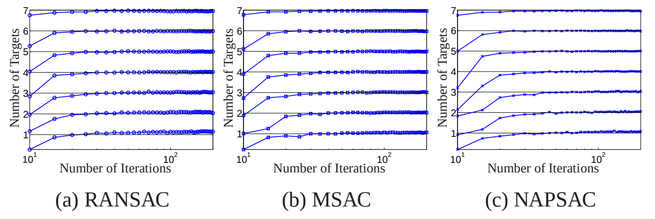


Figure 7. The number of iterations to converge to the true number of targets is shown for (a) RANSAC, (b) MSAC, and (c) NAPSAC.

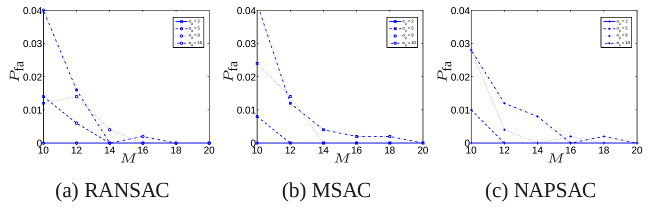


Figure 9. The false alarm rate of each of the algorithms exponentially decay to zero as we increase the number batch samples M.

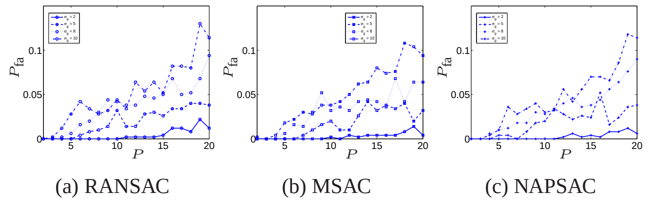


Figure 10. As we increase the number of peaks P, we allow more clutter into the random sampling algorithms. Hence, the false alarm rate shows an increasing trend as P increases. In this example, the threshold for declaring a target is $M - 1$, where $M = 10$. Hence, the false alarm rates are lower than the ones shown in Fig. 8.

Field Data Results

A uniform circular acoustic array with 8 microphones with 1 meter radius is used to collect the acoustic data for a five vehicle convoy at the Aberdeen Proving Grounds. The acoustic data sampling rate was $F_s = 1024$Hz. The convoy consisted of two military Hummers (HMMV) and three commercial sports utility vehicles (SUV), traveling on gravel on an oval track. Detection and tracking of the commercial vehicles presented a difficult challenge because the commercial vehicles were in between the two louder military vehicles, hence they were acoustically suppressed (Fig. 14). Hence, this presented an opportunity to test our tracking system. Our results for different beamformer outputs are shown in Fig. 15.

One of the main problems while processing the field acoustic data is getting reliable DOA estimates in multi target scenarios. We observed that the beamformer would only output

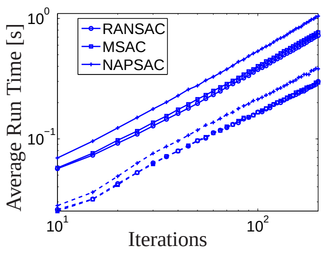


Figure 11. We compare the average run times of each of the algorithms with (solid lines) and without a stopping condition (dashed lines) in their iterations. Without the stopping condition, the algorithm runs a predetermined number of iterations. For the solid lines, we let the algorithms run a full 200 iterations. For the dashed lines, the algorithms stopped themselves when they found a ROI with number of inliers greater than or equal to $M - 1$ with $M = 9$. The stopping condition allows us to save computation with no penalty. Also, the average run times exponentially increases with the number of iterations.

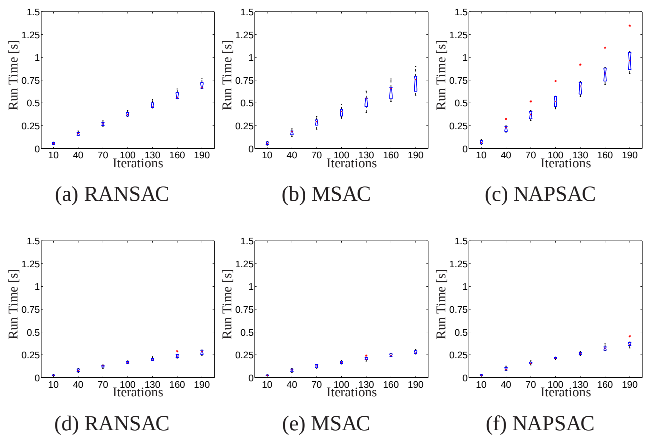


Figure 12. The variation of the run times are demonstrated. (a-c) Without stopping condition. (d-f) With stopping condition.

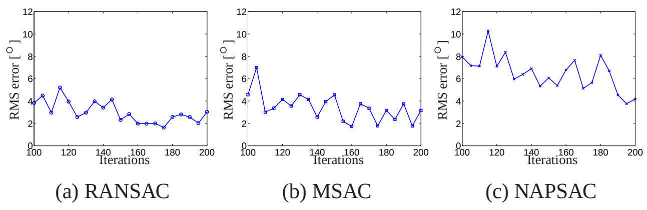
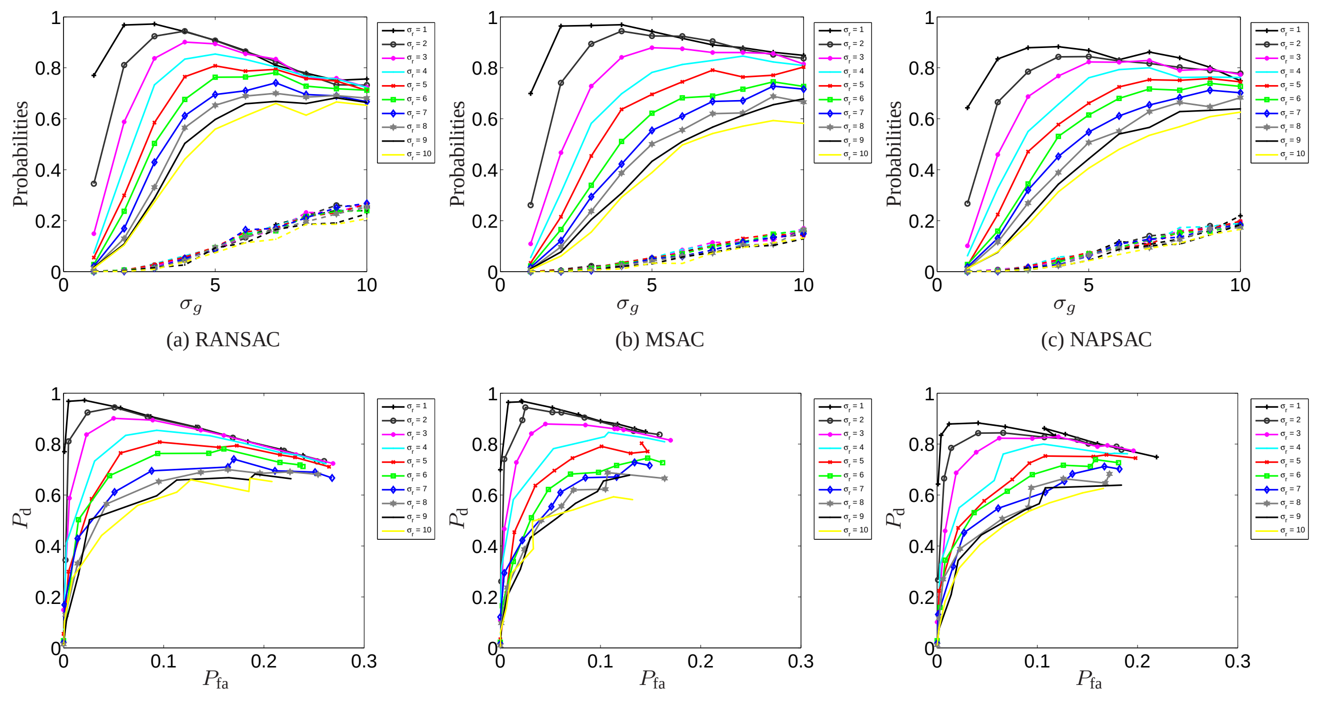


Figure 13. The estimation performance of the random sampling algorithms are compared. It is seen that RANSAC and MSAC performs similarly. NAPSAC performs the worst.



(a) RANSAC (b) MSAC (c) NAPSAC

(d) RANSAC (e) MSAC (f) NAPSAC

Figure 8. (a-c) The detection probability (solid lines) and the false alarm probabilities (dashed lines) are shown for the random sampling algorithms for varying gate size σ_g. Note that for this particular case, the bearing data noise is changed between 1 to 10 degrees to show the effect of the model mismatches. Our threshold for accepting a ROI is at least $M/2 + 1 = 6$ peaks. (d-f) The receiver operating characteristics (ROC) is shown. In general, ROC curves are always concave and non-decreasing by construction. Hence, some of the ROC curves in the plots exhibit uncharacteristic behaviors. Possible reasons are the mismatch of the gate size and the bearing variance, and the presence of clutter.

DOA estimates corresponding to a subset of the targets. After careful investigation, it was determined that the acoustic signatures of the weak targets were being suppressed by the louder ones. To detect more targets, we modified the beamformer to use up to 10 frequencies and to also select several peaks in the steered frequency response. The beamformer also used multiple snapshots of data. These modifications result in the ability to detect silent targets; however, it substantially increased the amount of clutter.

We modified the random sampling block to cope with the presence of multiple observations per target. Moreover, since the number of peaks could change from one data set to another, we formulated adaptive thresholds to accept the RANSAC ROI output and declare a target. A number of measures are taken. The first measure is to put a minimum bound on the number of batch indexes, in which inliers are found. This is to ensure the DOA estimates are spread over the entire batch of data as one would expect from an actual target. A value of 85% of the number of batch indexes is used. The next measure is to put a lower limit on the total number of inliers found by the RANSAC block. Hence, we assume that the maximum number of targets our system would track is less than the number of microphones in the acoustic array and compute the number of DOA estimates per target in a scenario with the maximum number of targets. The third measure is an upper limit on the gradient of the line found by RANSAC. After taking into consideration real world constraints of vehicle motion and observing the field data, it was determined that a change in more than 6 degrees per batch index would not be considered a target. Similar thresholds were used to decide when to delete targets.

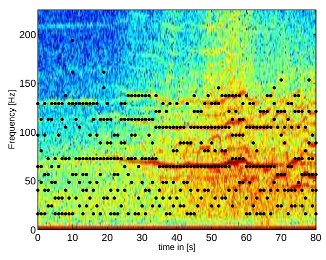


Figure 14. Time-frequency plot of the acoustic signal is shown. The dots are the narrow-band frequencies used by the beamformer block to determine candidate target bearings.

Our system was able to robustly deal with the clutter and correctly track the convoy as can be seen in Fig. 15. There are a few spurious targets due to the persistence of the beamformer sidelobes. However, these sidelobes have a well understood characteristics and it is possible to detect targets that correspond to the power leakage due to the sidelobes. For the particle filter, we used 2500 particles since we did not incorporate the ROI processing in our proposal stage. As future work, we will incorporate the ROI processing to increase the efficiency of our particle tracker. The gate size is judiciously chosen to be 9 degrees. The gate size can also be chosen adaptively by using a bank of random sampling blocks. We are currently investigating how to statistically and automatically infer the results of such random sampling banks.

7. Conclusions

In this paper, we demonstrated the application of RANSAC, MSAC, and NAPSAC algorithms for processing region-of-interests to track multiple targets using bearing measurements. We showed that each of these algorithms are well suited for ROI processing in conjunction with a tracker. Among the random sampling algorithms, we determined that RANSAC is the most useful because of its performance and its flexibility to also handle multi frequency target tracking case. Our tracking results show significant promise in the ROI processing algorithms proposed in this paper since a simple bootstrap particle filter could handle a difficult convoy scenario.

8. Acknowledgements

We would like to thank Dr. Rama Chellappa for his help and comments in the preparation of this manuscript.

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] V. Cevher, R. Velmurugan, and J. H. McClellan, "Acoustic multi target tracking using direction-of-arrival batches," to appear IEEE Transactions on Signal Processing, available at http://www.umiacs.umd.edu/users/volkan/publications.htm.

[6] 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.

[7] W. Ng, J. Li, S. Godsill, and J. Vermaak, "A Hybrid Approach for Online Joint Detection and Tracking for Multiple Targets," in *Aerospace, 2005 IEEE Conference*, 2005, pp. 1–16.

[8] W. Ng, J. Li, S. Godsill, and J. Vermaak, "Multiple target tracking using sequential Monte Carlo methods and efficient data association and initialisation," *IEE Proceedings Radar, Sonar & Navigation*, 2004.

[9] 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.

[10] 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.

[11] J. E. Beasley and et al., "A genetic algorithm for the set covering problem," *European Journal of Operational Research*, vol. 94, no. 2, pp. 392–404, 1996.

[12] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

[13] D. S. Hochbaum, *Approximation algorithms for NP-hard problems*, PWS Publishing Co. Boston, MA, USA, 1996.

[14] M.A. Fischler and R.C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[15] P. H. S. Torr, "Geometric motion segmentation and model selection," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 356, no. 1740, pp. 1321–1340, 1998.

[16] H. Chen, P. Meer, and D. E. Tyler, "Robust regression for data with multiple structures," in *2001 IEEE Conference on Computer Vision and Pattern Recognition*, 2001, pp. 1069–1075.

[17] C. V. Stewart, "Bias in robust estimation caused by discontinuities and multiplestructures," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 8, pp. 818–833, 1997.

[18] H. Wang and D. Suter, "Robust adaptive-scale parametric model estimation for computer vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1459–1474, 2004.

[19] P. H. S. Torr and A. Zisserman, "MLESAC: a new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[20] R. Hartley and A. Zisserman, *Multiple View Geometry in computer vision*, Cambridge University Press, 2003.

[21] P. J. Huber, "Robust estimation of a location parameter,"

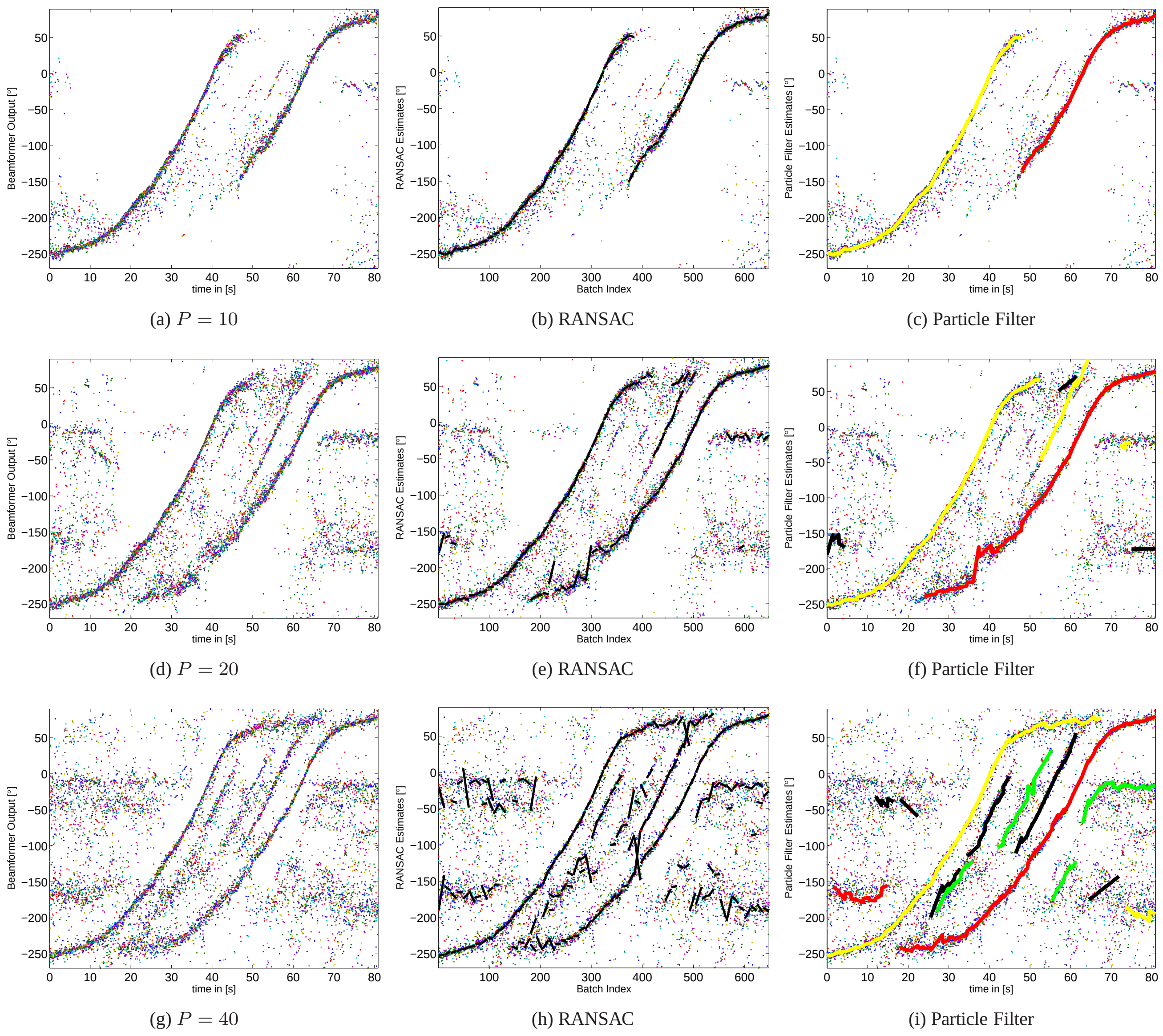


Figure 15. (a) The MUSIC beamformer results are shown for the 10 strongest frequencies, where we only picked the highest peak in each power vs. angle pattern. Corresponding RANSAC (b) and particle filter results (c) for the bearing batches. (d) The MUSIC beamformer results are shown for the 5 strongest frequencies. This time, we used the 4 highest peaks in each power vs. angle pattern. Corresponding RANSAC (e) and particle filter results (f) for the bearing batches. (c) The MUSIC beamformer results are shown for the 10 strongest frequencies. This time, we used the 4 highest peaks in each power vs. angle pattern. Corresponding RANSAC (h) and particle filter results (i) for the bearing batches. Note that the actual number of targets is 5.

The Annals of Mathematical Statistics, vol. 35, pp. 73–101, March 1964.

[22] D. R. Myatt, P. H. S. Torr, S. J. Nasuto, J. M. Bishop, and R. Craddock, “Napsac: High noise, high dimensional robust estimation-its in the bag,” in *British Machine Vision Conference, Cardiff, UK*, 2002, pp. 458–467.

[23] 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.

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

[25] 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.

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

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

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

[29] R. Mahler, L. D. Syst, and M. N. Eagan, “Nonadditive probability, finite-set statistics, and informationfusion,” in *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, 1995, vol. 2.

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

[31] 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.

[32] 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.

[33] 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.

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

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

[36] 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.

[37] 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.

[38] 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.

[39] B. Tordoff and D. W. Murray, “Guided Sampling and Consensus for Motion Estimation,” in *Proceedings of the 7th European Conference on Computer Vision-Part I*. 2002, pp. 82–98, Springer-Verlag London, UK.

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

[41] 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.

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

[43] 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.

[44] 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.

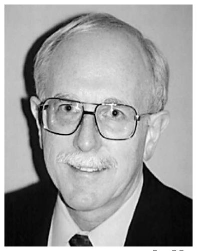
[45] M. Bolić, P. M. Djurić, and S. Hong, “New resampling algorithms for particle filters,” in *ICASSP*, 2003.

Volkan Cevher *was born in Ankara, Turkey, in 1978. He received his B.S. degree in Electrical Engineering from Bilkent University, Ankara, Turkey in 1999 as a valedictorian. During summer of 2003, he was employed by Schlumberger Doll Research. In Fall 2004, he was the co-recipient of the Center for Signal and Image Processing Outstanding Research Award. He received his Ph.D. degree in Electrical Engineering from Georgia Institute of Technology in 2005. He worked a post-doc at Georgia Institute of Technology under the supervision of Dr. James H. McClellan till the end of 2005. He is currently working with Dr. Rama Chellappa as a research associate on computer vision problems. His research interests include structure from motion, sensor network management problems (sensor build and placement strategies), Monte-Carlo Markov chain methods (specifically particle filters), target tracking models, adaptive filters, time frequency distributions, fractional Fourier transform, brain-computer interface problems, and array signal processing.*

Faisal Shah *was born in Hancock, Michigan. He received his B.S. and M.S. degrees in Electrical Engineering from Georgia Institute of Technology in 2004 and 2006 respectively. He is currently pursuing his Ph.D. degree at Georgia Institute of Technology under the advisement of Prof. James H. McClellan. His research interests include particle filtering, target tracking, and array signal processing.*

Rajbabu Velmurugan *received his B.E. degree from Government College of Technology, Coimbatore, India in 1995 and M.S. degree from Clarkson University, Potsdam, USA in 1998, both in electrical engineering. He worked in Larsen & Toubro Ltd., India before joining Clarkson University and in The*

MathWorks from 1998 to 2001. He is currently pursuing his Ph.D. degree at Georgia Institute of Technology under the advisement of Prof. James H. McClellan. His research interests include particle filtering techniques, design and implementation of real-time signal processing systems, and array signal processing. He is also interested in teaching and involved in projects related to signal processing education.



James H. McClellan *received the B.S. degree in Electrical Engineering from L.S.U. in 1969, and the M.S. and Ph.D. degrees from Rice University in 1972 and 1973, respectively. From 1973 to 1982, he was a member of the research staff at Lincoln Laboratory and then a professor at MIT. From 1982 to 1987, Dr. McClellan was employed by Schlumberger Well Services. Since 1987, he has been a Professor in the School of Electrical and Computer Engineering at Georgia Tech, where he presently holds the John and Marilu McCarty Chair.*
He is a co-author of the texts Number Theory in Digital Signal Processing, Computer Exercises for Signal Processing, DSP First: A Multimedia Approach, *and* Signal Processing First, *which received the McGraw-Hill Jacob Millman award for an outstanding innovative textbook in 2003. In 1998, Prof. McClellan received the W. Howard Ector Outstanding Teacher Award at Georgia Tech, and in 2001, the Education Award from the IEEE Signal Processing Society. In 1987, he received the Technical Achievement Award for work on FIR filter design, and in 1996, the Society Award, both from the IEEE Signal Processing Society. In 2004, he was a co-recipient of the IEEE Jack S. Kilby Signal Processing medal. Prof. McClellan is a Fellow of the IEEE and a member of Tau Beta Pi and Eta Kappa Nu.*