CONCURRENT DETECTION AND TRACKING FOR GMTI

Ross Deming, consultant, Anteon Corporation, USA
Leonid Perlovsky, Air Force Research Laboratory, USA

Abstract—We develop a technique for performing multitarget tracking from GMTI radar data using dynamic logic (DL). The approach is specifically designed for target-rich, high-clutter, scenarios in which state-of-the-art trackers will have impractically high computational complexity during data association. DL provides a way to perform data association without combinatorial explosion, so that the complexity scales only linearly with additional data and/or targets. Thus, since we can operate with large amounts of clutter, we can push down the pre-detect thresholds and track in very low signal to clutter ratio environments. We begin by introducing parameter estimation for generic Gaussian mixtures. We then proceed to show that multi-target tracking using DL is a straightforward variation of this generic case. We present tracking results based upon realistic synthetic data at S/C as low as -6dB in both Doppler and RCS. This represents several orders of magnitude improvement over the rule-of-thumb required S/C for existing trackers.

I. INTRODUCTION

A sensor’s ability to provide information on the movement of objects of interest depends on its ability to initiate and maintain simultaneous tracks on multiple targets. This task is difficult for low RCS, low Doppler, targets and/or when multiple targets are closely spaced relative to the accuracy of the system. Here, the crux is data association, that is, how to automatically assign each detection sample to its corresponding target or to the background clutter. In highly cluttered, target-rich, situations, the data association problem can cause standard trackers to become impractical due to unreasonably high computational complexity, as we describe below. In contrast, the tracker described in this paper performs data association in an efficient manner, and does not suffer from a computational explosion. In principle, since our tracker can handle a high level of false detections, we can therefore push down the detection thresholds, and realize a significant improvement in terms of signal-to-clutter ratio over existing trackers.

The extended Kalman filter (EKF) [1]-[3] is probably the most popular framework for tracking problems. Here, the motion of the target is modeled by a discrete-time state transition equations. The EKF then provides a recursive solution for updating the target state as new measurements are acquired. More recently, sequential Monte Carlo methods, a.k.a. “particle filters”, have appeared as an alternative to the EKF-based methods [4]. Particle filters which, like the EKF, are based upon stochastic state equations, generalize the traditional Kalman Filter methods to situations involving non-linear state and measurement equations and non-Gaussian noise.

Whether an EKF or particle filter is used, the foremost problem for multiple target tracking in clutter is data association, i.e. how to assign each data sample to a certain track or clutter class. Multiple hypothesis tracking (MHT) [5] provides an optimum framework for data association. However, since it involves an evaluation of every possible mapping between data and tracks, the computational complexity scales exponentially with the number of targets and data samples. Thus, MHT becomes impractical for scenarios with many targets and/or many clutter samples [6]-[8]. There are various approximations (e.g. pruning) that can be made to MHT to reduce the computational burden, however by using these shortcuts the approach becomes suboptimal. Joint probabilistic data association (JPDA) [3] is another popular method for performing data association which is more efficient than MHT because one only needs to evaluate the association probabilities separately at each time step. It should be noted that JPDA involves track maintenance but not track initialization. Therefore, since detection is performed separately from tracking, JPDA is not optimal [8] and cannot initiate tracks at low signal-to-clutter (S/C) ratios. In contrast, MHT is optimal, but impractical since the number of mappings between data and targets will grow exponentially. It should also be noted that in the higher-clutter cases we consider in this paper, both JPDA and MHT would have combinatorial complexity, especially in cases with multiple sensor platforms.

The method we consider in this paper is based upon an alternative approach to MHT based upon dynamic logic (DL) [7]-[10]. A major difference between this approach and JPDA is that detection (i.e. track initiation), data association, and tracking are performed concurrently. Concurrent processing allows the tracker to approach ideal performance [8] and allows tracking in high clutter since no threshold is needed for detection. Here, the probability density of the data samples (including range plus azimuth plus classification features and, if available, Doppler) is modelled as a mixture. Each component of the mixture corresponds to a different target, and model parameters describe the kinematics of the target trajectories. Then, an iterative method related to expectation-maximization (EM) is used to efficiently hill climb in the space of all parameters and all possible mappings between data samples and targets. Computational complexity for the dynamic logic approach scales linearly with the number of targets, data samples, and sensors, which represents a significant improvement over both MHT and JPDA. Thus, since we can operate with large amounts of clutter, we can...
push down the pre-detect thresholds and track in very low signal to clutter ratio environments. Incidentally, DL provides a means to model the clutter distribution concurrently with data association, etc. DL is different than other approaches based upon mixture models, as described in [7].

DL has previously been applied to multi-target tracking problems in radar and sonar processing [7], [9] for the case of a stationary sensor platform. This paper discusses a new implementation of dynamic logic to multi-target tracking in which there can be multiple, moving, sensor platforms, and therefore the measurement error covariance can be different for each time sample. Thus, the parameter estimation equations require a modification, which we derive here. We also present new results pertaining to this more complicated platform geometry.

In Section II we lay the groundwork for the tracker by discussing parameter estimation for generic Gaussian mixtures. Then, in Section III, we proceed to show that multi-target tracking is a relatively simple variation of parameter estimation for generic Gaussian mixtures. Finally, in Section IV, we present some sample results based upon synthetic data.

II. PARAMETER ESTIMATION FOR GENERIC GAUSSIAN MIXTURES.

We begin by describing maximum-likelihood (ML) parameter estimation for a generic Gaussian mixture. The results given in this section are not new, however, they will serve as a basis to develop the multi-target tracking theory in the next section.

Mathematically, the Gaussian mixture model is represented by the equation

$$ p(X|\Theta) = \sum_{k=1}^{K} E_k p(X|\Theta_k), $$

where each mixture component (class) $k$ has the normal (Gaussian) distribution

$$ p(X|\Theta_k) = \frac{1}{\sqrt{(2\pi)^D|C_k|}} e^{-\frac{1}{2}(X-M_k)^T C_k^{-1}(X-M_k)}. $$

Here, $X = [x\ y\ z\ \ldots]^T$ is vector of data in multi-dimensional feature space, $\Theta = (\Theta_1, \Theta_2, \ldots, \Theta_K, E_K)$ is the total set of parameters for the mixture, $\Theta_k = \{M_k, C_k\}$ is the subset of mean and covariance parameters for component $k$, and $E_k$ is the mixture weight for component $k$. Throughout the discussion, vectors and matrices are indicated in boldface, and the superscript $^T$ denotes the vector or matrix transpose. Since $E_k$ denotes the fraction of the total samples that belong to class $k$ it is, by definition, equivalent to the a priori probability $P(k)$ for class $k$. It is a well-known fact that these probabilities should sum to one, i.e.

$$ \sum_{k=1}^{K} P(k) \equiv \sum_{k=1}^{K} E_k = 1. $$

All components of the mixture need not be purely Gaussian. For example, some components can be Gaussian in $x$ but uniform in $[y\ z]$. Background clutter in tracking problems is typically modelled with uniform components, or combinations of uniform and Gaussian components, as we discuss further in the next section. A uniformly-distributed component is described by the equation

$$ p(X|\Theta_k) = \frac{1}{V}, $$

where $V$ is the area or volume in $X$ of the region of interest.

We now introduce the well-known log-likelihood function

$$ LL(\Theta) = \sum_{n=1}^{N} \ln p(X_n|\Theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} E_k p(X_n|\Theta_k). $$

Here, the subscript $n = (1, 2, 3, \ldots, N)$ denotes the sample index, so that $X_n = [x_n\ y_n\ z_n\ \ldots]^T$ is the data vector corresponding to sample $n$. If you are trying to define a pdf model that best represents the sampled data, $LL$ gives a quantitative measure of the “goodness of fit”. One must adjust the model parameters in order to maximize $LL$, while incorporating the constraints in Eq.(3). The details of this optimization are given, for example, in [7], where we find that by setting to zero the partial derivatives of $LL$ with respect to each of the parameters, it is found that the maximum-likelihood parameters for the generic Gaussian mixture exactly satisfy the equations

$$ \hat{E}_k = \frac{1}{N} \sum_{n=1}^{N} P(k|n), $$

$$ \hat{M}_k = \frac{\sum_{n=1}^{N} P(k|n)X_n}{\sum_{n=1}^{N} P(k|n)}, $$

and

$$ \hat{C}_k = \frac{\sum_{n=1}^{N} P(k|n) (X_n - \hat{M}_k)(X_n - \hat{M}_k)^T}{\sum_{n=1}^{N} P(k|n)}, $$

where

$$ P(k|n) = \frac{E_k p(X_n|\Theta_k)}{\sum_{j=1}^{K} E_j p(X_n|\Theta_j)}. $$

Since Eq.(9) is actually a form of Bayes’ rule, $P(k|n)$ can be interpreted as the association probability, i.e. the probability that sample $n$ belongs to component (class) $k$. Eqs. (7) and (8) are intuitively pleasing in the sense that they are precisely the usual definitions for the sample mean and covariance, except that here each sample is weighted by the association probability $P(k|n)$. Note, for uniform components, Eq.(6) is still valid for estimating $\hat{E}_k$, while Eqs.(7) and (8) are not necessary.

Eqs. (6)-(8) make it appear that the parameters $\hat{\Theta} = \{\hat{E}_k, \hat{M}_k, \hat{C}_k\}$ can be computed directly, in a straightforward manner. However, if these equations are considered more carefully, they are found to be a messy set of coupled nonlinear equations, because the quantity $P(k|n)$ on the right-hand side is a function of all the unknown parameters $\Theta$, as
can be seen from Eqs. (9) and (2). If a person could solve these equations directly, they would yield the exact values for the maximum-likelihood parameters – unfortunately a direct solution is intractable. However, we can compute the parameters indirectly, by iterating back and forth between the parameter estimation equations (6)-(8) and the probability estimation equation (9).

In other words, the parameter estimates \( [E_k^{(I)}, M_k^{(I)}, C_k^{(I)}] \) in iteration \( I \) are computed using the association probabilities \( P^{(I-1)}(k|n) \) computed from the previous iteration \( (I-1) \). Next, the association probabilities \( P^{(I)}(k|n) \) in iteration \( I \) are updated using the parameter estimates \( [E_k^{(I)}, M_k^{(I)}, C_k^{(I)}] \) from iteration \( I \), and so on. Mathematically, this procedure corresponds to a recursive set of equations, analogous to Eqs. (6)-(9), which are

\[
\begin{align*}
\hat{E}_k^{(I)} &= \frac{1}{N} \sum_{n=1}^{N} P^{(I-1)}(k|n), \\
\hat{M}_k^{(I)} &= \frac{\sum_{n=1}^{N} P(k|n)^{(I-1)} X_n}{\sum_{n=1}^{N} P^{(I-1)}(k|n)}, \\
\hat{C}_k^{(I)} &= \frac{\sum_{n=1}^{N} P^{(I-1)}(k|n) (X_n - \hat{M}_k^{(I)})(X_n - \hat{M}_k^{(I)})^T}{\sum_{n=1}^{N} P^{(I-1)}(k|n)},
\end{align*}
\]

where

\[
P^{(I)}(k|n) = \frac{E_k^{(I)} p(X_n|\Theta_k^{(I)})}{p(X_n|\Theta^{(I)})} = \frac{E_k^{(I)} p(X_n|\Theta_k^{(I)})}{\sum_{j=1}^{K} E_j^{(I)} p(X_n|\Theta_j^{(I)})}.
\]

To initialize the procedure, parameter values are assigned at random, or by other means. Then the above set of equations is repeated until convergence, typically on the order of 25–100 iterations. In practice, one can decide whether convergence has been obtained by plotting the log-likelihood vs. iteration number and looking for a levelling off in the curve. The procedure is guaranteed to converge, in the sense that \( LL \) will never increase between successive iterations, as proven in reference [7]. However, convergence to an extraneous local minimum is sometimes possible. If this error occurs, and can be detected, the user must repeat the procedure after choosing a new set of initial conditions.

For convenience, Eqs. (6)-(8) can be expressed in an equivalent short-hand notation if we make the definition

\[
\langle * \rangle_k \equiv \sum_{n=1}^{N} P(k|n) (*).
\]

Then

\[
\begin{align*}
\hat{E}_k &= \frac{1}{N} \langle 1 \rangle_k, \\
\hat{M}_k &= \frac{\langle X_n \rangle_k}{\langle 1 \rangle_k}, \\
\hat{C}_k &= \frac{\langle (X_n - \hat{M}_k)(X_n - \hat{M}_k)^T \rangle_k}{\langle 1 \rangle_k},
\end{align*}
\]

and

\[
\text{III. Multi-target Tracking.}
\]

Now, the parameter estimation equations for generic Gaussian mixtures are modified in a straightforward manner to treat multi-target tracking. This problem has already been solved in [7], Chapter 7.2, for the simpler case in which the sensor is stationary during data collection. In the analysis below, we generalize this technique to allow sensor positions to change significantly during data collection, and we also allow sensor accuracy to be different for each sample for both range and cross-range (and, if available, Doppler). Thus, we have a very flexible model that can be used to combine the measurements from multiple, flying, sensors. Also this same model can be used for different sensor types, for example radar or electro-optical, we just need to specify the appropriate sensor accuracy.

We express the pdf of the data using the standard mixture model of Eqs. (1) and (2). However, we can rewrite this model by separating \( X \) into several types of component elements; (i) the vector of classification features \( Z \) (e.g., target RCS or length), (ii) the coordinate vector \( R = [x \ y]^T \) (\( x = \text{north} \) and \( y = \text{east} \)), and (iii) the Doppler measurements \( D \). Thus, the model can be segmented as follows:

\[
X = \begin{bmatrix} Z \\ R \\ D \end{bmatrix}, \quad M_k = \begin{bmatrix} M_{zk} \\ M_{Rk} \\ m_{Dk} \end{bmatrix}, \quad C_k = \begin{bmatrix} C_{zk} & 0 & 0 \\ 0 & C_{Rn} & 0 \\ 0 & 0 & \sigma_D^2 \end{bmatrix},
\]

where \( M_{Rk} = [m_{zk} \ m_{yk}]^T \). The matrix parameter \( C_{Rn} \) is determined by the sensor accuracy, and describes an “error ellipse” on the map. The tilt of this ellipse depends upon the direction of the line-of-sight vector from the sensor to the target. Because the sensor position is allowed to vary significantly, we allow the ellipse to be different for each measurement \( n \), and therefore \( C_{Rn} \) is a function of \( n \). Since the sensor position can be arbitrary, data from multiple, moving, sensors is incorporated seamlessly.

Next, we rewrite the equation for the model components \( p(X|\Theta_{zk}) \), using the fact that \( Z, R, \) and \( D \) are independent random variables, as shown by the block-diagonal form of the \( C_k \) matrix above. Thus, the equation for target components can be rewritten in the factored form

\[
p(X|\Theta_{zk}) = p(Z|\Theta_{zk}) p(R|\Theta_{Rk}) p(D|\Theta_{Dk}),
\]

where

\[
p(Z|\Theta_{zk}) = \frac{1}{\sqrt{(2\pi)^D |C_{zk}|}} e^{-\frac{1}{2} (Z-M_{zk})^T C_{zk}^{-1} (Z-M_{zk})},
\]

\[
p(R|\Theta_{Rk}) = \frac{1}{2\pi \sqrt{|C_{Rn}|}} e^{-\frac{1}{2} (R-M_{Rk})^T C_{Rk}^{-1} (R-M_{Rk})}
\]

\[
p(D|\Theta_{Dk}) = \frac{1}{\sqrt{(2\pi)^D \sigma_D^2}} e^{-\frac{1}{2} (D-m_{Dk})^2}.
\]

The clutter components are slightly different than Eq.(18). For example we can make them uniform over the coordinate vector \( R \) so that

\[
p(X|\Theta_{zk}) = \frac{1}{\sqrt{|p(Z|\Theta_{yk})|}} p(D|\Theta_{Dk}),
\]
where \( V \) is the area (north/east) of the region of interest.

In this paper we use a constant-velocity model for target trajectories, which will be fairly accurate for high revisit rates. More complicated models (constant acceleration, link-track, etc.) can also be incorporated in a straightforward manner, as discussed in [7], [9]. Using the constant-velocity model and Gaussian noise model, it is easy to show that the noisy position measurements are distributed around the mean parameter

\[
M_{Rk} = \begin{bmatrix} m_{rk} \\ m_{yk} \end{bmatrix} = \begin{bmatrix} x_{0k} + \dot{x}_kt \\ y_{0k} + \dot{y}_kt \end{bmatrix} \tag{19}
\]

Here, \( \{x_{0k}, y_{0k}\} \) are the time-zero coordinates in north/east, while \( \{\dot{x}_k, \dot{y}_k\} \) are the velocities, and \( t \) denotes time. The Doppler measurements \( D \), if available, are noisy measurements of the target velocity projected onto the line-of-sight vector. These measurements are normally distributed around the mean parameter \( m_{DK} \), which is related to the true target velocity \( \{\dot{x}_k, \dot{y}_k\} \) via

\[
m_{DK} = \dot{x}_k \cos \phi_n - \dot{y}_k \sin \phi_n, \tag{20}
\]

where \( \phi_n \) is the angle that the line-of-sight vector makes counterclockwise relative to the \( x \) = north coordinate axis.

“Tracking” consists of estimating \( x_{0k}, y_{0k}, \dot{x}_k, \dot{y}_k \) for each target \( k = 1, 2, 3, ..., K \). A by-product is the estimation of mean and covariance parameters \( M_{zk} \) and \( C_{zk} \) governing the distribution of classification features for each target \( k \).

Parameter estimation is performed by maximizing \( LL \), as in the generic case, by setting partial derivatives to zero. The resulting equations for \( E_k, M_{zk}, \) and \( C_{zk} \) are identical to Eqs.(15)-(17) for the generic mixture (after substituting \( Z \) for \( X \)). For the tracking parameters, the equations are coupled, and a bit more complicated. First we introduce an expanded notation for the inverse of the covariance matrix:

\[
C_{RN}^{-1} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}, \tag{21}
\]

Then, it can be shown that the ML estimates for the tracking parameters satisfy the \( 4 \times 4 \) matrix equation

\[
\bar{A} \begin{bmatrix} x_{0k} \\ \dot{x}_k \\ y_{0k} \\ \dot{y}_k \end{bmatrix} = \bar{B} \tag{22}
\]

where

\[
\bar{A} = \begin{bmatrix} \langle c_{11} \rangle_k & \langle c_{11} t_n \rangle_k & \langle c_{12} \rangle_k & \langle c_{12} t_n \rangle_k \\ \langle c_{21} \rangle_k & \langle c_{21} t_n \rangle_k & \langle c_{22} \rangle_k & \langle c_{22} t_n \rangle_k \\ \langle c_{11} t_n \rangle_k & \langle c_{11} t_n \rangle_k + \frac{(\cos^2 \phi_n)}{\sigma_D^2} & \langle c_{12} t_n \rangle_k & \langle c_{12} t_n \rangle_k - \frac{(\sin \phi_n \cos \phi_n)}{\sigma_D^2} \\ \langle c_{21} t_n \rangle_k & \langle c_{21} t_n \rangle_k - \frac{(\sin \phi_n \cos \phi_n)}{\sigma_D^2} & \langle c_{22} t_n \rangle_k & \langle c_{22} t_n \rangle_k + \frac{(\sin^2 \phi_n)}{\sigma_D^2} \end{bmatrix}
\]

and

\[
\bar{B} = \begin{bmatrix} \langle c_{11} x_n + c_{12} y_n \rangle_k \\ \langle c_{21} x_n + c_{22} y_n \rangle_k \\ \langle c_{11} x_n t_n + c_{12} y_n t_n \rangle_k + \frac{(D_n \cos \phi_n)}{\sigma_D^2} \\ \langle c_{21} x_n t_n + c_{22} y_n t_n \rangle_k - \frac{(D_n \sin \phi_n)}{\sigma_D^2} \end{bmatrix},
\]

where the triangular bracket notation is defined by Eq.(14). Here, \( \{x_n, y_n, D_n, t_n\} \) are the north, east, Doppler, and time, respectively, measured for sample \( n \). Eq.(22) is inverted, separately for each \( k \), to solve for the tracking parameters \( x_{0k}, y_{0k}, \dot{x}_k, \dot{y}_k \), during each iteration of the algorithm. It is interesting to note that for diagonal covariance, i.e., \( c_{12} = c_{21} = 0 \), \( c_{11} = 1/\sigma_x^2 \), \( c_{22} = 1/\sigma_y^2 \), Eq.(22) reduces to equations 7.2-24 in [7] for the case of a single, stationary, sensor platform.

Let us review the model parameters. The full set is \( \Theta_k = \{\Theta_{zk}, \Theta_{Rk}, \Theta_{DK}\} \), for \( k = 1, 2, 3, ..., K \). Here, the parameters in the classification feature space are \( \Theta_{zk} = \{M_{zk}, C_{zk}\} \). The parameters in the tracking space (for the target model components, not clutter), are \( \Theta_{Rk} = \{x_{0k}, y_{0k}, \dot{x}_k, \dot{y}_k\} \) which are related to the mean parameter \( M_{Rk} \) by Eq.(19). Finally, the parameters \( \Theta_{DK} \) in the Doppler space are, again, \( \{\dot{x}_k, \dot{y}_k\} \) which are related to the mean parameter \( m_{DK} \) by Eq.(20). The covariance parameters \( c_{ij} \) and \( \sigma_D \) (in north/east and Doppler, respectively) are a special case since they depend upon known quantities, and we’ll discuss them below in more detail.

In summary, the tracking algorithm works in a similar fashion to the generic case described in the previous section. We start with an initial guess for the parameters, then alternate back-and-forth between updating the association probabilities, then the parameters, until convergence. Proof of convergence is identical to the proof for the generic Gaussian mixture since the model for tracking is simply a special case of the generic Gaussian mixture model. In this paper we are emphasizing mainly the tracking problem rather than detection, however we note that that detection, or “object-track declaration”, can be performed using a log-likelihood ratio test, e.g., as described in Chapter 7.2.9 of [7].

The inverse covariance element parameters \( c_{ij} \) and the Doppler error parameter \( \sigma_D \) deserve a special note – they are not estimated since they are related to the sensor accuracy and are therefore known quantities. However, rather than simply plugging in the known values, it has been found that faster and more reliable convergence is obtained by initializing these parameters to large values, so that model components overlap to seamlessly cover the entire spatial...
field-of-interest. Then, as the iterations progress, these parameters are shrunk according to a predetermined schedule, reaching steady-state after, say, 50–100 iterations. The steady-state values represent the known accuracy of the sensor. Explicitly, the $c_{n}^{ij}$ are related to the sensor range $\sigma_{r}$ and cross-range $\sigma_{x,r}$ accuracies via

$$
\begin{bmatrix}
    c_{n}^{11} & c_{n}^{12} \\
    c_{n}^{21} & c_{n}^{22}
\end{bmatrix} = A_{n}\begin{bmatrix}
    1/\sigma_{r}^{2} & 0 \\
    0 & 1/\sigma_{x,r}^{2}
\end{bmatrix} A_{n}^{T},
$$

where $A_{n} = \begin{bmatrix}
    \cos \phi_{n} & \sin \phi_{n} \\
    -\sin \phi_{n} & \cos \phi_{n}
\end{bmatrix}$ is the rotation matrix.

IV. SAMPLE RESULTS.

We now present some results from synthetic data designed to test the tracking algorithm. The tracking software was written in Matlab. Noisy position (east/north), RCS, and Doppler measurements were generated for three targets over six revisits separated by a 10 second interval. In addition, 100 samples of clutter per revisit frame were generated, uniformly distributed in east/north, and normally distributed in RCS and Doppler. Three target components were used for the model, plus one uniform clutter component. Figures 1 and 2 illustrate the scenario. All plots are rendered in terms of the east/north axes, extending 1km in each direction, with all revisit frames superimposed. In Figure 1, the left-hand plot shows the distribution of data samples (both target and clutter) over the area-of-interest. These points appear as a collection of bright dots on a dark background, with brightness corresponding to the value of the Doppler. From this plot, it is theoretically possible to visually select six dots for each target and use them to estimate the tracks but, due to the large amount of high-amplitude clutter, the plot is too confusing to make this a practical task. The right-hand plot shows the true target tracks as dark lines over a white background. Figure 2 (a)-(d) show the evolution of the track model over the course of 25 DL iterations. Specifically, we are imaging the pdf [Eq.(1)] of the data projected onto the east/north axes. Initially (a), the model components are broad and unresolved due to large initial values for covariance parameters. Then, as the model evolves (b)-(c), the components shrink and lock onto the samples corresponding to the targets. Upon convergence (d) the model is able to properly associate the samples with the correct targets and estimate the tracks. This experiment was designed to be very challenging in the sense that the target RCS feature and Doppler are buried in the noise, with signal-to-clutter S/C of around −3dB (for example, the Doppler of target 2 is 5 m/sec, while the clutter has a standard deviation of 7 m/sec in Doppler). A “rule-of-thumb” S/C requirement for GMTI tracking is about 10–15dB, therefore we are showing several orders of magnitude improvement over the state of the art. Note that the radar accuracy for this experiment is $\sigma_{r} = 10m$, $\sigma_{x} = 30m$, and $\sigma_{D} = 0.5m$/sec.

It is illuminating to plot the evolution of the parameter estimates during DL iterations. Figure 3 shows cross-range velocity $\dot{y}_{k}$ vs. iteration for the three targets. It can be seen that the estimates vary considerably during model evolution until finally converging very nearly to the true values. A good question to ask is what happens when you don’t know in advance how many targets are present. For this simulation, five target model components (plus one uniform clutter component) were actually used (i.e., two extra components). As shown in Figure 4, the model adapted to discount the extra components by setting their weights to nearly zero upon convergence. Therefore, in this example it was not necessary to know beforehand the true number of targets – the model adapted to choose the best number (3) of target components.

Next, we tested the algorithm on a more difficult scenario in which 30 targets were present, with considerable overlap between the tracks. In this case, the tracker was able to converge and estimate all target tracks with acceptable accuracy. Figure 5 shows qualitatively how the model evolved with increasing iterations for this more complicated example. As in Figures 1 and 2, the axes in the plots are north/east, each spanning 1km.

Finally, we tested the ability of the tracker to compute tracks from combined data acquired using multiple view angles, for example if there were multiple, flying, sensors. In figure 6, we compare tracking results using a single sensor...
The approach was shown to perform well on synthetic data. Incorporate data from multiple, moving, sensor platforms. Existing trackers stemming from the data association problem.

\[ \phi \]

Of two sensors (right, plot b) having look angles \( \phi = \pm 45^\circ \). There are 3 targets present. The blue dotted lines indicate the true tracks, while the pink dashed lines show the estimated tracks. The black X's indicate GMTI detections derived from targets over 6 revisits, while the black dots indicate clutter. The colored surfaces indicate the probability density function projected onto the east/north axes, i.e., giving the target position ambiguity for the different revisits (note these surfaces have a bit different meaning than the surfaces shown in Figures 2 and 5). The oblong shapes of the ellipses indicate the differences in range vs. cross-range accuracies (here, 10m range accuracy vs. 100m cross-range accuracy). The results indicate much closer agreement between true tracks and estimated tracks for the multiple sensor case (right, plot b).

V. CONCLUSION.

In this paper we describe a novel approach to multi-target tracking that mitigates the combinatorial complexity of existing trackers stemming from the data association problem. The tracker is optimum in the sense that all available information is used, including classification features, when available. Also, the approach is flexible enough to seamlessly incorporate data from multiple, moving, sensor platforms. The approach was shown to perform well on synthetic data.

REFERENCES


