An Efficient Message Passing Algorithm for Multi-Target Tracking[*]

Zhexu (Michael) Chen[a], **Lei Chen**[a], **Müjdat Çetin**[b,a], and **Alan S. Willsky**[a]
[a]Laboratory for Information and Decision Systems, MIT, Cambridge, MA, USA
[b]Faculty of Engineering and Natural Sciences, Sabancı University, İstanbul, Turkey

Abstract – *We propose a new approach for multi-sensor multi-target tracking by constructing statistical models on graphs with continuous-valued nodes for target states and discrete-valued nodes for data association hypotheses. These graphical representations lead to message-passing algorithms for the fusion of data across time, sensor, and target that are radically different than algorithms such as those found in state-of-the-art multiple hypothesis tracking (MHT) algorithms. Important differences include: (a) our message-passing algorithms explicitly compute different probabilities and estimates than MHT algorithms; (b) our algorithms propagate information from future data about past hypotheses via messages backward in time (rather than doing this via extending track hypothesis trees forward in time); and (c) the combinatorial complexity of the problem is manifested in a different way, one in which particle-like, approximated, messages are propagated forward and backward in time (rather than hypotheses being enumerated and truncated over time). A side benefit of this structure is that it automatically provides smoothed target trajectories using future data. A major advantage is the potential for low-order polynomial (and linear in some cases) dependency on the length of the tracking interval N, in contrast with the exponential complexity in N for so-called N-scan algorithms. We provide experimental results that support this potential. As a result, we can afford to use longer tracking intervals, allowing us to incorporate out-of-sequence data seamlessly and to conduct track-stitching when future data provide evidence that disambiguates tracks well into the past.*

Keywords: Multi-target tracking, graphical models, message passing, data association, smoothing, multi-hypothesis tracking.

1 Introduction

Multi-target tracking (MTT) using data from multiple sensors is a very important, well-studied, and challenging problem that has a variety of applications, ranging from military target tracking to civilian surveillance. While a variety of important practical considerations add to this challenge, even if we limit attention to the most basic problem of maintaining track on a fixed set of targets using data from multiple sensors, we are met by a fundamental problem, namely the exponential explosion (over time) of potential associations of measurements from each sensor at each time with each target.

Practical solutions to this NP-complete problem of data association and target tracking consequently require some type of approximation. One of the most widely used approaches to such problems is commonly known as multiple hypothesis tracking (MHT) [1]. While tremendous advances have been made in organizing the computations and data structures associated with MHT, allowing it to be applied to practical applications of considerable size, the fundamental structure of MHT has several implications, some of which are well-known while others are perhaps not. Roughly speaking, MHT keeps track of sequences of data association hypotheses over time. In principle, to maintain consistency across targets we need to form consistent global hypotheses that preclude assigning the same measurement to two different tracks. While ingenious methods have been developed to deal with this global consistency constraint without explicit construction of global hypotheses, the fact remains that exponential growth in complexity is not eliminated. In particular, the extension of a track hypothesis over time requires the growing of a hypothesis tree, which is extended at each point in time as new measurements are received and incorporated. This combinatorial explosion requires approximation. While the number of variants for such approximations are numerous, they all generally involve two components, namely limiting the depth of the hypothesis tree - i.e., how far back into the past we keep track of possible assignments - and a method for collapsing hypotheses that differ only in assignments at the back end of that tree. A basic method for limiting tree depth is the so-called N-scan approximation. One widely used method for collapsing such hypothesis trees is simply to choose the branch extending from time $t - N$ to time t with highest likelihood or probability. This corresponds to pruning the hypothesis tree by keeping only a single root at time $t - N$.

There are a number of issues associated with existing MHT algorithms. First, although the N-scan approximation

[*]This work was partially supported by the U.S. Army Research Office under MURI Grant W911NF-06-1-0076, by the U.S. Air Force Office of Scientific Research under Grant FA9550-08-1-0180 and MURI Grant FA9550-06-1-0324, and by the Scientific and Technological Research Council of Turkey under Grant 105E090.

controls the explosion of hypotheses by limiting the depth of hypothesis trees, the complexity within the tracking window is still exponential in N. This puts a severe limit on how large one can choose N. An additional issue is the apparent logical inconsistency between the association and location estimation operations: while future data are used for computing probabilities for various hypotheses, these future data are not used for estimating (i.e. smoothing) the target states at this earlier time.

In this paper, we take a fundamentally different approach to solve the multi-sensor multi-target tracking and data association problem by exploiting the use of *graphical models* and *efficient message passing algorithms*. This framework offers the potential for approximations quite different than, but just as good as those in state-of-the-art MHT algorithms, but with drastically reduced complexity. One significant aspect of using graphical model representations as a starting point is that they lead directly to so-called message-passing algorithms to compute various probabilities, likelihoods, and estimates associated with variables at nodes in the graph. A second aspect is that there are different ways in which to construct graphical models for the same problem, each of which exposes different aspects of the overall probabilistic structure, making particular computations more natural in one representation than in another and also leading to very different ways in which to introduce approximations to control complexity. The graphical representation we introduce here leads to algorithms that do not enumerate track hypotheses as in MHT but rather directly compute probabilities of individual data associations at each point in time as well as both causally filtered and smoothed estimates of track states at each point in time. Thus, in contrast to MHT approaches, the one presented here naturally computes different quantities that are not easy to extract from MHT representations. Of course the flip side is that the computations explicitly exposed in MHT - e.g., track hypotheses over time - are not explicitly formed in our approach.

While this new perspective in modeling is interesting, simply by changing the way we model the problem will not change the complexity of solving it. As we know, the exact solution to MTT is exponential in the duration of the tracking window. So is the case for exact MTT using graphical models. Thus, to make target tracking over time tractable, it is necessary to use some approximation, however in the message-passing framework used here, we are interested in approximating *messages*. We develop our own methods using automatic, statistically principled approaches involving message approximation through multiresolution clustering, gating in message construction, and an N-scan approximation. In our examples we demonstrate that in some scenarios excellent performance can be obtained with complexity *that grows almost linearly with the length of the tracking interval*. As a result, we can consider far greater tracking intervals than methods that have to deal with exponential complexity. This not only allows for incorporation of data that arrive quite late but also allows greatly enhanced possibilities for track-stitching. We demonstrate all of these aspects in our experiments.

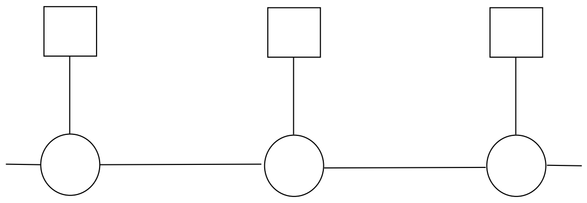


Figure 1: The first of the two graphical models we use for MTT. This graph collapses all targets and all sensors at a single point of time into one target node and one data association node, respectively.

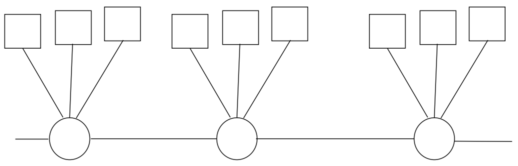


Figure 2: The second of the two graphical models we use for MTT. This graph distributes the global assignment variable at each point of time into individual data association nodes for each sensor.

2 Graphical Models for Tracking

2.1 Graphical Model Structure

A graphical model is simply a Markov random field defined on a graph in which nodes index variables defining our problem and an edge between nodes captures statistical relationships among the variables at the nodes connected by that edge. A set of nodes forms a clique if there are edges between all pairs of these nodes. If the joint distribution of all variables factors as a product of potential functions on cliques, then the variables are said to be Markov on the graph. We use the two graphical model structures in Figures 1 and 2. Each circle in these graphs represents the kinematic states of all targets at one time point, whereas each square connected to a circle represents the data associations at that particular time. The model in Figure 1 lumps all associations from all sensors at a single point in time together, whereas the model in Figure 2 uses one association node per sensor at each individual point in time. We note that circles represent continuous random variables, whereas squares denote discrete ones. Edges between successive points in time capture the statistical structure of the Markovian target dynamics. It is important to emphasize that these models capture the same type of statistical structure as that used in other tracking algorithms (e.g., an MHT algorithm), but they suggest very different algorithms based on message passing.

Although the static data association problem at a single point in time is already a challenging problem, it is not the focus of this paper. Rather, the focus here is in finding an efficient way to do tracking over a period of time. For more elaborate work on using graphical models to solve large static data association problems, see our previous work in [2].

In our first model (Figure 1), at each time point, all targets are lumped together to form one global target node and all sensors are lumped together to form one global assignment node. Here, every assignment node takes on discrete values, each of which represents a possible global data association assignment for all sensors at that time point. Each target node is the collection of kinematic states of all M targets at that time point: $x_t = [x_{t,1}^T, x_{t,2}^T, ..., x_{t,M}^T]^T$, where $x_{t,i}$ is the kinematic state of target i at time t.

If there are M targets and K sensors, then the complexity to enumerate global data associations at a single point of time is $(M!)^K$. To reduce that complexity, we propose the second model (Figure 2), in which the global assignment variable at each point in time is distributed, reducing the complexity of data association at each point of time to $K(M!)$. Each assignment node now corresponds to a sensor, and the value of such an assignment node indicates the data association between observations generated by that sensor and the targets it observes. From a statistical viewpoint, the second model asserts that the assignment of measurements at each sensor is conditionally independent of those at the other sensors, given the target states, a reasonable assumption in practice. For the sake of notational simplicity, we derive most of our formulae using the first model. In various parts of our discussion, we mention how the expressions would change for the second model. In the experiments, we use the second model, due to its reduced memory requirements.

Now let us introduce the form of the probability density associated with our graphical model. For a time period from $t = t_0$ to $t = T$, let x denote the kinematic states of all targets at all time points, y denote the collection of all observations from all sensors at all time points, and a denote all data association assignments for all observations at all time points. Then the joint probability density for the whole time window is given by:

$$p(x, y, a) = \prod_{t=t_0}^{T} p(y_t|a_t, x_t) \prod_{t=t_0}^{T} p(a_t)p(x_{t_0}) \prod_{t=t_0}^{T-1} p(x_{t+1}|x_t)$$

where x_t, y_t, and a_t are hidden target kinematic states, observations, and assignment variables at time t. The dynamic model and the observation model that make this equality possible will be described in subsequent subsections.

2.2 Data Association (Assignment) Nodes

For the ith observation ($i = 1, \ldots, O_{t,k}$), of sensor k at time t, let us define the assignment variable as:

$$a_{t,k}(i) = \begin{cases} 0 & \text{if observation } i \text{ is assigned as a false alarm} \\ m & \text{if observation } i \text{ is assigned to target } m \end{cases}$$

By stacking all assignment variables $a_{t,k}$ for all sensors $k = (1, ..., K)$, we obtain the global assignment variable a_t at time t.

We define the potential function for an assignment node in such a way that it takes into account the effects of false alarms and missed detections. Suppose that out of the $O_{t,k}$ observations made by sensor k at time t, $O_{t,k}^{FA}$ are assigned as false alarms, and $O_{t,k}^{DT}$ are assigned to targets for a particular assignment. Assuming for simplicity that all M targets are in the observation range of each sensor, the node potential $\psi_a(a_t) = p(a_t)$ for assignment node a_t is given by:

$$\psi_a(a_t) = \prod_{k=1}^{K} P_D^{O_{t,k}^{DT}} (1-P_D)^{M-O_{t,k}^{DT}} P_{FA}^{O_{t,k}^{FA}} (1-P_{FA})^{O_{t,k}-O_{t,k}^{FA}} \tag{1}$$

where P_D is the probability of detection, and P_{FA} is the probability of false alarm. If we used the graphical model in Figure 2, the potential function for the assignment node of the kth sensor at time t would simply consist of the kth factor in (1).

2.3 Target Dynamic Subgraphs

We represent target dynamics using linear models: $x_t = Ax_{t-1} + u_{t-1}$, where A is the transition matrix; u_{t-1} is a stationary zero-mean white Gaussian noise process; and x_t is the kinematic state vector at time t, in which the kinematic states $x_{t,m}$ ($m = 1, ..., M$) of all M targets are stacked. The potential function for the target nodes captures only target initial conditions and is given by:

$$\psi_x(x_t) = \begin{cases} p(x_{t_0}) = \mathcal{N}(x_{t_0}; \mu_{t_0}, \Sigma_{t_0}) & \text{if } t = t_0 \\ 1 & \text{if } t > t_0 \end{cases} \tag{2}$$

where μ_{t_0} and Σ_{t_0} are the parameters of the prior distribution for each target at the start of the time interval of interest.

The potential function for the edges connecting the target nodes is given by:

$$\psi_{t,t+1}(x_t, x_{t+1}) = p(x_{t+1}|x_t) \tag{3}$$

2.4 Edges Joining Associations and Targets

We use the observation likelihoods as edge potentials, and a linear Gaussian model for the sensor measurements. Let $y_{t,k}(i)$ denote the ith observation from sensor k at time t. Unless this observation is assigned to a false alarm, its value depends on the kinematic state of target $a_{t,k}(i)$: $y_{t,k}(i) = C_{t,k} x_{t,a_{t,k}(i)} + v_{t,k}$, where $C_{t,k}$ is the observation matrix, and $v_{t,k}$ is a stationary, zero-mean, white Gaussian noise process. By stacking all the observations $y_{t,k}(i)$ ($i = 1, \ldots, O_{t,k}$) produced by sensor k at time t, we obtain the observation vector $y_{t,k}$ for the kth sensor. Then by stacking the observations $y_{t,k}$ from all sensors at time t, we obtain the overall observation vector y_t at time t. Based on this observation model, we define the potential function for the edges connecting assignment nodes and target nodes as:

$$\psi_{a,x}(a_t, x_t) = p(y_t|a_t, x_t) = \prod_{k=1}^{K} p(y_{t,k}|a_{t,k}, x_t) \qquad (4)$$

If we used the graphical model in Figure 2, the potential function between the kth association node and the target node at time t would simply be the kth factor in (4).

3 Approximate, Efficient Algorithm for Multi-Target Tracking

In this section we describe the message-passing computations required for inference in our graphical model. We use belief propagation (BP) to estimate the posterior probability distribution of target kinematic states x, as well as associations a, given observations y. BP message passing equations in principle provide exact computation of posterior marginals on tree-structured graphs. However exact computation of the messages in practice necessitates some special structure. Two such special cases that have been widely exploited are the cases of graphs involving only discrete or only Gaussian random variables and messages. The graphical models we have constructed in Section 2 result in discrete and Gaussian-mixture messages. Hence, although not as simple as the two cases mentioned, our models exhibit some special structure as well. Exploiting this structure, in Section 3.1 we discuss performing exact belief propagation for multi-target tracking.

Part of the novelty of our approach is the structure of our message-passing implementation. Among other things, this accomplishes two things. The first is that it exploits the Markov structure of our graphs to pass messages backward in time in order not only to smooth target state estimates using future data (which may be of interest in itself for some applications) but also to use these smoothed estimates in the process of updating and resolving data association hypotheses at previous points in time (bringing the "data to the hypothesis" rather than the "hypothesis to the data" as in hypothesis-tree-based approaches such as MHT). The second consequence of this implementation is that it focuses the challenge of dealing with exponential complexity in a different manner than in MHT. In particular, this challenge manifests itself in terms of managing the complexity of *messages* passed from node to node, rather than managing temporally-growing association hypothesis sequences. Roughly speaking, in our algorithm, each mode of the Gaussian mixture messages acts like a *particle*[1] to be transmitted among the nodes. Running exact BP on our graphs leads to exponentially growing number of particles in BP messages, hence exponentially growing computational complexity in time. In Sections 3.2 through 3.4 we describe three methods to manage and reduce that complexity via various approximations. The first two of these are fairly standard in concept although different in detail because of the nature of our implementation. The third method for controlling complexity, described in Section 3.4, has no counterpart in standard MHT algorithms and is a key benefit of our formalism, as it corresponds to approximating messages to meet a specified fidelity criterion.

[1]For the sake of clarity, we should point out that the meaning of the term "particle" here is different from its standard usage in the context of particle filtering [3].

3.1 BP on the Tracking Graph

We can identify three types of messages in the graphical models in Figures 1-2: from a continuous target node to another continuous target node, from a discrete assignment node to a continuous target node, and from a continuous target node to a discrete assignment node.

Messages from a discrete assignment node to a continuous target node can be computed as follows:

$$M_{a\to x}(x_t) = \kappa \sum_{a_t} \psi_a(a_t)\psi_{a,x}(a_t, x_t) \qquad (5)$$

Given the definitions in (1) and (4), this message is basically a sum of Gaussian distributions.

We compute the forward messages $M_{t\to t+1}(x_{t+1})$ sent from a continuous target node at time t to the next target node at time $t+1$ as follows:

$$\kappa \int \psi_{t,t+1}(x_t, x_{t+1})\psi_x(x_t)M_{t-1\to t}(x_t)M_{a_t\to x_t}(x_t)dx_t. \qquad (6)$$

With both $M_{t-1\to t}(x_t)$ and $M_{a_t\to x_t}(x_t)$ being Gaussian mixtures, $M_{t\to t+1}(x_{t+1})$ is also a Gaussian mixture. Note that this message computation involves multiplication and integration of Gaussian mixtures, for which we derive and use expressions based on the development in [4, 5]. Note that, for backward messages, the equation is similar (with minor changes of subscripts). If we used the distributed model in Figure 2, then the only change would be that $M_{a_t\to x_t}(x_t)$ would be replaced by a product of messages from individual sensor nodes. As one can imagine, the number of modes in these target-to-target messages increases multiplicatively from time to time, which necessitates the kind of approximations we describe in subsequent subsections.

The messages from a continuous target node to a discrete assignment node $M_{x\to a}(a_t)$ are computed as follows:

$$\kappa \int \psi_{a,x}(a_t, x_t)\psi_x(x_t)M_{t-1\to t}(x_t)M_{t+1\to t}(x_t)dx_t \qquad (7)$$

As the assignment variable a_t is a discrete variable, this message is a finite-dimensional vector. Note that if we used the model in Figure 2, this message would denote the message to one particular sensor node. In that case, we would have an additional factor in the integrand in (7) consisting of the product of messages from the other sensor nodes to the target node, and we would replace $\psi_{a,x}(a_t, x_t)$ with the appropriate edge potential between the target node and that particular sensor node.

3.2 Gating in Message Construction

Gating is a standard technique used in MHT as well as other tracking algorithms to limit the number of data association hypotheses being generated. In the context of our message passing algorithm, gating is done in the computation of the message in (6), and in particular in computing the product of the messages in the integrand, i.e., when messages from assignment nodes, or *assignment messages*, are multiplied with messages from target nodes, or *target messages*. Without gating, every particle (i.e., a mode in the Gaussian mixture) in the target message would be multiplied with every particle in the assignment message. With gating, rather than multiplying a particle in the target message with every single particle from the assignment message, each particle in the target message is only multiplied with the ones in the assignment message with data associations that are consistent with its gating constraints. The gating regions can be determined by the means and the covariances in target messages, because these messages can be interpreted as estimates of target kinematic states.

3.3 N-Scan Approximation

The version of N-scan used in our experiments involves stopping sending messages back to points in time after they exit the N-scan interval. The only issue, then is the last messages sent going forward from an exiting point in time. In standard N-scan algorithms this might correspond to choosing a single most likely data association hypothesis as this point exits the window. In our algorithm, after receiving the data at time t, messages are passed backward in the network, until we compute $p(a_{t-N})$ for some fixed N. Now, when sending messages from this assignment node back to the target node using (5), rather than considering all possible associations, we set some threshold β, order the possible associations based on their probabilities, and keep the minimum number of associations whose sum of probabilities just exceeds β. Note that the number of hypotheses kept is determined by the algorithm in an adaptive manner. In this way, we eliminate all the less likely associations whose sum of probabilities is around $1 - \beta$.

3.4 Message Approximation by Clustering

A critical component in managing complexity that is available to us thanks to our message-passing algorithm is the approximation of messages prior to passing them to neighboring nodes - i.e., approximating one Gaussian mixture distribution with another one with fewer modes. For this approximation, we use a clustering procedure that adaptively reduces the number of particles to be used in each message passing stage. We emphasize that this approximation is done solely for the temporary purpose of transmitting a message, and all of the possible data association hypotheses are still preserved in the assignment nodes in our graph.

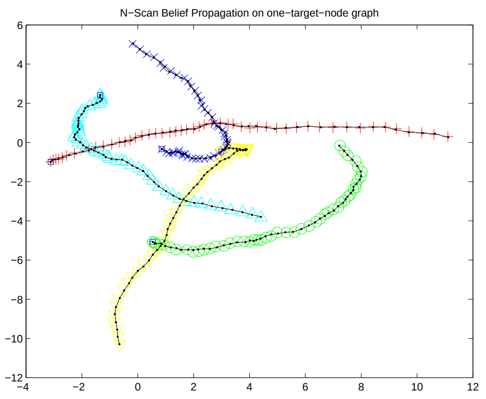
We use a multiresolution clustering approach based on K-dimensional trees (KD-trees) [6]. A KD-tree is a space-partitioning data structure used to organize a large set of data points. In KD-trees data are stored in a multi-scale fashion, which forms the basis of their use in a clustering algorithm. We are interested in approximating an input Gaussian mixture distribution, with another one with a smaller number of modes by clustering together similar modes. Given a Gaussian mixture, we construct a KD-tree, in which the root node corresponds to the input Gaussian mixture, and each leaf node corresponds to a single mode in the Gaussian mixture. For the sake of brevity, we do not describe our procedure for constructing the tree. We represent each node by a K-dimensional data vector consisting of the elements of the mean vector together with the elements of the covariance matrix of the corresponding Gaussian.

Given the constructed tree, we calculate and store three statistics for each node: a weight, mean vector, and covariance matrix. With these statistics, each node can be viewed as a Gaussian approximation of its children. Given the constructed KD-tree with computed statistics, we then use it for clustering. We take a walk down the KD-tree starting from the root node. At each node, we calculate the symmetrized Kullback-Leibler (KL) divergence between the two children, and we stop at that node if the KL-divergence between its children is smaller than a threshold specified by the user. We keep all the nodes at which this procedure has stopped, and use that as the approximate representation of the input mixture. This effectively makes a multi-resolution cut through the tree, in which the number of nodes kept is the number of modes (particles) in the approximate representation. We use this clustering procedure to limit the number of particles used in our messages. Since the number of particles is what leads to exponential complexity of the exact algorithm, clustering plays a key role in beating that complexity. As will be demonstrated in our experiments, this procedure helps us achieve almost linear complexity in some scenarios.

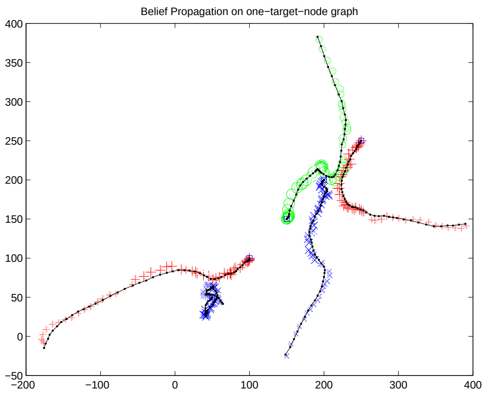
4 Experimental Results

4.1 Setup

In our simulations, multiple targets move in a 2-D surveillance area. The number of targets is known *a priori*. The movement of each target is modeled by a linear, time-invariant state-space model, in which the kinematic state vector for each target consists of 2-D position, velocity, and acceleration. Target state dynamics involve some temporal correlation in acceleration. The process noise mainly drives the acceleration. We consider three types of sensors monitoring the surveillance area. Type I and Type II sensors are bearing-only sensors located far away from the surveillance region, providing one-dimensional measurements. Type I sensor measures horizontal position and velocity, whereas Type II does the same for the vertical dimension. Type III sensor provides near-field measurements of 2D positions.



(a)



(b)

Figure 3: Sample tracking results with $N = 5$. (a) Type I, II, & III sensors, high SNR. (b) Type I & II sensors, low SNR.

We include false alarms and missed detections, the probabilities of which are set to be 0.05. Measurements are corrupted by additive Gaussian noise. Initial kinematic states are generated randomly, and subsequent states are generated according to the dynamic model mentioned above.

4.2 Tracking Performance and Complexity

In Figure 3(a), we show a sample tracking result (we use N=5 in N-scan, and a KL threshold of 0.1). This is only one example out of the 100 runs we have generated. In all of them, there are 5 targets, 3 sensors (one of each type), duration is 50 time frames, and SNR is high. Black curves indicate the true target trajectories, and markers of each color show the estimated target position through the mean of each particle. Uncertainty in these estimates is also shown through one-standard-deviation ellipses, which are too small to visually observe in this plot. Weights of the particles are encoded through the density of the colors. We observe that our approach produces very good tracking accuracy in many runs of this scenario. Figure 3(b) presents

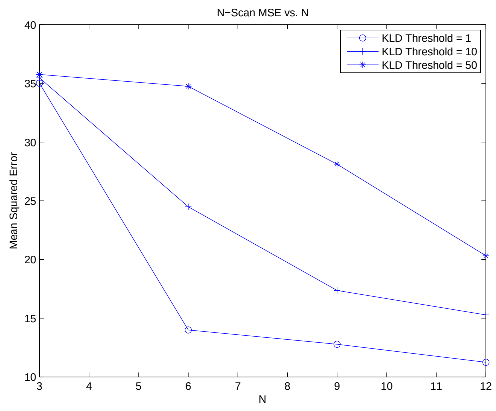


Figure 4: Mean-squared tracking error as a function of N and the KL threshold in a low-SNR scenario with Type I & II sensors.

a more challenging scenario: we use only Type I & II sensors and we add measurement noise with a variance of 100, hence this is a low-SNR scenario. In this case, we naturally observe some degradation as compared to the result in Figure 3(a), however we still achieve what appears to be satisfactory tracking accuracy. Figure 4 shows the overall mean-squared tracking error for this challenging scenario as the length of the N-scan window, and the KL threshold are varied. As expected, we achieve better performance as N is increased, and the KL threshold is decreased. Of course, this benefit should come with the price of more computations. Based on this observation, we next explore the computational complexity as a function of N. In Figure 5 we show the relationship between running time and N, for a five-target scenario involving all three type of sensors.[2] We conclude that by using adaptive KD-tree clustering as the hypothesis reduction method, while maintaining acceptable tracking accuracy, the message-passing algorithm achieves almost linear complexity with respect to the duration of the tracking window in this particular scenario.

4.3 Handling Delayed Information

We now present two examples demonstrating that our approach can incorporate delayed information in a seamless fashion thanks to its ability to use long tracking windows together with its forward-backward message passing structure.

In Figure 6, we compare our message-passing algorithm with $N = 15$ and $N = 3$, in a scenario in which observations from $t = 8$ to $t = 15$ arrive late at $t = 19$. If the tracking window is small ($N = 3$ as in (a)) then when late data arrive, the tracker is not able to incorporate those late data as the tracking window has already moved passed the range with late data. As a result, the tracker confuses

[2]Similar results are obtained for the case of Type I & II sensors.

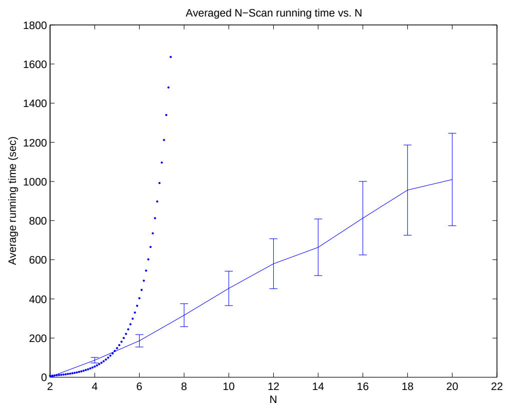
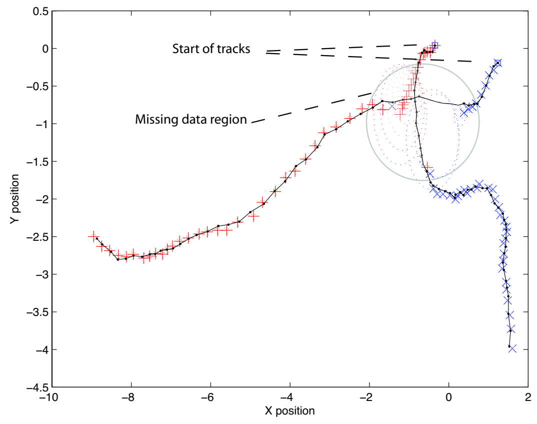
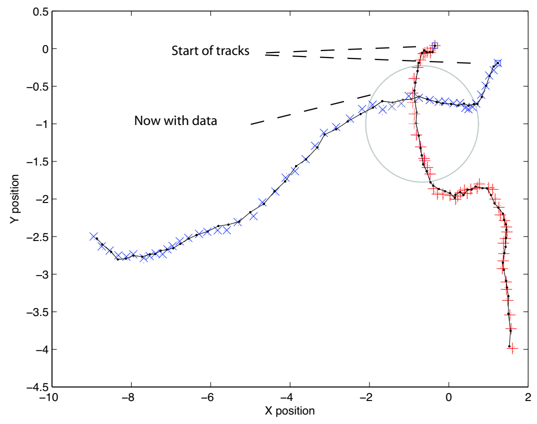


Figure 5: Running time as a function of N for a 5-target, high-SNR scenario with all three types of sensors, averaged over 100 runs. Error bars indicate the one-standard-deviation region. To contrast the complexity of our approach with that of a hypothetical MHT tracker, we also show an exponential curve.



(a)



(b)

Figure 6: Late data arrival example, with all three types of sensors. Observations from $t = 8$ to $t = 15$ arrive late at $t = 19$. (a) $N = 3$, resulting in inaccurate and uncertain tracking. (b) $N = 15$, recovering the target tracks for the interval of late data arrival.

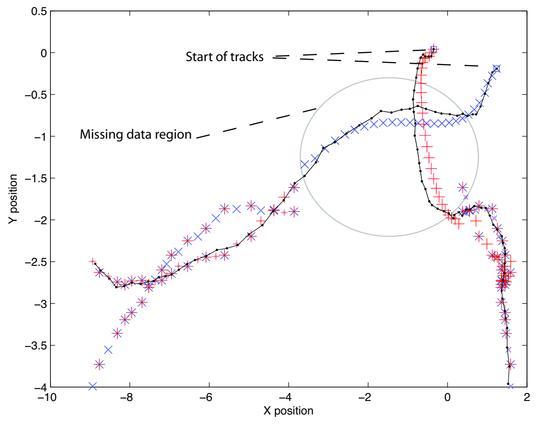
the two targets, and exhibits large estimation uncertainty in the late data interval. If the tracking window is long enough ($N = 15$ as in (b)), then to incorporate the late data when they arrive, the tracker just needs to conduct a regular backward-forward message-passing within its tracking window, resulting in much better tracking performance.

In Figure 7, we show an example of track-stitching, using our message-passing N-Scan algorithm with $N = 30$. In this scenario with 50 time frames, observations are missing for time points from $t = 5$ to $t = 25$. When we use a short tracking window of $N = 3$, the tracker cannot associate the tracks before and after the missing data region, resulting in the two ghost tracks in Figure 7(a). On the other hand, when we use a longer tracking window with $N = 30$, spanning across the period of missing data, then the tracker can associate the tracks before and after missing data, and ”stitch” the tracks together as shown in Figure 7(b).

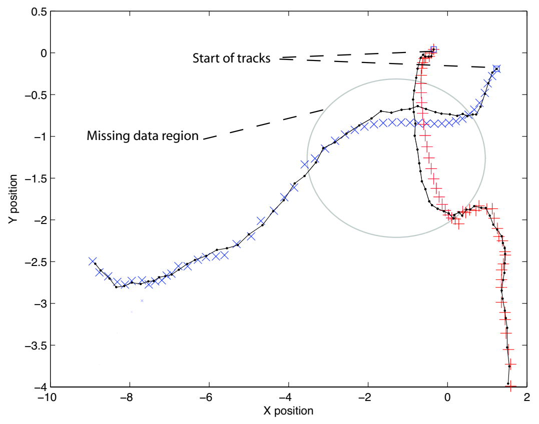
5 Discussion

We have presented a framework to solve the multi-target tracking (MTT) problem based on graphical model representations of the probabilistic structure of the MTT problem and message passing algorithms arising from such representations. The graphical model structure and associated inference algorithms offer enormous flexibility to overcome several limitations faced by existing MTT algorithms. In particular this formalism localizes the combinatorially explosive nature of MTT problems in a very different place, namely in the *messages* passed in the algorithm, both forward *and* backward in time. This opens up the possibility of very different approximation algorithms based not on pruning or eliminating data association hypotheses but rather on approximation of likelihood messages. We have seen through experiments that our approach to adaptively managing these approximations can lead to complexity that grows almost only linearly with the length of the tracking time interval in some scenerios, allowing much longer intervals to be considered. This facilitates one of several potential advantages of our approach, namely the stitching of tracks over considerable time intervals when only occasional target discriminating information becomes available. Moreover, the nature of our graphical models makes the incorporation of out-of-sequence data seamless, requiring literally no changes to algorithmic structure. In addition, this message-passing structure automatically produces smoothed target estimates, something that can be of considerable value in many applications other than real-time tracking.

This is only a first introduction of this framework and



(a)



(b)

Figure 7: Track stitching example, using Type I & II sensors. Observations are missing from $t = 5$ to $t = 25$. (a) $N = 3$, resulting in ghost tracks. (b) $N = 30$, achieving track stitching.

considerably more testing and considerations of complexities not included here must be undertaken. A more detailed computational complexity analysis covering a wider range of scenarios is already underway. Although we may not get linear complexity in the tracking interval in more complicated scenarios than the ones considered here, we still expect to get a low-order polynomial complexity, beating the exponential complexity of MHT-based trackers. In order to focus on the key novelties of this new formalism, we have stripped away some aspects that will need to be included in the future. For example, we have assumed linear-Gaussian target and measurement models (so that all of our probabilistic quantities are Gaussian sums). As our method intrinsically involves particle-like representations for messages, the incorporation of nonlinear dynamics and measurements is readily accommodated. In addition, as mentioned previously, we focus here on what is known as the track maintenance problem, and extensions to include track initiation and termination need to be developed in the future. We have presented one particular way to perform approximate inference in mixture models. Another approach to this problem would be to use nonparametric belief propagation (NBP) [7], which, in order to manage the size of messages being passed on the graph, employs a sampling technique to approximate them. When one uses a sampling-based approach for inference, managing the number of samples for complexity control is an interesting issue. If this can be done effectively, it would perform a similar function to our clustering-based message approximation approach. In this paper we have focused on the dynamic aspect of the tracking problem, and have assumed that the static data association problem (i.e., computing the association probabilities at each time instant) is tractable. An extension of the work presented in this paper would be to combine our dynamic tracking framework, with advanced (distributed) static data association techniques. Developing these and the other extensions not mentioned here due to space limitations offer considerable promise for new, high-performance MTT algorithms with many attractive characteristics.

References

[1] D. B. Reid, “An algorithm for tracking multiple targets,” *IEEE Trans. on Automatic Control*, vol. AC-24, pp. 843–854, 1979.

[2] L. Chen, M. J. Wainwright, M. Çetin, and A. S. Willsky, “Data association based on optimization in graphical models with application to sensor networks,” *Mathematical and Computer Modelling, Special Issue on Optimization and Control for Military Applications*, vol. 43, no. 9-10, pp. 1114–1135, May 2006.

[3] S. Arulampalam, S. Maskell, N. J. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear non-Gaussian Bayesian tracking,” *IEEE Trans. Signal Processing*, vol. 50, pp. 174–188, February 2002.

[4] R. Cowell, “Advanced inference in Bayesian networks,” in *Learning in Graphical Models*, ser. Adaptive Computation and Machine Learning, M. I. Jordan, Ed. MIT Press, Nov. 1998, ch. 2, pp. 27–49.

[5] S. L. Lauritzen and N. Wermuth, “Graphical models for associations between variables, some of which are qualitative and some quantitative,” *The Annals of Statistics*, vol. 17, no. 1, pp. 31–57, Mar. 1989.

[6] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, Sept. 1975.

[7] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky, “Nonparametric belief propagation,” in *Computer Vision and Pattern Recognition (CVPR)*, 2003.