Robust Neural-Network-Based Data Association and Multiple Model-Based Tracking of Multiple Point Targets

Mukesh A. Zaveri, S. N. Merchant, and Uday B. Desai, *Senior Member, IEEE*

***Abstract*—Data association and model selection are important factors for tracking multiple targets in a dense clutter environment without using *a priori* information about the target dynamic. We propose a neural-network-based tracking algorithm, incorporating a interacting multiple model and show that it is possible to track both maneuvering and nonmaneuvering targets simultaneously in the presence of dense clutter. Moreover, it can be used for real-time application. The proposed method overcomes the problem of data association by using the method of expectation maximization and Hopfield network to evaluate assignment weights. All validated observations are used to update the target state. In the proposed approach, a probability density function (pdf) of an observed data, given target state and observation association, is treated as a mixture pdf. This allows to combine the likelihood of an observation due to each model, and the association process is defined to incorporate an interacting multiple model, and consequently, it is possible to track any arbitrary trajectory.**

***Index Terms*—Data association, expectation maximization (EM), interacting multiple model, neural network (NN).**

I. Introduction

TRACKING of multiple point targets in the presence of a dense clutter has significance in surveillance and navigation. Various approaches have been proposed for multiple target tracking in [1] and [2]. The performance of a tracking algorithm depends on: 1) the data association method used for observation to track assignment and 2) the model selected to track the movement of a target. The most common method for data association is the nearest neighbor (NN) method that assigns one observation to a track based on minimum statistical distance. But there is always uncertainty about the origin of observation, and hence, it may result in a false track. The performance of the NN method degrades in a dense clutter environment. To avoid this uncertainty about the origin of an observation, a joint probabilistic data association filter (JPDAF) and multiple hypotheses tracking (MHT) have been developed [1]. In both cases, the complexity of the algorithm increases with an increase in the number of observations and the number of targets.

Manuscript received December 8, 2004; revised October 14, 2005. This paper was recommended by Editor A.-O. Nii.

M. A. Zaveri is with the Computer Engineering Department, Sardar Vallabhbhai National Institute of Technology, Surat 395007, Gujarat, India (e-mail: mazaveri@ee.iitb.ac.in).

S. N. Merchant and U. B. Desai are with the SPANN Laboratory, Electrical Engineering Department, Indian Institute of Technology (IIT)-Bombay, Mumbai 400076, India (e-mail: merchant@ee.iitb.ac.in, ubdesai@ee.iitb.ac.in).

Digital Object Identifier 10.1109/TSMCC.2007.893281

A maximum-likelihood approach and probabilistic multiple hypotheses tracking (PMHT) algorithm and different modified versions of PMHT have been proposed [3]–[6] to reduce the computational complexity. In all these approaches, assignment probabilities are calculated and used for the target state update. Also, these approaches do not use different dynamic models for the target, and hence, it is not possible to track an arbitrary trajectory. In PMHT, the centroid of observations is used to update the target's state. Hence, its performance depends on the noise level (the amount of clutter) and validation region size. The centroid of observations may lead to a false track, as the target movement is arbitrary. A recursive multiple target tracking algorithm using expectation maximization (EM) has been proposed by Molnar and Modestino [7]. This algorithm has been developed for use with one particular model and, hence, is not able to track any arbitrary trajectory. All the observations are used to update the state of a target. In this approach, assignment probabilities and target step update are evaluated in an iterative mode. Using a single type of filter, it is not possible to track an arbitrary trajectory.

To overcome the disadvantages of these methods, we have proposed an interacting multiple model–expectation maximization (IMM-EM) algorithm [8], [9] to track multiple point targets in the presence of dense clutter. This proposed method is computationally efficient as it does not consider all the data association events like in JPDAF and all validated observations are used to update the target state instead of the centroid of observations as in the case of PMHT, and consequently, it avoids the uncertainty in the state vector update. It does not assign any observation to any target, nor does it use this centroid to update the state of the target and it is free from the disadvantages of NN and PMHT [10], [11]. Incorporation of IMM [12] allows one to track any arbitrary trajectory.

In this paper, we extend our previously proposed method [8], [9] by utilizing neural network for data association. The novelty of the paper is that we address the difficult problem of multiple target tracking and data association in the presence of dense clutter. We have formulated the problem and provided a solution in a multimodel framework using: 1) IMM algorithm for tracking; 2) EM algorithm to evaluate assignment weights for observation to track assignment, which, in turn, avoids the uncertainty about the origin of an observation and, hence, prevents forming false track in the presence of the dense clutter; and 3) neural network to reduce computational complexity of the EM algorithm, which makes the algorithm fast and robust.

Neural-network-based data association has been developed to evaluate assignment probabilities in [13]. In our method, using a similar approach, assignment weights are calculated, but in a multimodel framework, and then, all these assignment weights are used for the state update. Neural-network-based evaluation of assignment weights makes data association robust. The main aim of the investigation is to develop an algorithm for *real-time tracking* with robust data association for dim multiple targets in the presence of dense clutter.

The major benefits and differences between the proposed algorithm and the one described in [8] are as follows.

1) In the proposed algorithm, a Hopfield neural network is used to evaluate assignment weights.
2) In [8], the assignment weights and state update for each target are done in an iterative manner that requires more computations and, hence, makes the overall algorithm execution slow. Whereas, in the proposed algorithm, only assignment weights are evaluated in an iterative manner by Hopfield network and state update for each target is done in a single step. This makes the algorithm faster as compared to that of [8]. Moreover, this makes the algorithm amenable to real-time implementation (assuming that we have a firm-ware implementation of the Hopfield network).
3) Neural networks (NNs) are known to extract the stored patterns from the data efficiently. Here, we use an NN to extract the data association pattern (i.e., observation to track assignment using assignment weights) from the available observations at the given time. This provides a robust approach for data association as compared to [8]. In fact, the algorithm given by us in [8] fails in the presence of dense clutter, but the present algorithm exploits NN works successfully.

II. Neural-Network-Based IMM-EM Algorithm

A. Problem Formulation

In this section, the problem is described in a multimodel framework to track both maneuvering and nonmaneuvering targets. The data association problem is solved as a constrained optimization problem using an NN proposed by Hopfield. The constraints for the data association problem are analogous to those of traveling salesman problem (TSP). Let N_T be the number of targets at time t, and it may vary with time. $\mathbf{\Phi}_t$ represents concatenated combined state vectors for all targets $s = 1, \ldots, N_T$, i.e.,

$$\mathbf{\Phi}_t = (\Phi_t(1), \Phi_t(2), \ldots, \Phi_t(N_T))^T$$

where $\Phi_t(s)$ is a combined state vector at time instant t for target s. The state at time instant t by model m for target s is represented by $\phi_t^m(s)$. Let the observation process be $\mathcal{Y}$ and its realization at time instant t be denoted by a vector

$$\mathbf{Y}_t = (y_t(1), y_t(2), \ldots, y_t(N_o))^T$$

where N_o denotes the number of observations obtained at time t, which may vary with time. To assign observations to targets, an association process, denoted by $\mathcal{Z}$, is formed. It is used to represent the true but unknown origin of observations. $\mathbf{Z}_t$ is a realization of an association process $\mathcal{Z}$ at time instant t, and it is referred to as the association matrix.

For each model, a validation matrix (association matrix) is defined. For IMM, we represent $\mathbf{Z}_t$ as combined (logically OR operation) realization of $\mathcal{Z}$ and is defined as

$$\mathbf{Z}_t = \mathbf{z}_{t,1} + \mathbf{z}_{t,2} + \cdots + \mathbf{z}_{t,M} \tag{1}$$

where $\mathbf{Z}_{t,m}$ is the association matrix at time instant t for model m. Here, M is the total number of models used in the IMM algorithm. Each $\mathbf{Z}_{t,m}$ is $(N_T + 1) \times (N_o + 1)$ matrix; the (s, i)th element of the association matrix $z_{t,m}(s, i)$ is given by

$$z_{t,m}(s,i) = \begin{cases} 1, & \text{if observation } y_t(i) \text{ originated from and} \\ & \quad \text{falls in the validation gate of target}s \\ 0, & \text{otherwise} \end{cases}$$

for $s = 0, \ldots, N_T$, $i = 0, \ldots, N_o$. Let $N_T = 5$ (targets) and $N_o = 4$ (observations) be present at time t, and $M = 2$ (models) for IMM, then the combined association matrix is defined as

$$\mathbf{Z}_t = \begin{matrix} \\ N_T \\ \downarrow \end{matrix} \begin{matrix} \begin{matrix} N_0 & \rightarrow \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \end{matrix}$$

$$= \mathbf{z}_{t,1} + \mathbf{z}_{t,2} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

In the matrix $\mathbf{Z}_t$, it is assumed that $Z_t(0, 0) = 0$ and has no significance. $Z_t(0, i) = 1$ for some i represents a false alarm and $Z_t(s, i) = 0$ for all $i (1 \le i \le N_o)$ implies that the target s is occluded. Similarly, for some $s (1 \le s \le N_T)$, if $Z_t(s, i) = 1$ for more than one value of $i (1 \le i \le N_o)$, then it represents a multiple observation event. An unresolved target event is identified, if for some i, $Z_t(s, i) = 1$ for more than one value of s.

The recursive estimate of combined state $\mathbf{\Phi}_t$ can be evaluated using Bayes' rule

$$p(\mathbf{\Phi}_t|\mathbf{Y}^t) = \frac{p(\mathbf{Y}_t|\mathbf{\Phi}_t)}{p(\mathbf{Y}_t|\mathbf{Y}^{t-1})} p(\mathbf{\Phi}_t|\mathbf{Y}^{t-1}) \tag{2}$$

where $\mathbf{Y}^t = (\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_t)$. The $p(\mathbf{Y}_t|\mathbf{\Phi}_t)$ represents an observation probability density function (pdf) at time instant t and $p(\mathbf{\Phi}_t|\mathbf{Y}^{t-1})$ is the predicted target state pdf $p(\mathbf{\Phi}_t|\hat{\mathbf{\Phi}}_{t-1})$. $\hat{\mathbf{\Phi}}_{t-1}$ is the previous state estimate. $p(\mathbf{Y}_t|\mathbf{Y}^{t-1})$ is the normalizing factor. The solution of (2) is direct if association is known. With unknown association, $p(\mathbf{Y}_t|\mathbf{\Phi}_t)$ can be solved by integrating it with respect to all possible configurations $\mathbf{Z}_t$ of association process $\mathcal{Z}$

$$\begin{aligned} p(\mathbf{Y}_t|\mathbf{\Phi}_t) &= \int_{\mathbf{Z}_t} p(\mathbf{Y}_t, \mathbf{Z}_t|\mathbf{\Phi}_t)\, d\mathbf{Z}_t \\ &= \int_{\mathbf{Z}_t} p(\mathbf{Y}_t|\mathbf{Z}_t, \mathbf{\Phi}_t) p(\mathbf{Z}_t|\mathbf{\Phi}_t)\, d\mathbf{Z}_t. \end{aligned} \tag{3}$$

It is difficult to implement (3) due to the term $p(\mathbf{Z}_t|\mathbf{\Phi}_t)$. EM method allows the estimation of association probability as a by-product. The EM approach is described in the following section.

B. EM Approach

Here, EM algorithm [14], [15] is used to evaluate MAP target state estimate using the state estimate at $t-1$. In our case, the EM approach is used to update the state sequentially, and not in batch mode as done earlier [4], [5].

The complete data are denoted as $\mathbf{X}^t = (\mathbf{X}_1, \ldots, \mathbf{X}_t)$, where $\mathbf{X}_t = (\mathbf{Y}_t, \mathbf{Z}_t)$. $\mathbf{Y}_t$ is the observation process and $\mathbf{Z}_t$ is the association process at time instant t. The estimate of $\mathbf{\Phi}_t$ is given by Bayes' rule

$$p(\mathbf{\Phi}_t|\mathbf{X}^t) = \frac{p(\mathbf{X}_t|\mathbf{\Phi}_t)}{p(\mathbf{X}_t|\mathbf{X}^{t-1})} p(\mathbf{\Phi}_t|\mathbf{X}^{t-1}) \tag{4}$$

where $p(X_t|X^{t-1})$is the normalizing term. We assume that the current state estimate of $\mathbf{\Phi}_t$ is only dependent on the current associated observation, i.e., $\mathbf{X}_t$, and the previous state estimate $\hat{\mathbf{\Phi}}_{t-1}$ (Markov). With known previous state estimate $\hat{\mathbf{\Phi}}_{t-1}$, the conditional pdf $p(\mathbf{\Phi}_t|\mathbf{X}^t)$ can be approximated by $p(\mathbf{\Phi}_t|\mathbf{X}_t, \hat{\mathbf{\Phi}}_{t-1})$ and, consequently, $p(\mathbf{\Phi}_t|\mathbf{X}^{t-1})$ in (4) can be replaced by $p(\mathbf{\Phi}_t|\hat{\mathbf{\Phi}}_{t-1})$ multiplied by the normalizing factor. The MAP estimate of $\mathbf{\Phi}_t$ is given by

$$\begin{aligned}\hat{\mathbf{\Phi}}_{t,\text{map}} &= \arg\max_{\mathbf{\Phi}_t}[\log\, p(\mathbf{\Phi}_t|\mathbf{X}^t)] \\ &= \arg\max_{\mathbf{\Phi}_t}[\log\, p(\mathbf{X}_t|\mathbf{\Phi}_t) + \log\, p(\mathbf{\Phi}_t|\hat{\mathbf{\Phi}}_{t-1})]. \end{aligned} \tag{5}$$

Equation (5) is evaluated using the following two iterative steps: Expectation (*E step*):

$$Q(\mathbf{\Phi}_t|\hat{\mathbf{\Phi}}_t^{(p)}) = E\{\log\, p(\mathbf{X}_t|\mathbf{\Phi}_t)|\mathbf{Y}_t, \hat{\mathbf{\Phi}}_t^{(p)}\} \tag{6}$$

Maximization (*M step*):

$$\hat{\mathbf{\Phi}}_t^{(p+1)} = \arg\max_{\Phi_t}\left[Q(\Phi_t|\hat{\mathbf{\Phi}}_t^{(p)}) + \log\, p(\mathbf{\Phi}_t|\hat{\mathbf{\Phi}}_{t-1})\right] \tag{7}$$

where p represents the pth iteration of the algorithm. The solution of (6) is an evaluation of conditional expectation of $\mathbf{Z}_t$ given $\mathbf{Y}_t$ and $\hat{\mathbf{\Phi}}_t^{(p)}$. The expectation is taken over all the possible association of $\mathbf{Z}_t$. The above steps are iterated till convergence condition is satisfied for each model. This is required as all models are also interlinked through assignment weights and probabilities, which is described in the following section. For simulation, iterations in the algorithm stop when $\|\hat{\phi}_t^{(p)} - \hat{\phi}_t^{(p+1)}\|_2 < \epsilon$ for each model, where ϵ to be chosen is a very small number (for example, 0.01).

With the assumption that $\mathbf{z}_t(i)$'s $(1 \le i \le N_o)$ are independent of $y_t(j)$ for $i \ne j$, and solving (6), it can be shown that the E-step gives assignment wights and assignment probabilities as a by-product [8]. The assignment weight is given by

$$\hat{\mathbf{Z}}_t^{(p)}(s,i) = \frac{\sum_{m=1}^{M}\left[p_m[y_t(i)|\mathbf{z}_t(i)=e_s, \phi_t^{m(p)}(s)]\mu_t^m(s)\right]\pi_t^{(p)}(s)}{\sum_{n=0}^{N_T}\sum_{m=1}^{M}\left[p_m[y_t(i)|\mathbf{z}_t(i)=e_n, \phi_t^{m(p)}(n)]\mu_t^m(n)\right]\pi_t^{(p)}(n)} \tag{8}$$

and the assignment probability for any observation assigned to target t is given by

$$\pi_t^{(p)}(s) = \frac{\sum_{i=1}^{N_o}\hat{\mathbf{z}}_t^{(p)}(s,i)}{N_o} \tag{9}$$

with the following constraint:

$$\sum_{s=1}^{N_T}\pi_t^{(p)}(s) = 1. \tag{10}$$

The disadvantage with the EM algorithm is that it calculates assignment weights and updates state vectors for the targets in an iterative mode. In the proposed neural-network-based approach, assignment weights are obtained from a Hopfield neural network, and these are directly used in a single step to update state vector for the targets.

C. Neural-Network-Based Data Association

A neural network has been used for extracting patterns from the data. This idea can be exploited to evaluate assignment weights by treating observations as the data and observation to track association as a pattern stored in the given data. Using the Hopfield neural network, calculation of assignment weights is described in the following section. Once assignment weights are calculated, these can be used to update the state vector of the target.

The TSP is to find an optimum route for a salesman so that the length of the traveled path is minimized with a constraint that no city is excluded and visited twice. It is a constrained multidimensional optimization problem. Given a set of n cities and distance between each of them, there are $n!/2n$ distinct valid paths, out of which the shortest path is to be chosen. The neural network solution for TSP [13] is as follows. A total of n^2 interconnected neurons is arranged in an $n \times n$ matrix. The output voltage of each neuron is constrained to be between 0 and 1. Every row in the matrix of neurons corresponds to a city and every column represents the order of appearance in the travel path. A valid tour will have only one "1" in every row and column. If V_{si} is an output voltage of the neuron at location (s,i), then the best path is a minimum over all V_{si} of the energy function

$$\begin{aligned}\mathcal{E} = &\frac{A}{2}\sum_s\sum_i\sum_{j\ne i}V_{si}V_{sj} + \frac{B}{2}\sum_s\sum_i\sum_{r\ne s}V_{si}V_{ri} \\ &+ \frac{C}{2}\left(\sum_s\sum_i V_{si} - n\right)^2 \\ &+ \frac{D}{2}\sum_s\sum_i\sum_{r\ne s}d_{sr}V_{si}(V_{r,i+1} + V_{r,i-1})\end{aligned} \tag{11}$$

where d_{sr} is the distance between the cities s and r. Finding the assignment weights $\hat{\mathbf{z}}_t(s,i)$ from the likelihoods $p(y_t(i)|\mathbf{z}_t(i) = e_s, \Phi_t)$ is similar to the TSP. If the output voltages V_{si}'s, $i = 0, 1, \ldots, N_o$, $s = 1, 2, \ldots, N_T$, of a $(N_T+1)\times(N_o+1)$ matrix of neurons are defined as assignment weights, then rows would indicate targets and columns represent observations. The

zeroth column corresponds to no observation from a given target and zeroth row corresponds to false observations. The constraints for the data association are as follows. Each row sum must be unity. At most, one large entry is to be favored in every row and column. We assume that no two observations come from the same target and no observation can come from two targets. Latter assumption helps us to take care of occlusion also. $\hat{\mathbf{z}}_t(s,i)$ should be large if $p(y_t(i)|\mathbf{z}_t(i)=e_s,\mathbf{\Phi}_t)$ $[\equiv p\{y_t(i)|\Phi_t(s)\}]$ is large and there is a combination of the remaining targets and observations that result in a large product of the corresponding likelihoods. All these requirements are fulfilled by minimizing the energy function as described in [13]

$$\mathcal{E}=\min_{V_{is}}\left[\frac{A}{2}\sum_{i=1}^{N_o}\sum_{s=1}^{N_T}\sum_{\substack{r=1\\r\neq s}}^{N_T}V_{si}V_{ri}+\frac{B}{2}\sum_{i=1}^{N_o}\sum_{s=1}^{N_T}\sum_{\substack{j=1\\j\neq i}}^{N_o}V_{si}V_{sj}+\frac{C}{2}\sum_{s=1}^{N_T}\left(\sum_{i=1}^{N_o}V_{si}-1\right)^2+\frac{D}{2}\sum_{i=1}^{N_o}\sum_{s=1}^{N_T}(V_{si}-\rho_{si})^2+\frac{E}{2}\sum_{i=1}^{N_o}\sum_{s=1}^{N_T}\sum_{\substack{r=1\\r\neq s}}^{N_T}\left(V_{si}-\sum_{\substack{j=1\\j\neq i}}^{N_o}\rho_{rj}\right)^2\right]. \tag{12}$$

The ρ_{si}'s are the normalized likelihoods given by

$$\rho_{si}=\frac{\sum_{m=1}^{M}p_m[y_t(i)|\mathbf{z}_t(i)=e_s,\phi_t^{m(p)}(s)]\mu_t^m(s)}{\sum_{j=1}^{N_o}\sum_{m=1}^{M}p_m[y_t(j)|\mathbf{z}_t(j)=e_s,\phi_t^{m(p)}(s)]\mu_t^m(s)} \tag{13}$$

where $i=1,2,\ldots,N_o$ and $s=1,2,\ldots,N_T$. The first two terms of (12) favor only one large entry in every row and column. The third term biases the final solution toward a normalized set of numbers. The fourth term favors associations that have a high likelihood. Minimization of the fifth term requires that V_{si} not be large unless for each $r=s$, there is a unique $j\neq i$ such that the likelihood ρ_{rj} is large. Thus, the fifth term favors a highly likely combination of the remaining targets and observations. Using the Kronecker delta function

$$\delta_{ij}=\begin{cases}1, & \text{if } i=j\\ 0, & \text{if } i\neq j\end{cases} \tag{14}$$

(12) can be written as

$$\mathcal{E}'=-\frac{1}{2}\sum_{i=1}^{N_o}\sum_{j=1}^{N_o}\sum_{s=1}^{N_T}\sum_{r=1}^{N_T}T_{ij}(sr)V_{si}V_{rj}-\sum_{i=1}^{N_o}\sum_{s=1}^{N_T}V_{si}I_{si} \tag{15}$$

where

$$T_{ij}(sr)=-\left[A\delta_{ij}(1-\delta_{sr}+B\delta_{sr}(1-\delta_{ij})+C\delta_{sr}+D\delta_{sr}\delta_{ij}+E(N_T-1)\delta_{sr}\delta_{ij}\right].$$

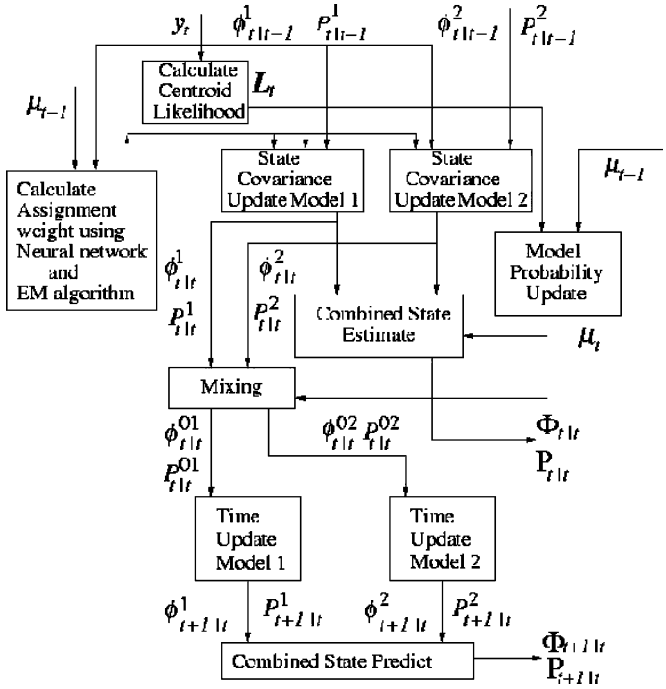


Fig. 1. NN_IMM_EM algorithm for two models.

It represents strengths of connection, and it does not depend on ρ_{si}. I_{si} in (15) is given by

$$I_{si}=C+(D+E)\rho_{si}+E(N_T-1-\sum_{r=1}^{N_T}\rho_{ri}).$$

Thus, a new network need not be designed every time a new set of likelihood function is available. The dynamic equations are as follows:

$$V_{si}=g(u_{si}) \tag{16}$$

$$\frac{du_{si}}{dt}=-\frac{u_{si}}{t_0}-A\sum_{\substack{r=1\\r\neq s}}^{N_T}V_{ri}-B\sum_{\substack{j=0\\j\neq i}}^{N_o}V_{sj}-C\left(\sum_{j=0}^{N_o}V_{sj}-1\right)-[D+E(N_T-1)]V_{si}+(D+E)\rho_{si}+E\left(N_T-1-\sum_{r=1}^{N_T}\rho_{ri}\right). \tag{17}$$

The parameters A,B,C,D, and E can be adjusted depending upon different constraints and properties.

Equations (16) and (17) describe the evolution of the states for the neural network solution to data association problem. The function $g(u_{si})$ in (16) is defined as

$$V_{si}=g(u_{si})=\frac{1}{2}\left(1+\tanh\frac{u_{si}}{u_0}\right) \tag{18}$$

and the inverse of this relation is

$$u_{si}=\frac{u_0}{2}\ln\left(\frac{V_{si}}{1-V_{si}}\right). \tag{19}$$

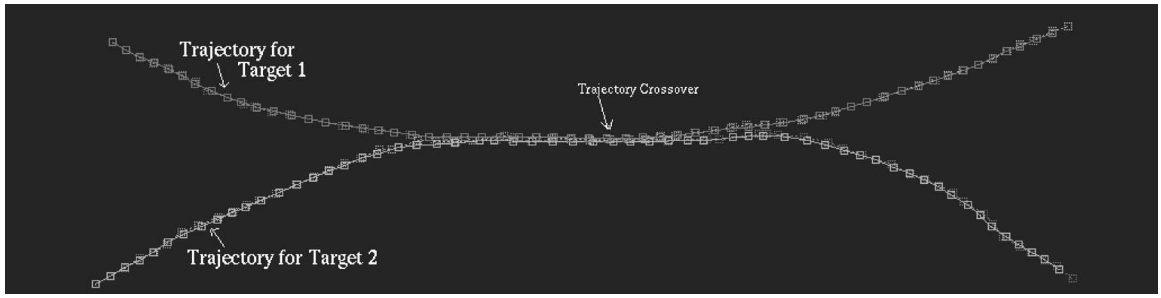


Fig. 2. Trajectory using SMM model for ir44 clip :: crossover.

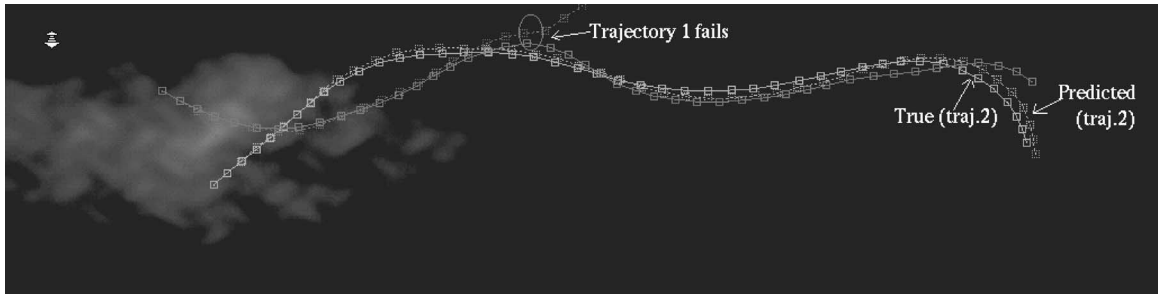


Fig. 3. Trajectory using CA model for ir50 clip.

The initial states may be chosen to be such that they, approximately, produce

$$V_{si} = \frac{1}{N_o + 1} \tag{20}$$

so that $\sum_{i=1}^{N_o} V_{si} = 1, \forall t$ initially. The corresponding initial states may be obtained from (19). In our simulation, the initial value is set as $V_{si} = \rho_{si}$, i.e., initial observation to track likelihoods. The constant u_0 is chosen to be 0.02. The energy function (15) has to be minimized via the dynamic equations (16) and (17). To simulate the performance of the NN-based data association in a digital computer, the differential equation (17) is approximated by a difference equation:

$$\begin{aligned} u_{si}^{(p+1)} = {} & \left(\frac{t_0 - \zeta}{t_0}\right) u_{si}^{(p)} - \zeta A \sum_{\substack{r=1 \\ r \neq s}}^{N_T} V_{ri}^{(p)} - \zeta B \sum_{\substack{j=0 \\ j \neq i}}^{N_o} V_{sj}^{(p)} \\ & - \zeta C \left(\sum_{j=0}^{N_o} V_{sj}^{(p)} - 1 \right) - \zeta [D + E(N_T - 1)] V_{si}^{(p)} \\ & + \zeta (D + E) \rho_{si} + \zeta E \left(N_T - 1 - \sum_{r=1}^{N_T} \rho_{ri} \right) \end{aligned} \tag{21}$$

where t_0 and ζ are selected to be 1 and 0.00001 s, respectively. For our simulations, maximum number of iterations are kept fixed to 500 for the recursive equation (21). The constants A, B, C, D, and E are chosen experimentally to produce a stable operating point. Values $A = 1000, B = 1000, C = 1000, D = 100$, and $E = 100$ are found to be appropriate with 0.05% clutter level. The initial conditions $V_{si} = \rho_{si}$ provide a convenient alternative to the nearly uniform and randomized initial values. The coefficients of the function $\mathcal{E}$ have to be chosen properly for a given number of targets and observations. A proper selection of these coefficients is crucial to the success of the neural-network-based solution. These parameters are selected after extensive simulations and experimentation with large number of video clips. It is important to note that the values of the parameters are found to be independent of the target size, image spatial resolution, temporal resolution, range of target, etc.

D. Proposed NN-IMM-EM Tracking Algorithm

The algorithm is split into three steps. In first step, the assignment weights are calculated using a Hopfield network. In the second step, these assignment weights are used to update the state vector of each model for a given target. It is followed by the prediction step. The logical steps of the proposed algorithm are depicted in Fig. 1. We assume the observation and target state to be Gaussian distributed. For each target s and model m, the state ϕ_t^m is assumed to belong to a Gaussian distribution. The predicted state pdf $p(\phi_t|\hat{\phi}_{t-1})$ is Gaussian with mean $\hat{\phi}_{t|t-1}$ and error covariance $P_{t|t-1}$. With independence assumption for each target and for each model, it can be shown [8] that the state estimate for each model can be obtained by the standard Kalman filter equations using IMM algorithm. The proposed tracking algorithm operates as follows. As the current set of observation $\mathbf{Y}_t$ becomes available, the following two steps are performed at time instant t. The observation set $\mathbf{Y}_t$ is validated using the combined state prediction $\Phi_{t|t-1}(s)$ for target s $(s = 1, \ldots, N_T)$. In the data association step, using a Hopfield neural network assignment the weights are calculated and state

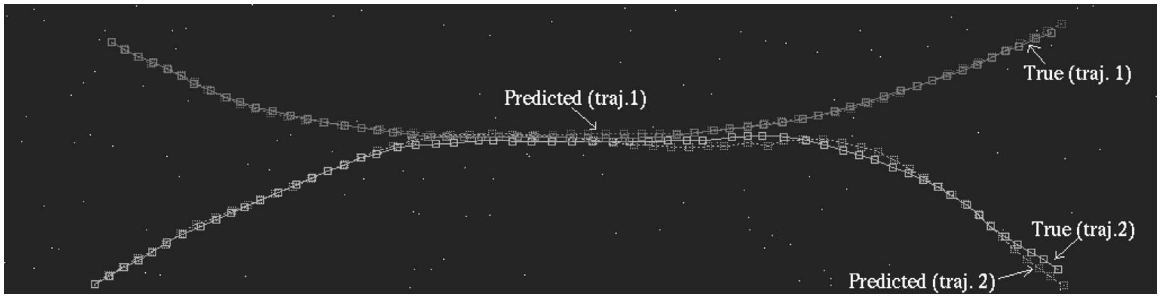


Fig. 4. Trajectory using proposed algorithm for ir44 clip with clutter level 0.05% at frame number 57.

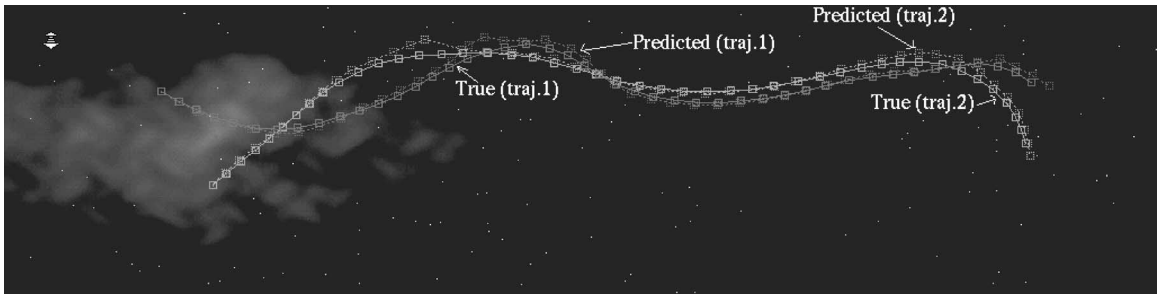


Fig. 5. Trajectory using proposed algorithm for ir50 clip with clutter level 0.05% at frame number 44.

TABLE I
MEAN PREDICTION ERROR IN POSITION FOR CLIPS ir44, ir49, and ir50

Traj. No.	Without clutter	With 0.05% clutter
ir44		
1	1.8730	1.9147
2	3.6895	3.9563
ir49		
1	3.1750	3.5274
2	3.3655	3.3619
ir50		
1	3.6960	3.7714
2	3.5550	3.6575

is updated for each target and for each model. It is followed by an IMM step for each target.

1) NN-based data association and state update step:
   a) For each target $s = 1, \ldots, N_T$ and for each filter model $m = 1, \ldots, M$,
      i) Initialize state $\hat{\phi}^m(s) = \hat{\phi}^m_{t|t-1}(s)$ and covariance $P^m(t) = P^m_{t|t-1}(s)$. $\hat{\phi}^m_{t|t-1}(s)$ and $P^m_{t|t-1}(s)$ represent previously predicted state and covariance, respectively.
      ii) Initialize assignment probability as defined in (8) for the calculation of the effective observation and the likelihood function. This likelihood is used to evaluate model probability in IMM step. It is important to note that this is done only once in the beginning of the algorithm. In our case, assignment probability is initialized uniformly. Actual assignment weights, which are given by (8) and, consequently, assignment probabilities are evaluated by the Hopfield network, as described later.
      iii) Calculate the effective observation for each target s using

$$y^c_t(s) = \frac{\sum_{i=1}^{N_o} \hat{\mathbf{z}}_t(s,i) y_t(i)}{\sum_{i=1}^{N_o} \hat{\mathbf{z}}_t(s,i)} \tag{22}$$

      where $\hat{\mathbf{Z}}_t(s,i)$ is given by (8).
      iv) Calculate the likelihood function (for each model $m = 1, \ldots, M$)

$$\mathcal{L}^m = \mathcal{N}[\tilde{y}^m; 0, S^m]$$

      where

$$\tilde{y}^m = y^c_t(s) - H^m_t \bar{\phi}^m_{t|t-1}(s)$$

      and

$$S^m = H^m_t P^m_{t|t-1}(s)(H^m_t)^T + R^m_t.$$

      It is used to update model probability in the IMM step.
   b) NN-based assignment weight calculation:
      i) Initialize assignment weights $\hat{\mathbf{z}}_t(s,i)$ using (20) or (8) for each target $s = 1, \ldots, N_T$ and for each filter model $m = 1, \ldots, M$.
      ii) Calculate initial neuron input voltage using (19).
      iii) Repeat the following steps at each iteration for predefined number of iterations or till error converges to a fixed threshold value, i.e., $\|\mathcal{E}'^{(p-1)} - \mathcal{E}'^{(p)}\| < \epsilon$, where p represents iteration number.

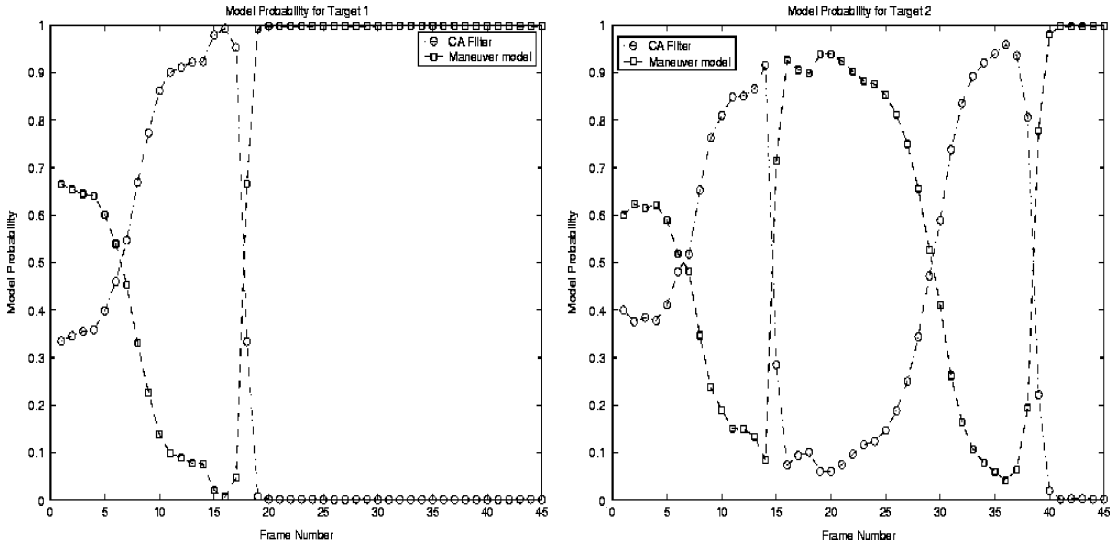


Fig. 6. Model probability (ir50 clip with clutter level 0.05%).

A) Calculate neuron input voltage using (21).
B) Calculate neuron output voltage (assignment weight) as given by (18).
C) Evaluate energy function (15).

c) State and state covariance update (measurement update): A Gauss–Newton method is used to update $\hat{\phi}_t^m \cdot \phi$ is updated by $\Delta\phi$. The $\Delta\phi$ is found by solving $F^{-1}\,\Delta\phi = g$

$$\hat{\phi}_{t|t}^m(s) = \hat{\phi}^m + \Delta\phi$$

where $\Delta\phi$ is derived using

$$F^{-1} = P_{t|t-1}^{-1} + \sum_{i=1}^{N_o} \hat{\mathbf{z}}_t^{(p)}(s,i) H^T R^{-1} H \quad (23)$$

and

$$g = P_{t|t-1}^{-1}(\phi - \hat{\phi}_{t|t-1}) + \sum_{i=1}^{N_o} \hat{\mathbf{z}}_t^{(p)}(s,i) H^T R^{-1}[y_t(i) - h(\phi_t)] \quad (24)$$

where H is the observation gradient $\nabla_\phi h(\phi_t)$. Equation (24) can be simplified further by initializing $\phi = \phi_{t|t-1}$. The state covariance is updated using approximate equation for error covariance update and is given by

$$P_{t|t} \approx F\left[P_{t|t-1}^{-1} + \sum_{i=1}^{N_o} [\hat{\mathbf{z}}_t^{(p)}(s,i)]^2(t) H^T R^{-1} H\right] F^T. \quad (25)$$

At the end of the above step, for each target and for each model, updated state $\hat{\phi}_{t|t}^m(s)$ and updated covariance $P_{t|t}^m(s)$ are obtained.

2) IMM step: For each target $s = 1, \ldots, N_T$, repeat the following steps (for clarity, subscript t for target is dropped from the following equations):

a) Mode probability update:
For each model $m = 1, \ldots, M$, calculate the model probability

$$\mu_t^m = \frac{\mu_{t|t-1}^m \mathcal{L}^m}{\sum_{i=1}^M \mu_{t|t-1}^i \mathcal{L}^i}.$$

b) Combine observation update for state and covariance, respectively

$$\hat{\Phi}_{t|t} = \sum_{m=1}^M \hat{\phi}_{t|t}^m \mu_t^m$$

$$P_{t|t} = \sum_{m=1}^M \left[P_{t|t}^m + (\hat{\Phi}_{t|t} - \hat{\phi}_{t|t}^m)(\hat{\phi}_{t|t} - \hat{\phi}_{t|t}^m)^T\right] \mu_t^m.$$

c) For each model $m = 1, \ldots, M$, calculate the following:

i) model-conditional initialization (mixing) for state and covariance

$$\hat{\phi}^{0m} = \sum_{i=1}^M \hat{\phi}_{t|t}^i \mu^{i|m}$$

$$P^{0m} = \sum_{i=1}^M \left[P_{t|t}^i + (\hat{\phi}^{0m} - \hat{\phi}_{t|t}^i)(\hat{\phi}^{0m} - \hat{\phi}_{t|t}^i)^T\right] \mu^{i|m}$$

where $\mu_{t+1|t}^m = \sum_{i=1}^M \xi_{im} \mu_t^i$ and $\mu^{i|m} = \xi_{im}\mu_t^i / \mu_{t+1|t}^m$. Here, ξ_{im} is the transition probability.

ii) Time update (model-conditional filtering) for state and covariance

$$\phi_{t+1|t}^m = F_t^m \hat{\phi}^{0m}$$

$$P_{t+1|t}^m = F_t^m P^{0m} (F_t^m)^T + Q_t^m.$$

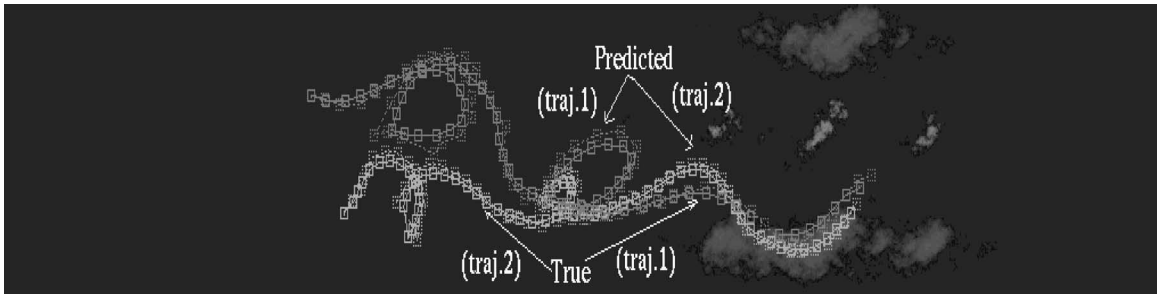


Fig. 7. Trajectory using proposed algorithm for m7_1 clip at frame number 69.

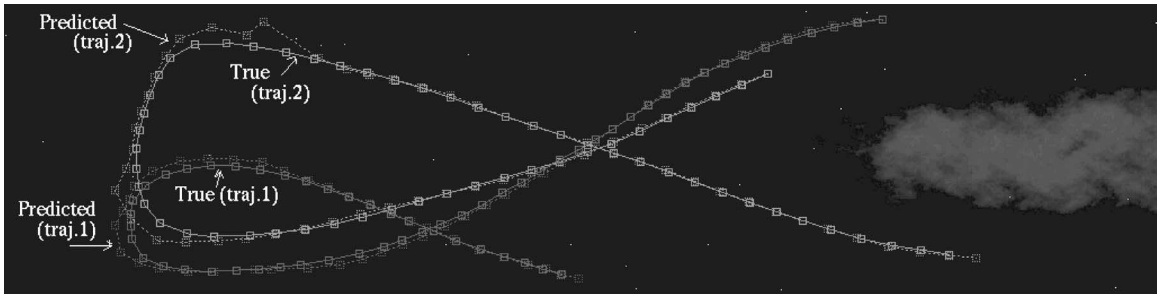


Fig. 8. Trajectory using proposed algorithm for m16 clip with clutter level 0.01% at frame number 59.

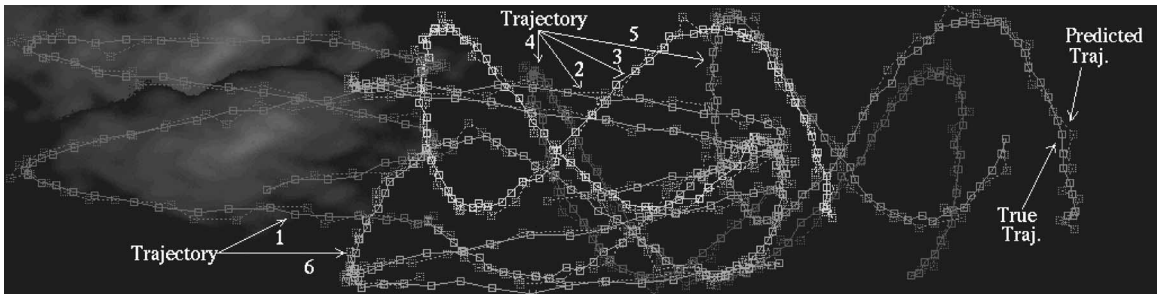


Fig. 9. Target trajectories for in21_1 sequence at frame number 79.

d) Overall target time update for state and covariance

$$\hat{\Phi}_{t+1|t} = \sum_{m=1}^{M} \phi^m_{t+1|t}\mu^m_{t+1|t}$$

$$P_{t+1|t} = \sum_{m=1}^{M} \Big[P^m_{t+1|t} + (\hat{\Phi}_{t+1|t} - \phi^m_{t+1|t}) \times (\hat{\phi}_{t+1|t} - \phi^m_{t+1|t})^T\Big] \mu^m_{t+1|t}.$$

The transition probability is initialized as

$$\Xi = \begin{bmatrix} 0.998 & 0.001 \\ 0.001 & 0.998 \end{bmatrix}$$

and initial model probability is set to $\mu = \{0.5 \quad 0.5\}$.

E. Simulation Results

For evaluation of the proposed algorithm, we have used infrared (IR) clips that are synthetically generated. In SPANN Laboratory, IIT-Bombay, an IR scene simulator [16]–[18] has been developed that allows one to specify type of clouds, number of clouds, number of targets, background intensity, and trajectories for targets. The trajectories are simulated using B-spline functions. This application is developed by using NET framework on a Microsoft platform. Virtual reality modeling language (VRML) is used for scene rendering. For temperature-to-intensity conversion, MODTRAN software is used. For simulation, the generated frame size is 1024×256, the target size is 1×1, and very high target movement of ± 20 pixels per frame. Many IR video clips are simulated with different types of trajectories to evaluate the performance of the proposed algorithm. Two sets of clips have been generated: 1) the first clip set consisting of maneuvering trajectories is generated using B-splines (e.g., ir44, ir49, ir50, etc.), and it is important to note that these generated trajectories do not follow any specific model; 2) for the second clip set, mixed trajectories are generated using constant acceleration (CA) model for nonmaneuvering trajectories and cosine and sine functions for nonlinear (maneuvering) trajectories (e.g., ip4, in23_1, in31_1, etc.). The nonlinear function gives x and y positions of the target at each time t. Extensive

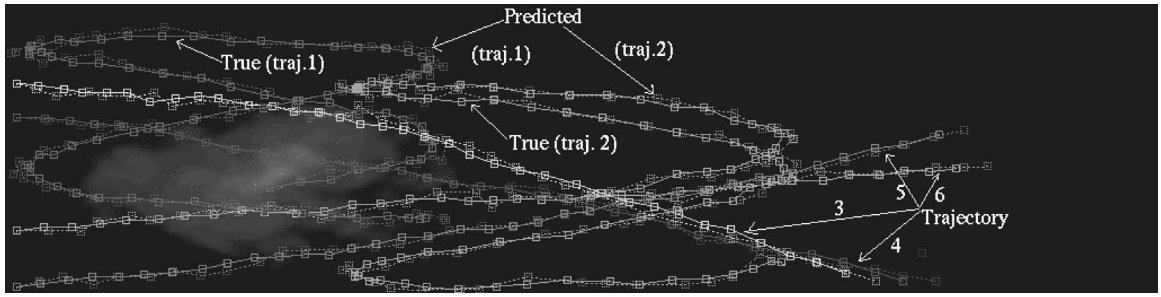


Fig. 10. Target trajectories for in23_1 sequence at frame number 79.

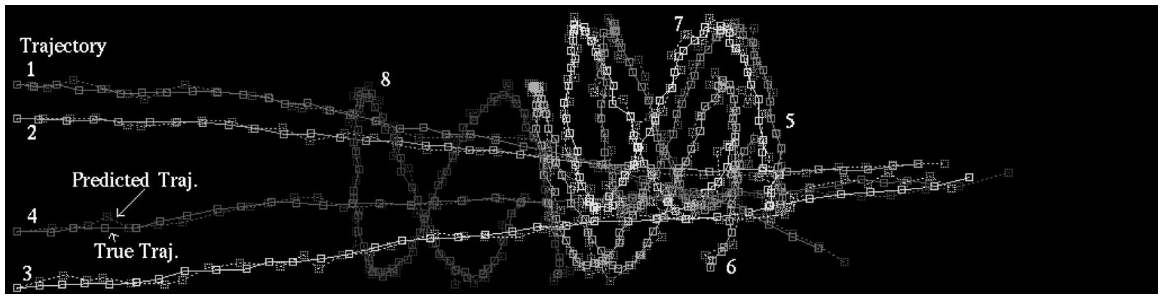


Fig. 11. Target trajectories for in24 sequence at frame number 79.

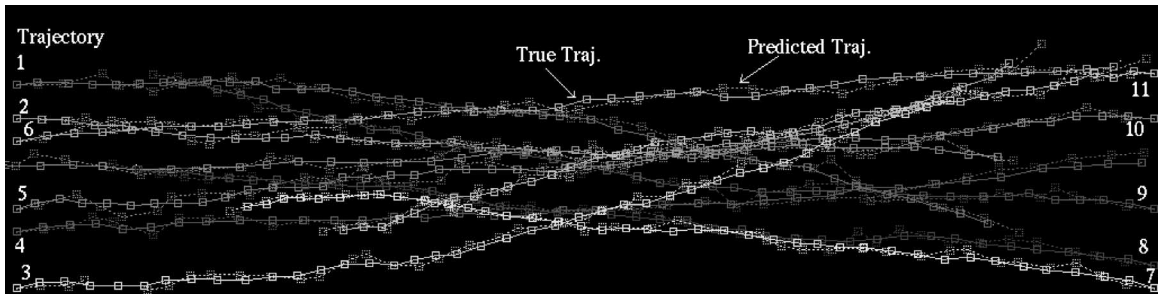


Fig. 12. Target trajectories for ip4 sequence at frame number 39.

simulations have been done and simulation results for a few of the clips from these two different sets are described here. It is assumed that each input clip is processed with the target detection algorithm [19]. At each time instant, the output from the detection module is treated as the observation set. As the tracking is done in an image sequence, the observation consists of x and y positions only. For our case, t is discrete and also represents the frame number in an image sequence. In general, the nonlinear functions are of the following forms:

$$x(t) = \alpha^t + A * \text{tri\_fun}(wt)$$
$$y(t) = B + A * \text{tri\_fun}(wt) \tag{26}$$

where tri_fun may be $\cos$ or $\sin$ function, α takes value less than 1.0, and w is measured in radians.

Different values of the process and observation noise covariances are used: 1) for generation of the trajectories and 2) for the models used in tracking. This facilitates the simulation of mismatch models, thereby providing realistic trajectories to evaluate different tracking algorithms. For generating the nonmaneuvering and maneuvering trajectories using known models (e.g., constant velocity model and nonlinear equations), the process noise covariance and observation noise covariance for the position are set to 5.0 and 5.0. The value of the process noise covariance, for both the velocity and the acceleration of the target, in case of nonmaneuvering trajectories is set to 0.001. In our simulations, we have used CA and Singers' maneuver model [20] (SMM) for tracking. Both the models have six state parameters, namely, position, velocity, and acceleration for x and y positions. For tracking purposes, in our simulations, the model observation noise covariance for the position is set to 9.0 for both the models. For the SMM model, process noise covariance matrix is set according to Singers' model, whereas for CA model, we have considered values ranging from 0.001 to 9.0. For all trajectories, filters are initialized using positions of the targets in the first two frames.

We have compared our proposed tracking algorithm with the technique proposed by Molnar and Modestino [7], which is based on a single model utilizing either SMM or CA. The results of this investigation are depicted in Figs. 2 and 3. Fig. 2

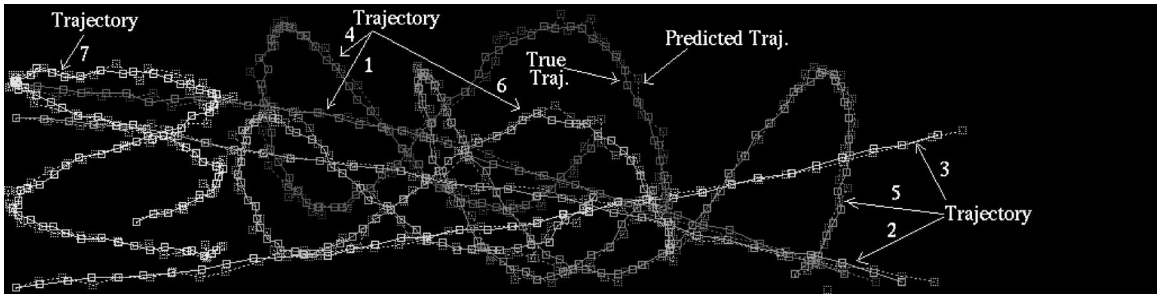
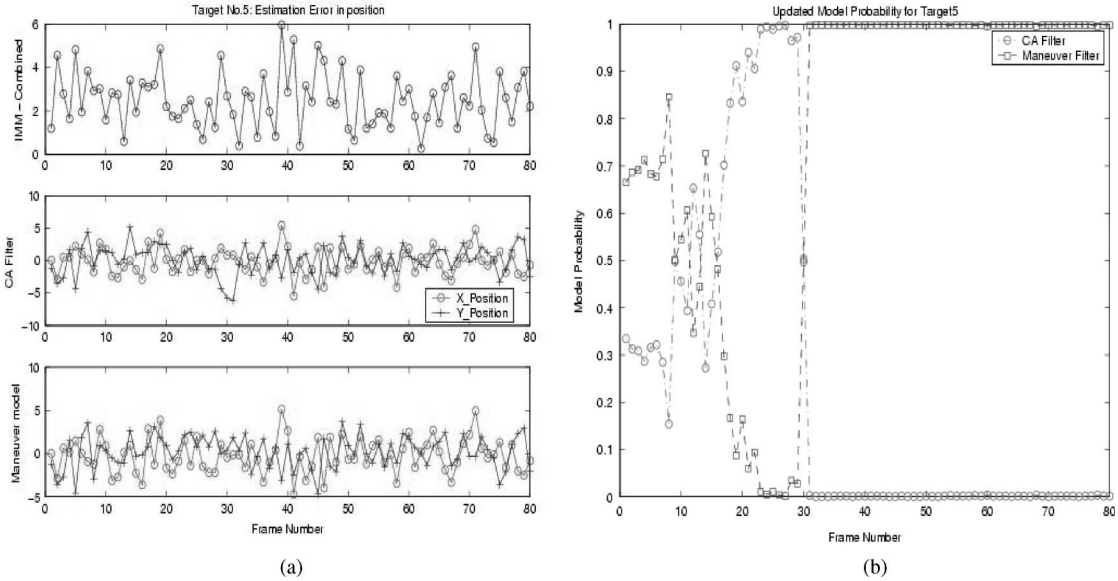


Fig. 13. Target trajectories for in31_1 sequence at frame number 79.



(a) (b)

Fig. 14. Target number 5 in clip in31_1. (a) Estimation error. (b) Model probability.

represents the trajectories tracked in an IR image sequence ir44, using one particular type of model, i.e., SMM. It shows a crossover of trajectories and failure to track targets. Fig. 3 depicts the failure of CA model to track a target in an IR clip ir50. Figs. 4 and 5 present the results for the aforementioned same clips, but with clutter, using our NN-based proposed tracking algorithm. Both these figures show that inclusion of IMM along with neural-network-based assignment weights calculation makes it possible to track an arbitrary trajectory in the presence of a dense clutter. These figures also show that in the presence of multiple targets, NN-based approach provides robust data association.

The percentage of clutter observations is determined based on Gaussian distribution [21]. Note that 0.05% clutter indicates that the 0.05% amount of the total number of pixels in an image are noisy and are treated as clutter (false) observations for tracking purpose. It is assumed that the clutter is uniformly distributed over an image area. The 0.05% clutter level gives three to four false observations on average in the validation region of size 28×28 formed around the predicted position for the tracking. The clutter (false observation) is shown with white dots in Figs. 4 and 5. The mean prediction error in position for clips ir44, ir49, and ir50 is depicted in Table I with and without clutter. Fig. 6 represents the variation with time in the likelihood of a model and, consecutively, the model probability, for different trajectories for ir50 with 0.05% clutter. In Figs. 2–5, the real trajectory is shown with a solid line, whereas predicted trajectory is shown using a dotted line of the same color.

We have also compared our proposed tracking method with our earlier proposed algorithms, namely, tracking using multiple filter bank (MFB) [22], IMM-PMHT [23], and IMM-EM [8]. Our earlier proposed algorithm IMM-EM is successful in tracking trajectories for IR clips, ir44 with 0.05% clutter level, but it fails to track targets in IR ir49 and IR clip ir50 both with 0.05% clutter level. The neural-network-based proposed approach is able to track targets successfully in both IR clips ir49 and ir50 with the same clutter level as described earlier. Similarly, our earlier proposed algorithm MFB is also able to track targets for IR clips ir44 and ir50 with 0.05% clutter level, but it fails to track both targets simultaneously in clip ir49 with 0.05% clutter.

Figs. 7 and 8 represent the results for IR clips m7_1 and m16, respectively. Both these clips contain highly maneuvering targets. Fig. 9 shows the tracked output for clip in21_1, which has six maneuvering trajectories. For clip in23_1 and in24, the tracked trajectories are depicted in Figs. 10 and 11, respectively.

TABLE II
MEAN PREDICTION ERROR IN POSITION FOR VARIOUS CLIPS FROM THE SECOND SET

Traj. No.	Clip in21-1	Clip in23-1	Clip in31-1
1	8.8571	8.2395	2.2631
2	7.8170	7.9452	2.2749
3	6.9168	2.4069	1.8022
4	6.7917	2.2380	6.8472
5	6.6313	1.7724	6.2401
6	6.5022	2.3206	6.4001
7			6.9887

TABLE III
COMPARISON AMONG THE VARIOUS APPROACHES FOR DIFFERENT TRAJECTORIES FOR CLIP in24

No.	Filter Bank		IMM-PMHT	IMM-EM	NN-IMM-EM
	CA	Maneuver			
in24					
1	4.7735	4.6954	5.9343	4.7415	4.8930
2	5.0036	5.3671	8.8255	5.2112	4.8954
3	5.2105	6.0135	13.7852	8.6469	8.5241
4	5.8943	5.1623	8.6341	4.4055	6.3432
5	22.3606	7.7275	fails	11.2904	7.1829
6	23.8804	7.2354	fails	7.4782	7.3049
7	20.0141	7.5430	fails	8.4782	7.4824
8	23.6234	9.0102	7.7677	5.4568	7.6854

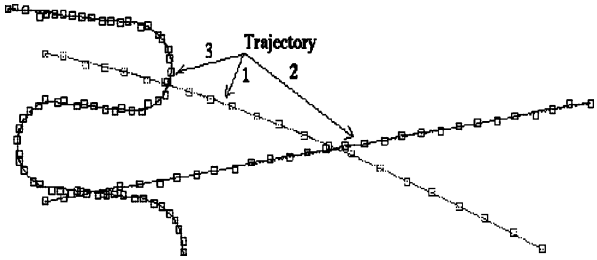


Fig. 15. True trajectory plot for Monte Carlo simulation.

Both clips contain maneuvering and nonmaneuvering targets. Fig. 12 represents tracked trajectories for clip ip4 having 11 targets; all targets are nonmaneuvering. This clip is simulated with half the number of targets moving toward each other and crossing each other.

Fig. 13 represents the result of the proposed tracking algorithm for clip in31_1, which contains seven targets. For target number 5, estimation error in position for each model and IMM-combined is plotted in Fig. 14(a), and evolution of model probability of respective model is shown in Fig. 14(b). Table II depicts mean prediction error in position for various clips from the second set of clips, where trajectories are generated and tracked using a mismatched model, i.e., process and observation noise covariances are different than that of different filters used for tracking. Mean prediction error in position for different trajectories for clip in24 is depicted and compared with that of using other algorithms in Table III. Similarly, Table IV depicts the mean prediction error for clip ip4. It is also compared with mean prediction error for the same trajectories using different algorithms.

TABLE IV
COMPARISON AMONG THE VARIOUS APPROACHES FOR CLIP ip4

No.	Filter Bank		IMM-EM	NN-IMM-EM
	CA	Maneuver		
ip4				
1	6.7715	11.3818	fails	6.2692
2	crossover	crossover	fails	6.6313
3	6.7462	10.8613	7.5339	6.1078
4	5.9693	10.0692	6.3888	5.9516
5	6.9806	10.0943	8.1273	6.8177
6	crossover	crossover	fails	5.8743
7	5.7263	9.6914	fails	5.4651
8	5.8394	10.7906	fails	4.9663
9	6.5709	11.1831	7.8600	6.4540
10	7.1382	10.3015	7.2267	6.9643
11	6.8848	12.0524	8.6793	7.0536

TABLE V
MEAN ESTIMATION ERROR IN POSITION

Traj. No.	NN_IMM_EM	Normalized Error	
		in X	in Y
1	2.0948	0.0710	-0.0011
2	2.2686	-0.1727	0.0537
3	1.9727	0.0537	-0.1550
4	2.7728	-0.0331	0.1597
5	2.4847	-0.0439	0.0093
6	2.5571	-0.1038	0.1111
7	2.6281	0.1204	0.0857

In clip ip4, there are 11 tracks. For this clip, two trajectories crossover occurs with the MFB [22] approach, which uses the NN method for data association. But our proposed algorithm, namely, NN-IMM-EM, is able to track all the targets simultaneously. For the same clip (ip4), the IMM-PMHT algorithm fails to track all targets simultaneously. It is due to the fact that the IMM-PMHT approach avoids uncertainty about the origin of the observation using the centroid of observations and it uses this centroid to update tracks. So, its performance depends upon the gate size used, and the number of observations falling inside the gate affects the calculation of a centroid of observations and results into the failure of tracking multiple targets spaced closely having arbitrary movement. The IMM-EM algorithm [8] is able to track six targets correctly while it looses the remaining tracks. It is due to the assumption that many observations may have originated from the same target, and a large validation gate size may have affected the calculation of the assignment weights. A similar situation is observed in clip in24. In such cases, as depicted in Fig. 12, only the NN-IMM-EM algorithm is able to track all targets simultaneously. We would like to mention that all the clips are compared with the same algorithms. In many cases, other algorithms, except the proposed one fail, and hence, those results are not depicted in the table.

For all clips, prediction position are compared with true position, and these are depicted in tables. Estimation error in position and normalized estimation error in x and y positions for different trajectories for in31_1 are depicted in Table V. We also performed Monte Carlo simulations with a different set of trajectory sets to evaluate the performance of the proposed tracking algorithm. Fifty simulations are performed for a given set of trajectories. The process noise covariance and observation

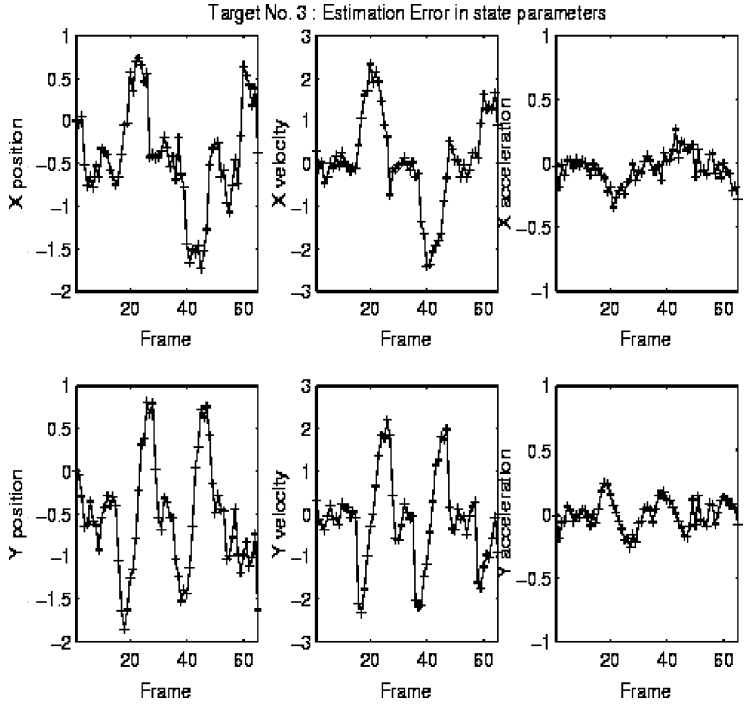


Fig. 16. Estimation error plot for trajectory 3 (Monte Carlo simulation).

noise covariance are set to 0.2 and 2.0, respectively, for trajectory generation. The number of clutter is assumed to be Poisson distributed. The size of the clutter window is 10×10 around the actual observed target position. The average number of clutter falls inside the clutter window is set to 1.

For one of the trajectory sets, the details are as follows. The trajectory set consists of three trajectories. (a) First is a CA trajectory with initial position, velocity, and acceleration set to (70, 70), (20, 3), and (0.5, 0.5). The trajectory exists for 22 frames. The second trajectory is generated using constant velocity model and the trajectory exists for 30 frames. The initial x–y position and velocity are set to (70, 200) and $(20, -3)$. The third trajectory is of "MIX" type. It exists for 70 frames. The initial position and velocity are set to (30, 30) and (10, 1). The target travels with constant velocity from frame 1 to 15. It takes three turns: 1) 15° per second from frame 16 to 27; 2) -15° per second from frame 36 to 47; and 3) 12° per second from frame 58 to 68. Then, target has acceleration of (0.02, 0.02) in x–y. The true trajectory plot is shown in Fig. 15. The estimation error plot for the third trajectory (MIX type) is depicted in Fig. 16.

To determine consistency of the filter, the normalized state estimation error squared (NEES) is evaluated [24]. Under hypothesis H_0 that the filter is consistent and linear Gaussian assumption, it is chi-squared distributed with n_Φ degrees of freedom–dimension of state vector Φ. For Monte Carlo simulation, NEES is averaged over $N = 50$ runs and it is given by

$$\bar{\varepsilon}_t = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_t(i), \qquad N\bar{\varepsilon}_t \sim \chi^2_{Nn_\Phi} \tag{27}$$

where $N\bar{\varepsilon}_t$ will have H_0, a chi-square density with Nn_Φ degrees of freedom. Hypothesis that the state estimation errors are consistent with the filter calculated covariances (also called chi-square test) is accepted if $\bar{\varepsilon}_t \in [r_1, r_2]$, where the acceptance interval is determined such that

$$P\{\bar{\varepsilon}_t \in [r_1, r_2] | H_0\} = 1 - \alpha.$$

In our simulations, one-sided chi-square test is performed with $\alpha = 0.05$ as depicted by (27). With $N = 50$ simulations and two state parameters (position only), the number of degrees of freedom is 100. From the chi-square distribution table, the value found for $r_1 = r/N = 124.342/50$ for one-sided test.

Similarly, to test the bias of the state estimate, which is a vector, test each component of the state estimation error individually. Under the hypothesis that the state estimation is unbiased, and assuming that the error is normally distributed and, therefore, each component, indexed by subscript j, is also normally distributed

$$e(j) = \tilde{\Phi}^j_{t|t} \sim \mathcal{N}[0, P^{jj}_{t|t}]. \tag{28}$$

Each component of the state error is divided by its standard deviation that makes it (under ideal conditions) $\mathcal{N}(0, 1)$, and tested if its mean can be accepted as zero. Test statistic for the normalized mean estimation error (NMEE) for component j of state from runs $i = 1, \ldots, N$ is given by

$$\bar{e}_t(j) = \frac{1}{N}\sum_{i=1}^{N} \frac{\tilde{\Phi}^j_{t|t}(i)}{\sqrt{P^{jj}_{t|t}}}. \tag{29}$$

Under ideal condition, this is distributed as $\mathcal{N}(0, 1/N)$ and two-sided or one-sided test is performed. In our simulations, for each state parameter, two-sided normal test statistic is performed with

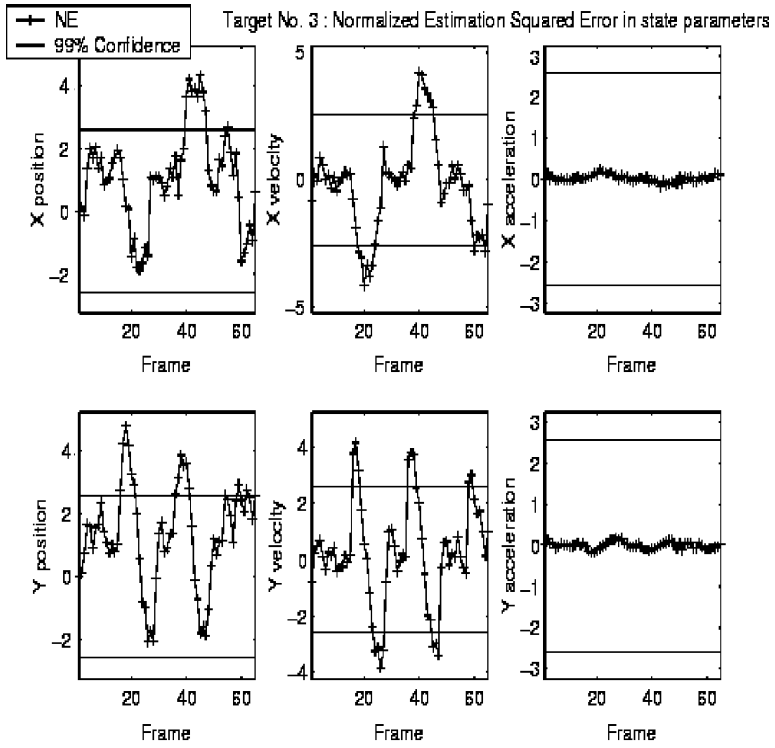


Fig. 17. NEES plot for trajectory 3 (Monte Carlo simulation).

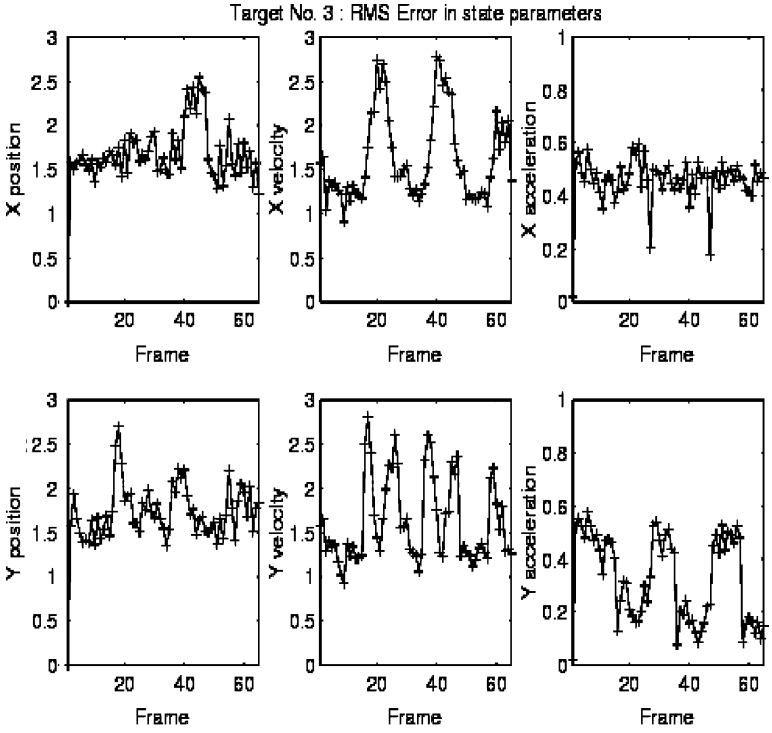


Fig. 18. RMS error plot for trajectory 3 (Monte Carlo simulation).

$\alpha = 0.01$, as given by (29). With normal test statistic, the upper and lower critical values are $(2.58, -2.58)$ for two-sided test. NEES plot for each state parameter is shown in Fig. 17. This confidence interval is marked with black line in the figure. The root mean square (rms) error for each state parameter is plotted in Fig. 18.

After experimenting with large number of clips and detailed Monte Carlo simulations with other algorithms, we concluded that the proposed method performs better than the existing ones (including our own in [8] and [9]).

Some of the reasons for the better performance of the proposed NN-IMM-EM algorithm are as follows.

1) The proposed method uses assignment weights for data association and, hence, avoids the implicit observation to track assignment that is the case for the NN-based data association method. It is well known that implicit observation may result into false track in the presence of a dense clutter.
2) In the case of IMM-PMHT, the centroid of the observations is used for data assignment and it is very sensitive to the size of validation gate and the amount of clutter. In the proposed method, neural network is used to extract the stored pattern from data without using the centroid. Thus, it is not sensitive to the size of the validation gate. Also, it was observed that NN-IMM-EM was robust with respect to the clutter.
3) Because of the learning ability of the neural network, we obtain a more accurate representation of the assignment weights compared to the IMM-EM of [8].

All these results lead to the conclusion that by using our proposed algorithm, it is possible to track arbitrary trajectories in the presence of multiple maneuvering targets and dense clutter.

III. Conclusion

The assignment weight calculations using neural network and EM method make data association robust. It is computationally efficient as state update for the target is not done in an iterative mode. The mixture probability for the observation given state of the target enables us to evaluate assignment weights by incorporating multiple models and, consequently, it is possible to track targets in the presence of the dense clutter. The proposed algorithm is able to track arbitrary trajectory with the inclusion of IMM based on only two filters, namely, CA and SMM, in an IR image sequence. The use of only validated observations to update target state makes the data assignment problem free from the disadvantages of NN- and PMHT-based methods.

References

[1] Y. Bar-shalom and T. E. Fortmann, *Tracking and Data Association*. New York: Academic, 1989.

[2] S. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Norwood, MA: Artech House, 1999.

[3] D. Avitzour, "A maximum likelihood approach to data association," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 28, no. 2, pp. 560–566, Apr. 1992.

[4] R. L. Streit, "Maximum likelihood method for probabilistic multihypothesis tracking," in *Proc. SPIE Conf. Signal Data Process. Small Targets*, vol. 2235, Jul. 1994, pp. 394–405.

[5] P. Willett, Y. Ruan, and R. Streit, "The PMHT for maneuvering targets," in *Proc. SPIE Conf. Signal Data Process. Small Targets*, vol, 3373, Jul. 1998, pp. 416–427.

[6] ——, "PMHT: Problems and some solutions," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 3, pp. 738–754, Jul. 2002.

[7] K. J. Molnar and J. W. Modestino, "Application of the EM algorithm for the multitarget/multisensor tracking problem," *IEEE Trans. Signal Process.*, vol. 46, no. 1, pp. 115–129, Jan. 1998.

[8] M. A. Zaveri, U. B. Desai, and S. Merchant, "Interacting multiple model based tracking of multiple point targets using Expectation Maximization algorithm in infrared image sequence," in *Proc. SPIE: Vis. Commun. Image Process. (VCIP 2003)*, vol. 5150, Lugano, Switzerland, Jul., pp. 303–314.

[9] M. A. Zaveri, S. N. Merchant, and U. B. Desai, "Interacting multiple model based tracking of multiple point targets using expectation maximization algorithm in infrared image sequence," *Opt. Eng.*, vol. 44, no. 1, Jan. 2005.

[10] T. Kirubarajan, M. Yeddanapudi, Y. Bar-shalom, and K. Pattipai, "Comparison of IMMPDA and IMM-assignment algorithms on real traffic surveillance data," in *Proc. SPIE Conf. Signal Data Process. Small Targets*, vol, 2759, May 1996, pp. 453–464.

[11] R. E. Helmick and G. A. Watson, "IMM-IPDAF for track formation on maneuvering targets in cluttered environments," in *Proc. SPIE Conf. Signal Data Process. Small Targets*, vol. 2235, Jul. 1994, pp. 460–471.

[12] X. R. Li and Y. Zhang, "Numerically robust implementation of multiple-model algorithms," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 1, pp. 266–277, Jan. 2000.

[13] D. Sengupta and R. A. Iltis, "Neural solution to the multitarget tracking data association problem," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 25, no. 1, pp. 96–108, Jan. 1989.

[14] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Process. Mag.*, vol. 13, no. 6, pp. 47–60, Nov. 1996.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likeklihood from incomplete data via the EM algorithm," *J. R. Soc. Stat. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[16] S. T. More, A. A. Pandit, S. Merchant, and U. Desai, "Synthetic IR scene simulation of air-borne targets," in *Proc. 3rd Conf. ICVGIP 2002*, Ahmedabad, India, Dec., pp. 108–113.

[17] A. Pandit, "Image rendering in IR scence simulation" M.Tech. thesis, Indian Inst. Technol., Bombay, India, Jan. 2003.

[18] S. More, "Synthetic IR scene simulation of air borne targets" M.Tech. thesis, Indian Inst. Technol., Bombay, India, Jan. 2003.

[19] M. A. Zaveri, S. Merchant, and U. B. Desai, "Multiple single pixel dim target detection in infrared image sequence," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS 2003)*, vol. 2, Bangkok, May, pp. 380–383.

[20] R. A. Singer, "Estimating optimal tracking filter performance for manned maneuvering targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-6, no. 4, pp. 473–483, Jul. 1970.

[21] F. Gini, M. V. Greco, A. Farina, and P. Lombardo, "Optimum and mismatched detection against K-distributed plus Gaussian clutter," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 34, no. 3, pp. 860–876, Jul. 1998.

[22] M. A. Zaveri, U. B. Desai, and S. Merchant, "Tracking multiple maneuvering point targets using multiple filter bank in infrared image sequence," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP 2003)*, vol. 2, Hong Kong, Apr., pp. 409–412.

[23] ——, "PMHT based multiple point targets tracking using multiple models in infrared imagesequence," in *Proc. IEEE Conf. Adv. Video Signal Based Surveillance (AVSS 2003)*, Miami, FL, Jul., pp. 73–78.

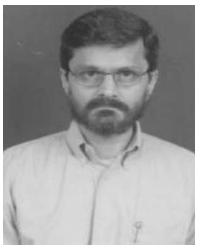
[24] Y. Bar shamlom, X.-R. Li, and T. Kirubarajan, *Estimation With Applications to Tracking and Navigation*. New York: Wiley, 2001.

Mukesh A. Zaveri received the B.E. degree in electronics engineering from Sardar Vallabhbhai Regional College of Engineering and Technology, Surat, India, in 1990, the M.E. degree in electrical engineering from Maharaja Sayajirao University, Baroda, India, in 1993, and the Ph.D. degree in electrical engineering from the Indian Institute of Technology–Bombay, Mumbai, in 2005.

He is currently an Assistant Professor and the Head of the Computer Engineering Department, Sardar Vallabhbhai National Institute of Technology. His current research interests include the area of signal and image processing, multimedia, computer networks, sensor networks, and wireless communications.

S. N. Merchant received the B.Tech., M.Tech., and Ph.D. degrees from the Indian Institute of Technology (IIT)-Bombay, Mumbai, all in electrical engineering.

He has served as a Chief Investigator on a number of sponsored and consultancy projects as a Principal Research Scientist in the Advanced Centre for Research in Electronics, IIT-Bombay, where he is currently a Professor in the Department of Electrical Engineering. His current research interests include the areas of signal and image processing, imaging and its application, and multimedia and wireless communications. He has served as a consultant in both private industries and defense organizations.



Uday B. Desai (S'75–M'78–SM'96) received the B.Tech. degree from the Indian Institute of Technology (IIT)-Kanpur, Kanpur, in 1974, the M.S. degree from the State University of New York, Buffalo, in 1976, and the Ph.D. degree from the Johns Hopkins University, Baltimore, MD, in 1979, all in electrical engineering.

From 1979 to 1984, he was an Assistant Professor in the Electrical Engineering Department, Washington State University, Pullman, where he was an Associate Professor from 1984 to 1987. Since 1987, he has been a Professor in the Electrical Engineering Department, IIT-Bombay, Mumbai, where he was the Dean of Students from August 2000 to July 2002. He has also been a Visiting Associate Professor at Arizona State University, Tempe, Purdue University, West Lafayette, IN, and Stanford University, Stanford, CA. He was a Visiting Professor at the Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, during summer of 2002. From July 2002 to June 2004, he was the Director of HP-IITM R&D Laboratory, IIT-Madras, Chennai.