Cross-Entropy Based Data Association for Multi Target Tracking

Daniel Sigalov
Technion – Israel Institute of Technology
Haifa, Israel, 32000
dansigal@tx.technion.ac.il

Nahum Shimkin
Technion – Israel Institute of Technology
Haifa, Israel, 32000
shimkin@ee.technion.ac.il

ABSTRACT

Multiple-target tracking (MTT) in the presence of spurious measurements poses difficult computational challenges related to the measurement-to-track data association problem. Different approaches have been proposed to tackle this problem, including various approximations and heuristic optimization tools. The Cross Entropy (CE) and the related Parametric MinxEnt (PME) methods are recent optimization heuristics that have proved useful in many combinatorial optimization problems. They are akin to evolutionary algorithms in that a population of solutions is evolved, however the solution improvement mechanism is based on statistical methods of sampling and parameter estimation. In this work we apply the Cross-Entropy method and its recent MinxEnt variants to solve approximately the multi-scan version of the data association problem in the presence of misdetections, false alarms, and unknown number of targets. We formulate the algorithms, and explore via simulation their efficiency and performance compared to other recently proposed algorithms.

Categories and Subject Descriptors

G.3 [Probability and Statistics]: Probabilistic algorithms

General Terms

Algorithms, Performance

Keywords

data association, target tracking, heuristic optimization, cross-entropy method, Monte-Carlo methods

1. INTRODUCTION

Multiple-target tracking (MTT) is an essential component of surveillance-related systems. A general formulation of the problem assumes an unknown and varying number of targets continuously moving in a given region. In the single-sensor version, the states of these targets are sampled by the sensor and the noisy measurements are provided to the tracking system. The detection probability is not perfect and the targets may go undetected at some sampling intervals. In addition, there are spurious reports of possible targets, or clutter measurements which arise independently of the targets of interest. A primary task of the MTT system is data association, namely, partitioning the measurements into disjoint sets, each generated by a single source (target or clutter). The secondary goal is estimation of the states based on the measurements originating from the targets of interest. The data association problem may be formulated in several ways. In single scan data association the raw measurements are processed one scan at a time and the target states are updated accordingly. Alternatively, several sets of measurements may be collected and processed together in batch mode – this is the multi scan data association. For an illustration see Fig. 2.

Several methods exist to handle the data association problem. These may be roughly grouped into two types: Bayesian and non-Bayesian. Among the Bayesian methods, there is the well known Joint Probabilistic Data Association Filter (JPDA) [1], which is a single scan filter where the states of existing targets are to be updated based on the latest set of measurements (scan). Data association is handled by summing over the probabilities of all feasible partitions, where no two targets can share a measurement and each target may be a source of at most one measurement per scan. A shortcoming of the basic JPDA is its inability to initiate and terminate tracks. In addition, calculating the probabilities of all feasible events is NP-hard [4] in the number of targets and measurements and the calculation becomes intractable even for a moderate size of the problem. Another well known approach is the multiple hypotheses tracker (MHT) [10], in which each hypothesis associates past observations with targets and, as a new set of observations arrives, a new set of hypotheses is formed by augmenting the previous ones. The hypothesis with the highest posterior is returned as a solution. The MHT is capable of initiating and terminating tracks. However, the number of hypotheses involved in the calculation grows exponentially over time. Thus, in order to overcome the complexity, several pruning and clustering methods are used at the expense of optimality.

The non-Bayesian approach is characterized by hard measurement to track association, such that some cost function is maximized. The problem may then be reformulated as an integer programming problem [7] or, more precisely, as a multidimensional assignment problem, which is NP-hard when the number of sets (scans) to be assigned is

ValueTools 2008 , October 21 − 23, 2008, Athens, GREECE.

greater than 3 [6]. Therefore, for the multi scan data association, one should invoke some approximations schemes for the multidimensional assignment such as Lagrangian relaxation techniques [5]. Note, however, that when there are only two sets of data to be assigned, there exist exact, polynomial time solutions which have been combined with particle filter based algorithms in the context of multi-target tracking [8].

Another option to solve the multi scan data association problem is by utilizing stochastic search methods. In [9] the problem was solved by applying the Markov Chain Monte Carlo (MCMC) method to obtain the partition with maximum posterior. Using the Metropolis algorithm, the authors proposed a set of moves for modifying a partition of the measurements, such that sampling from the posterior distribution was possible after a few thousands of moves. They showed a remarkable performance of the algorithm in comparison to the MHT method in terms of accuracy of the solution and running time. However, the algorithm is still susceptible to getting trapped in a strong local maxima. Such behavior is typical of local search algorithms.

The main contribution of this paper is the development of feasible algorithms that solve the multi scan data association problem and are capable of initiating and terminating a varying number of tracks. While the general setup and problem definition are very similar to those in [9], the solution approach is different. We invoke the Cross Entropy (CE) method [13] and the related Parametric MinxEnt (PME) method [14] in order to obtain the partition with the highest posterior. CE based schemes are approaches for combinatorial and continuous optimization and for estimation of rare-events probabilities. They are inherently global search methods and, therefore, reduce the risk of getting into a local maxima. The main idea is representing the solution space with a set of parameters and defining a probability distribution on their values. Then, two successive steps are iterated – sampling from the existing distribution, and updating this distribution using a subset of elite (better-valued) samples. The underlying principle of solution improvement is thus akin to evolutionary optimization algorithms (see, e.g. [16]), but the solution generation mechanism is different, and the whole scheme has very few meta-parameters that need to be tuned. The resulting Cross Entropy Data Association (CEDA) and Parametric MinxEnt Data Association (PMEDA) algorithms are applied to challenging tracking scenarios, and show improved performance relative to current state-of-the-art techniques, and in particular relative to the results reported in [9].

The structure of this paper is as follows. We formally state the (discrete-time) general multiple-target tracking problem in section 2. In section 3 we outline the CE and PME methods for combinatorial optimization. In section 4 we present general purpose CEDA and PMEDA algorithms for multiple target tracking. The algorithms are applied in simulation to hard tracking scenarios and their performance is compared with several popular algorithms in section 5.

2. PROBLEM DEFINITION

2.1 Preliminaries

Consider a surveillance scenario of duration $T \in \mathbb{Z}^+$. There are K targets moving around the surveillance region $\mathcal{R}$ for some duration $[t_i^k, t_f^k] \subset [1, T]$ for $k = 1, \ldots, K$ where K is an unknown integer. The volume of $\mathcal{R}$ is V and it is scanned periodically by a single sensor having scan period T_s normalized to one time unit. The notation $[t_i, t_j]$ should be interpreted as $\{t_i, t_i + 1, ..., t_j\}$.

2.2 Target Model

In this subsection we describe the target modeling commonly used in the target tracking literature (see e.g. [2]). Each target k starts at a random position in $\mathcal{R}$ at time t_i^k, moves around $\mathcal{R}$ until t_f^k and disappears. An existing target may disappear at each sampling time with probability p_z and persists with probability $1 - p_z$. The number of new objects arising at each time in $\mathcal{R}$ is modeled to have a Poisson distribution with a parameter $\lambda_b V$, where λ_b is the birth rate of new targets per unit time and volume. The initial position of a new target is uniformly distributed over $\mathcal{R}$. We describe the motion of an object by the discrete-time dynamics $F : \mathbb{R}^d \to \mathbb{R}^d$, where d is the dimension of the state variable, and $x_t \in \mathbb{R}^d$ is the state at time t. The object k moves according to $x_{t+1}^k = F(x_t^k, w_t^k),\ t = t_i^k, \ldots, t_f^k - 1$ where $w_t^k \in \mathbb{R}^d$ are white noise processes. In this work, we consider the same linear dynamical model for each target, namely, if a target is observed l times at $t_i,\ i = 1, ..., l$, its dynamic model may be expressed as:

$$x_{t_{i+1}} = A(t_{i+1}, t_i)x_{t_i} + G(t_{i+1}, t_i)w_{t_i} \qquad (1)$$

where w_{t_i} is a white Gaussian noise with covariance matrix Q. A and G are matrices of appropriate sizes, with entries determined by the sampling interval (t_i, t_{i+1}) for each i.

2.3 Sensor and Measurement Models

We assume that a single sensor scans the surveillance region periodically with scan time T_s of one time unit. Noisy observations of the position of each object are obtained with detection probability P_d. In addition, the sensor generates false alarms, whose number is assumed to have a Poisson distribution with parameter $\lambda_f V$, where λ_f is the false alarm rate per unit time per unit volume. The origin of each observation (i.e. target or false alarm) is not a-priori known, since each observation is assumed to carry only the cartesian position and the corresponding time tag.

Let n_t be the number of observations at time t, including both noisy observations and false alarms. Let $y_t^j \in \mathbb{R}^m$ denote the j-th observation at time t for $j = 1, \ldots, n_t$, where m is the dimensionality of each observation vector. Each target generates a unique observation at each sampling time if it is detected. We assume a linear observation model, namely, an arbitrary observation at time t_i, $y_{t_i}^j$, is generated as follows:

$$y_{t_i}^j = \begin{cases} C(t_i)x_{t_i} + v_{t_i}, & y_{t_i}^j \text{ is object originated} \\ u_t, & \text{otherwise} \end{cases}, \qquad (2)$$

where $v_{t_i} \in \mathbb{R}^m$ is a white Gaussian noise independent of w_{t_i} with covariance matrix R, C is a matrix of appropriate size, and $u_t \sim \text{Unif}(\mathcal{R})$ is the random process of false alarms, assumed to be uniformly distributed in space.

2.4 Solution Space and Optimization Criteria

Dealing with hard data association we seek for a partition of the measurements into disjoint sets. One of these sets is the collection of false alarms and the others are collections of measurements originating from the same target – one set

per target. Let $Y_t = \{y_t^j : j = 1, \ldots, n_t\}$ be the set of observations at time t, and $Y_{1:T} = \bigcup_{t\in\{1,\ldots,T\}} Y_t$ be the set of all observations. Ω is defined to be the set of partitions of $Y_{1:T}$ such that, for $\omega \in \Omega$:

1. $\omega = \{\tau_0, \tau_1, \ldots, \tau_K\}$,
2. $\bigcup_{k=0}^{K} \tau_k = Y_{1:T}$ and $\tau_i \cap \tau_j = \emptyset$ for $i \neq j$,
3. τ_0 is considered as the set of false alarms and $\tau_k,\ k \geq 1$ is considered as the kth track – a set of measurements that are attributed to the kth target[1].
$$\tau_i = \left\{ y_{t_1}^{i_1}, y_{t_2}^{i_2}, \ldots, y_{t_j}^{j} \mid j \in \mathbb{Z}^+, t_1 < t_2 < \ldots < t_j \right\}$$
4. $|\tau_k \cap Y_t| \leq 1$ for $k = 1, \ldots, K$ and $t = 1, \ldots, T$. That is, each measurement belongs to one track at most.
5. $|\tau_k| \geq 2$ for $k = 1, \ldots, K$, where $|\tau_k|$ denotes the cardinality of τ_k. That is, a track must contain at least two measurements.

We make two additional assumptions as part of the problem formulation.

A. The maximal velocity of any target is bounded by a known constant $v_{\max}$.

B. The number of consecutive missing observations of any track is bounded by a known constant $d_{\max}$. This assumption may be used as a criterion to distinguish an event of a new target's appearance from an event of a continuation of an existing target.

For further discussion of the last two restrictions the reader is referred to [9]. A partition $\omega \in \Omega$ is said to be valid or feasible. Once a partition $\omega \in \Omega$ is chosen, the tracks $\tau_1, \ldots, \tau_K \in \omega$ and the set of false alarms $\tau_0 \in \omega$ are completely determined. We thus face a problem of choosing the best (in some sense) partition ω^* given the set of observations $Y_{1:T}$. This is the so-called measurement oriented approach to data association. A natural criterion for this approach is to consider the Maximum a-Posteriori probability [2, 9, 10]. That is, looking for the optimal partition ω in the MAP sense:

$$\omega^* = \arg\max_{\omega\in\Omega} \mathbb{P}\left\{\omega \mid Y_{1:T}\right\}. \tag{3}$$

2.5 Cost Function

The expression for the posterior probability is commonly used in the target tracking literature [2, 1, 9, 10] and the complete derivation is omitted due to space limitations. Applying Bayes rule to (3) one obtains,

$$\omega^* = \arg\max_{\omega\in\Omega} p\left(Y_{1:T} \mid \omega\right) \mathbb{P}\left\{\omega\right\}.$$

For each partition ω we define m_t to be the number of targets at time t, a_t – the number of new targets at time t, z_t – the number of targets terminated at time t, d_t – the number of target detections at time t, u_t – the number of undetected targets at time t, and f_t – the number of false alarms at time t. Bearing in mind that n_t is the total number of measurements obtained at time t, it may be easily verified that the following relations hold: $m_t = m_{t-1} + a_t - z_t$, $u_t = m_t - d_t$, $f_t = n_t - d_t$. The prior probability of a partition is determined by the above and the clutter and new targets models. The likelihood accounts for the goodness of fit of the measurement to a given partition based on the target and measurement models. The final expression reads,

$$\mathbb{P}\left\{\omega \mid Y_{1:T}\right\} \propto \prod_{\tau\in\omega\setminus\{\tau_0\}} \prod_{i=2}^{|\tau|} \mathcal{N}\left(\tau(t_i); \hat{y}_{t_i}(\tau), B_{t_i}(\tau)\right) \tag{4}$$
$$\cdot \prod_{t=1}^{T} p_z^{z_t} (1-p_z)^{m_{t-1}-z_t} \cdot P_d^{d_t} (1-P_d)^{u_t} \lambda_b^{a_t} \lambda_f^{f_t},$$

where $\mathcal{N}(x; \mu, \Sigma)$ is the Gaussian density with mean μ and covariance Σ evaluated at x, $\tau(t_i)$ is the i-th measurement associated with track τ, $\hat{y}_{t_i}(\tau)$ is the i-th predicted measurement obtained from the standard Kalman Filter applied to the measurements associated with track τ, and B_{t_i} is the corresponding innovation covariance. For an extensive discussion and derivation of (4) the reader is referred to [15].

3. BACKGROUND ON CE AND PME

3.1 The CE Method for Combinatorial Optimization

Let $\mathcal{X}$ be a finite set of elements and $S(\cdot)$ be a performance function defined on $\mathcal{X}$. Our goal is to find the maximum of $S(\cdot)$ over $\mathcal{X}$. Namely,

$$S(\mathbf{x}^*) = \gamma^* = \max_{\mathbf{x}\in\mathcal{X}} S(\mathbf{x}). \tag{5}$$

A convenient way to introduce the CE method is from the parameter estimation perspective. When solving optimization problems using the CE method, one searches for a probability distribution concentrated near the global extremum of the objective function. Assume we can define a parameterized probability density function $f(\mathbf{x}; \mathbf{v})$ on the set $\mathbf{x} \in \mathcal{X}$. The goal is to construct a sequence of parameter vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots.$ such that $f(\mathbf{x}; \mathbf{v}_t)$ becomes concentrated around the global optimum $\mathbf{x}^*$ as t increases. This goal is achieved by sampling from $f(\mathbf{x}; \mathbf{v}_t)$ and constructing the next parameter vector $\mathbf{v}_{t+1}$ as the Maximum Likelihood estimate of the distribution parameter based on the elite samples. Namely,

$$\hat{\mathbf{v}}_{t+1} = \arg\max_{\mathbf{v}} \ln f(\tilde{\mathbf{X}}_1, ..., \tilde{\mathbf{X}}_{N\rho}; \mathbf{v}), \tag{6}$$

where $\tilde{\mathbf{X}}_1, ..., \tilde{\mathbf{X}}_{N\rho}$ are the $N\rho$ elite samples, achieving the best performance in the current set, and $f(\tilde{\mathbf{X}}_1, ..., \tilde{\mathbf{X}}_{N\rho}; \mathbf{v})$ is the joint density evaluated at $\tilde{\mathbf{X}}_1, ..., \tilde{\mathbf{X}}_{N\rho}$. The new parameter vector defines a new distribution from which we can sample again and repeat the procedure. Instead of updating the parameter vector $\mathbf{v}_t$ directly via the solution of (6) one may use the smoothed update which reduces the probability that some components of $\mathbf{v}_t$ will become degenerate at early stages,

$$\hat{\mathbf{v}}_t = \alpha \tilde{\mathbf{v}}_t + (1-\alpha)\hat{\mathbf{v}}_{t-1},\ 0 \leq \alpha \leq 1, \tag{7}$$

where $\tilde{\mathbf{v}}_t$ is the solution obtained from (6). The whole procedure is summarized in Alg. 1. The stopping criteria in step 5 of Alg. 1 may be lack of significant improvement for several iterations, or convergence to a degenerate distribution.

[1]We shall refer to the above set of measurements as a track, although usually a track is an estimated trajectory, that is after filtering (or smoothing) out the measurement noise.

Algorithm 1 The CE Algorithm for Optimization.

1: Define $\hat{\mathbf{v}}_0 = \mathbf{u}$. Set $t = 1$ (level counter)
2: Generate $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from $f(\cdot; \mathbf{v}_{t-1})$ and compute the sample $(1-\rho)$-quantile $\hat{\gamma}_t$ of the performances.
3: Find the MLE of the new parameter $\mathbf{v}_t$ based on the set of the elite samples. Namely, solve (6).
4: Smooth the estimate via (7).
5: If stopping criteria are met - stop, otherwise set $t = t+1$ and reiterate from step 2.

Assume now that $\mathbf{X} = (X_1, ..., X_n)$ is a random vector such that each X_i is a discrete random variable that can assume a finite number of values $\{a_1, ..., a_m\}$ and let

$$v_{jk} \triangleq \mathbb{P}\{X_j = a_k\} = \mathbb{E}_{\mathbf{v}}\left[\mathbb{I}_{\{X_j = a_k\}}\right]$$

be the components constituting the parameter vector $\mathbf{v}$. The important observation that makes the CE method very easy to apply to various optimization problems, such as the Traveling Salesperson and MaxCut [6, 13], is that in this case there is a simple componentwise analytical solution to (6) that reads [13]

$$\hat{v}_{jk} = \frac{\sum_{i=1}^N \mathbb{I}_{\{X_{ij} = a_k\}} \mathbb{I}_{\{S(\mathbf{X}_i) \geq \hat{\gamma}_t\}}}{\sum_{i=1}^N \mathbb{I}_{\{S(\mathbf{X}_i) \geq \hat{\gamma}_t\}}}, \tag{8}$$

where X_{ij} is the j-th element of the i-th sample $\mathbf{X}_i$ drawn from $f(\mathbf{x}, \mathbf{v}_{t-1})$. Namely, the updated value of each parameter is the relative frequency of the appearance of the corresponding value in the current elite sample.

3.2 The PME Method for Combinatorial Optimization

Recall that the goal of the CEM was to find a "good" sampling density concentrated near the global optimum of the problem at hand. Another option is to consider the (single constrained) Minimum Cross Entropy (MinxEnt) program that reads

$$\min_{f(\mathbf{x})} \left\{ \mathcal{D}(f|h) = \int \ln \frac{f(\mathbf{x})}{h(\mathbf{x})} f(\mathbf{x})\, d\mathbf{x} = \mathbb{E}_f \left[\ln \frac{f(\mathbf{X})}{h(\mathbf{X})} \right] \right\}$$

subject to the first moment constraint:

$$\mathbb{E}_f S(\mathbf{X}) = \gamma, \ \int f(\mathbf{x})\, d\mathbf{x} = 1, \ \int h(\mathbf{x})\, d\mathbf{x} = 1, \tag{9}$$

where f and h are n-dimensional pdf's, $S(\mathbf{x})$ is the known performance function, $\mathbf{x} \in \mathbb{R}^n$, and γ is a performance close to the optimal γ^*. Assuming $h(\cdot)$ is a known pdf that incorporates all the available information about $g^*(\cdot)$, the problem is to find the closest to $h(\mathbf{x})$ density $f(\cdot)$ in the Kullback-Leibler sense subject to the moment constraint. If no prior information is available, $h(\mathbf{x})$ is taken to be uniform. We shall restrict ourselves to the discrete distributions $f(\mathbf{x})$ and $h(\mathbf{x})$ parameterized by parameter vectors $\mathbf{v}, \mathbf{u}$ respectively – $f(\mathbf{x}, \mathbf{v})$ and $h(\mathbf{x}, \mathbf{u})$. The solution of the MinxEnt program is [11]

$$f(\mathbf{x}, \mathbf{v}^*) = \frac{h(\mathbf{x}, \mathbf{u}) \exp\{-S(\mathbf{x})\lambda\}}{\mathbb{E}_{\mathbf{u}}\left[\exp\{-S(\mathbf{X})\lambda\}\right]}, \tag{10}$$

where λ is a constant (temperature) obtained from the following equation

$$\frac{\mathbb{E}_{\mathbf{u}}\left[S(\mathbf{X}) \exp\{-S(\mathbf{X})\lambda\}\right]}{\mathbb{E}_{\mathbf{u}}\left[\exp\{-S(\mathbf{X})\lambda\}\right]} = \gamma, \tag{11}$$

and $\mathbf{X} \sim h(\mathbf{x}, \mathbf{u})$. For $\gamma = \gamma^*$, the optimal temperature is $\lambda^* = -\infty$ and the optimal density $f^*(\mathbf{x})$ is a Dirac delta function located at $\mathbf{x}^*$. Given a successful choice of γ and obtaining the corresponding value of λ we could, in principle approximate the optimal $\mathbf{x}^*$ by generating samples from (10). However, sampling from such distribution is not a trivial task. Thus, we shall approximate the distribution (10) as a product of marginal densities which will allow easy sampling similarly to the basic CE method. Note that if $h(\mathbf{x}, \mathbf{u})$ is a discrete (multidimensional) distribution with finite support, then so is $f(\mathbf{x}, \mathbf{v}^*)$ and, consequently, all its marginal distributions. Thus, all these distributions are completely determined by their parameters which may be calculated as follows. Assuming as before, that $\mathbf{X} = (X_1, ..., X_n)$ is a random vector such that each X_i is a discrete random variable that can assume a finite number of values $\{a_1, ..., a_m\}$, the PME estimator of

$$v_{jk} \triangleq \mathbb{P}\{X_j = a_k\} = \mathbb{E}_{\mathbf{v}}\left[\mathbb{I}_{\{X_j = a_k\}}\right]$$

is [14]

$$\hat{v}_{jk} = \frac{\sum_{i=1}^N \mathbb{I}_{\{X_{ij} = a_k\}} \exp\{-S(\mathbf{X}_i)\lambda\}}{\sum_{i=1}^N \exp\{-S(\mathbf{X}_i)\lambda\}}. \tag{12}$$

It is readily seen that (12) is essentially the same as (8) with indicators $\mathbb{I}_{\{S(\mathbf{X}_i) \geq \hat{\gamma}_t\}}$ being replaced by exponentials $\exp\{-S(\mathbf{X}_i)\lambda\}$. The optimal "temperature" parameter λ is obtained from the (numerical) solution of the stochastic version of (11) such that the single constraint in (**??**) is satisfied:

$$\frac{\sum_{i=1}^N S(\mathbf{X}_i) \exp\{-S(\mathbf{X}_i)\lambda\}}{\sum_{i=1}^N \exp\{-S(\mathbf{X}_i)\lambda\}} = \gamma. \tag{13}$$

Similarly to the CE method, we invoke a multi-stage procedure where a sequence of reference parameters $\{\mathbf{v}_t, t \geq 0\}$, a sequence of levels $\{\gamma_t, t \geq 1\}$ and a sequence of temperatures $\{\lambda_t, t \geq 1\}$ are generated. As before, we shall use the latest available information for the prior density. Namely, at stage t, $h(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}, \mathbf{v}_t)$. The whole optimization procedure is summarized in Algorithm 2. For both CE and PME optimization routines we expect the parameter vectors to converge to degenerate ones, such that by sampling from the final distribution we shall always obtain optimal or near-optimal solutions. Unlike CE, all samples are used to update the parameters in the PME method (although modifications are possible) and the solution for λ in (13) requires a line-search procedure and usually cannot be done analytically. We thus expect each iteration of the PME method to be slower than its CE counterpart. All "tuning" methods used for CE, such as smoothing and stopping rules, are directly applicable here as well.

Algorithm 2 The PME Algorithm for Optimization.

1: Define $\hat{\mathbf{v}}_0 = \mathbf{u}$. Set $t = 1$ (level counter).
2: Generate $\mathbf{X}_1, \ldots, \mathbf{X}_N$ from $f(\cdot; \hat{\mathbf{v}}_{t-1})$ and compute the performance mean from $\hat{\gamma}_t = \mathbb{E}_{\hat{\mathbf{v}}_{t-1}} S(\mathbf{X})$.
3: Use the same sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ and solve the stochastic program (13). Denote the solution by $\hat{\lambda}_t$.
4: Update $\hat{\mathbf{v}}_t$ componentwise via (12).
5: If stopping criteria are met - stop, otherwise set $t = t+1$ and reiterate from step 2.

4. CE BASED DATA ASSOCIATION

In this section we develop a family of CE-based algorithms to solve the multi-scan multi-target tracking problem. Since the only difference between the CE and PME methods for combinatorial optimization is the updating scheme of the parameters, we shall describe both methods together and explain the differences when needed. In order to apply the CE and PME methods to our problem we must specify the randomizing pdfs $\{f(\cdot;\mathbf{v})\}$ and the procedure for sampling the solutions from it. A convenient framework for both methods is to encode the optimization problem as a graph and to introduce the randomization on the graph's edges or nodes [13].

4.1 The Connectivity Graph

Let $n = \sum_{t=1}^{T} |Y_t| = |Y_{1:T}|$ be the total number of observations (both noisy detections and false alarms). We define $G = (V, E)$ to be the basic connectivity graph of the problem, where the set of nodes $V = Y_{1:T}$ is the set of all measurements as defined above and the graph edges are

$$\begin{aligned} E \quad = \quad & \{(y_{t_1}, y_{t_2}) \mid y_{t_1}, y_{t_2} \in Y,\ t_1 < t_2, \\ & \|y_{t_2} - y_{t_1}\| \le (t_2 - t_1) v_{\max},\ t_2 - t_1 \le d_{\max}\}. \end{aligned} \tag{14}$$

This graph connects every node (measurement) with any other node that can be an immediate successor in a feasible track, subject to speed and separation constraints. A feasible track which is a set of observations with increasing time tags is represented as a path in G, that is

$$\tau = \left\{ y^i, i = 1, 2, \ldots, j \mid (y^i, y^{i+1}) \in E,\ i = 1, \ldots, j-1 \right\}.$$

The nodes of the graph represent measurements (both noisy detections and false alarms), and the edges represent the possible event that their endpoints originate from the same target. Note that a node y that has no incoming and outgoing edges is a false alarm by default, and may be removed from the graph and permanently added to the set of false alarms τ_0. Henceforth, we shall assume that all such nodes are removed from the graph. From now on, we identify each valid partition of the measurements with the corresponding partition of the graph nodes. The goal is to find a partition of the the graph nodes into a set of vertex-disjoint paths $\{\tau_i, i \ge 1\}$ (which will represent tracks) and a set of isolated nodes τ_0 (which will represent false alarms) such that the posterior $\mathbb{P}\{\omega \mid Y_{1:T}\}$, defined in (4), is maximized. Let $S(\omega) = \mathbb{P}\{\omega \mid Y_{1:T}\}$ be the cost of a partition $\omega \in \Omega$.

We would like to incorporate the possibility of a (random) appearance of the new targets and termination of existing ones into the connectivity graph. This will be done by augmenting the basic connectivity graph by introducing new nodes and edges. In order to incorporate the possibility of termination of a target, we introduce an additional "sink" node t that represents termination of a track. Each other node i in the basic connectivity graph is connected to t by a directed edge represents the event that i corresponds to the last detection of the target prior to termination. Additionally, t has a self-loop which meaning will be made clear shortly. In order to be able to handle problems involving unknown number of targets and initiation of new ones we introduce additional n nodes into the graph – one for each original node of the basic graph. These will be labeled $s_1, s_2, ..., s_n$. Each new node s_i has two outgoing edges – one leading to the corresponding node $i \in V$ and another

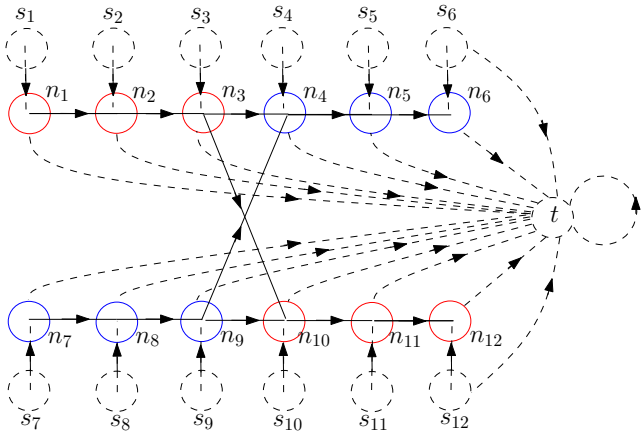


Figure 1: An example of the ACG. Solid circles represent the actual measurements. Node i is denoted as n_i. Dashed circles are the additional nodes.

to the termination node t. The event represented by the former edge is that node i is the first detection of a track. A path that represents a track with j measurements contains now $j+2$ nodes as follows – $\{s_{v_1}, v_1, ..., v_j, t\}$. We call the resulting structure an augmented connectivity graph (ACG) and refer to the original nodes in ACG, representing the actual measurements, as inner nodes. An example of an ACG is shown in Fig. 1.

4.2 Probability Distribution Using the ACG

Next we define the probability distribution on the set of partitions using the augmented connectivity graph. Let $G_A = (V_A, E_A)$ be the ACG of the problem. For each inner node i of the ACG, let $p_b(i)$ denote the probability that i is an initial node. Similarly, for each inner node i in the ACG, let $p_t(i)$ denote the probability that i is a final node in a path. Let P_A be the adjacency matrix of G_A as follows,

$$\begin{aligned} P_A = \{p_{ij} > 0 \quad : \quad & i, j \in V_A,\ \sum_{j \in V_A} p_{ij} = 1 \\ p_{ij} \quad = \quad & 0 \text{ if } (i,j) \notin E_A, p_{tt} = 1, \\ p_{ij} \quad = \quad & p_t(i) \text{ if } j = t,\ p_{ij} = p_b(j) \text{ if } i = s_j\}. \end{aligned} \tag{15}$$

From the construction of G_A, all rows of P_A have at least one non-zero entry and it may be interpreted as a one-step transition matrix of a directed Random Walk $\mathcal{M}$ on the augmented graph G_A. We define the following probability distribution on the edges of G_A:

$$f(\omega \mid P_A) = \begin{cases} \frac{1}{Z} \prod_{k=1}^{n} \prod_{r=1}^{|\tau_k|+1} \prod_{i,j:\{\omega \in \Omega_{ij}(r)\}} p_{ij}, & \omega \in \Omega \\ 0, & \omega \notin \Omega \end{cases} \tag{16}$$

where Z is a normalization constant and $\Omega_{ij}(r)$ is the set of all partitions in Ω for which there is a path with rth transition from node i to node j

4.3 Sampling of Candidate Solutions

We now describe how to sample a partition ω from the distribution (16) defined through the edges of the augmented connectivity graph G_A having adjacency matrix P_A. Recall that P_A may be interpreted as a one-step transition matrix

of the corresponding Random Walk $\mathcal{M}$. We describe the sampling procedure in two stages. First, we describe sampling a single path in G_A. We then proceed with sampling multiple non-intersecting paths.

4.3.1 Sampling a Single Path

Sampling a single random path from the graph G_A is performed by picking uniformly at random i_1 from $\{1, ..., n\}$ and generating a random walk on G_A starting at s_{i_1} according to the transition matrix P_A until hitting t. Note that since G_A is a directed acyclic graph – after at most T steps the sink node t will be reached and the process will terminate. The resulting path is $\tau = \ s_{i_1}, v_{i_1}, ..., v_{i_j}, t, j \geq 0\ $. It corresponds to a track associated with the measurements corresponding to the nodes $\ v_{i_1}, ..., v_{i_j}\ $. From our problem definition a valid track contains 2 or more measurements. Thus, the resulting path is interpreted as a track if it contains 4 or more nodes, namely if $j \geq 2$. Shorter paths are interpreted differently. For example, the only interpretation of $s_1 \rightarrow v_1 \rightarrow t$ is that v_1 is a FA and the only interpretation of the path $s_1 \rightarrow t$ is that v_{s_1} cannot be a first measurement in any track (it can, however, be an inner measurement of some other track).

4.3.2 Sampling Non-intersecting Paths

To sample from (16) we could generate independently n random walks on G_A according to the procedure described above. The resulting set of paths could have been returned if it represented a valid solution (i.e. different paths do not share nodes), or rejected otherwise. However, most samples generated in this way will not be valid. We thus consider an alternative, efficient sampling which ensures that only valid solutions will be sampled. To this end, we invoke the elimination principle similar to that used in the Traveling Salesperson Problem in [13]. Strictly speaking, we repeat the procedure for single path sampling n times from the n auxiliary nodes $s_1, s_2, ..., s_n$ as before, but, upon reaching a node we mark it as used and eliminate all incoming edges, that is nullify the probabilities on these edges and re-normalize the transition probabilities of remaining edges. This sampling procedure is summarized in Alg. 3.

Algorithm 3 Sampling Procedure for the CEDA/PMEDA.

1: Pick s_{i_1} u.a.r from $S = \{s_1, ..., s_n\}$.
2: Define $P^{(1)} = P_A$ and $t_{1_1} = s_{i_1}$ to be the first node of the first path. Set $u_t = t_{1_1}$ to be an auxiliary variable that holds the current node. Let $k = 1, j = 1$.
3: Obtain $P^{(j+1)}$ from $P^{(j)}$ by setting the u_t-th column of $P^{(j)}$ to 0 and normalizing the rows to sum up to 1.
4: Generate $t_{k_{j+1}}$ from the distribution formed by the u_t-th row of $P^{(j)}$. If $t_{k_{j+1}} = t$ then $S = S \backslash \{s_{i_1}\}$ and go to 4. Otherwise set $u_t = t_{k_{j+1}}$, $j = j + 1$ and reiterate from 3.
5: If $S = \emptyset$ then stop. Otherwise, set $j = 1$, $k = k + 1$. Pick s_{i_1} u.a.r from S and set $t_{k_1} = s_{i_1}$ (that is, start new path), $u_t = t_{k_1}$ and repeat from step 3.

4.4 Parameters Update

In order to update the distribution, we need to estimate the parameters of the new matrix P_A (15) based on the best samples that have been obtained. This is performed by taking the elite samples and calculating the Maximum Likelihood estimate of the elements of the new matrix P_A. As is well known [13], each parameter $p_{ij} = (P_A)_{ij}$ is estimated as $\hat{p}_{ij} = \frac{N_{ij}}{N_i}$ where N_{ij} is the number of times the edge (i, j) was used in the elite sample and N_i is the number of times node i was visited in that sample. The PME updating is performed in a similar manner as explained in section 3. In addition, in our experiments we have used the smoothed update as described in section 3.

4.5 Multi-Node State Representation

Recall that the state variable x_t contains, in addition to the target position, additional information such as target velocity which is useful for estimating its next position. The measurement variable z_t, however, is assumed to carry only position information. Since each inner node of the ACG represents a single measurement, it is insufficient for predicting likely positions for the next measurement. One possible solution that we employ here is to modify the parametric distribution that underlies the CE and PME algorithms in the following way. Instead of sampling the next node in each path based on the current node alone, we allow the sampling probability to depend on several recent nodes (2 nodes are used in the examples of the next section). While the number of parameters to be estimated is increased, the performance improvement is dramatic. This idea was introduced in [12] and may easily be extended to higher order motion models.

4.6 Initialization Scheme

We introduce an initial probability distribution by applying a Kalman Filter to every three (or more – depending on the target model) neighboring inner nodes of G_A which represent three consecutive measurements. The filter is initialized with the first two measurements using the two-point differencing technique [3]. Namely, we initialize the position as that of the first measurement and the velocity as the difference between the two measurements divided by the difference in their time tags. The corresponding probability of each edge of the graph is computed on the basis of the predicted value of the Kalman Filter compared with the third measurement.

Recall next that each node representing a measurement is linked to the sink node t in the ACG. The probabilities on these edges are initialized to some small value, the termination probability p_z, which was introduced in section 2. The probabilities on the edges outgoing from each node are then re-normalized to sum to 1. Finally, initial probabilities are introduced on the edges connecting the s nodes with the rest of the graph. Each such edge is assigned a small probability, say $p_b = 0.3$ and the complementary value is assigned to the edge connecting s to t.

4.7 Computational Requirements

The total running time is determined by the total number of iterations until convergence, each of which is determined by the number of candidate solutions, the time required to generate each solution, the time required to evaluate each solution, and the time to update the parameters for the following CE or PME iteration. Recall that a candidate solution is drawn by randomly choosing vertex-disjoint paths from the connectivity graph which is the problem was encoded to. Since the number of targets and their positions are not known a-priori, every node in the graph has a prob-

ability to represent a true detection. Thus, the procedure of sampling a single path in the graph is repeated, in the worst case, $\mathcal{O}(n)$ times where n is the total number of observations. Sampling a single path from the graph is linearly dependent on the average outdegree of the nodes in the graph, which is of the order of $\mathcal{O}(n)$, and on the surveillance duration T. Namely, the complexity of sampling a single candidate solution is $\mathcal{O}(T \cdot n^2)$. Evaluation of the cost of each solution requires, in the worst case, $\mathcal{O}(nT^2)$ since we apply a Kalman filter to at most n/T tracks of length T. The procedure of sampling a solution and evaluating its cost is repeated N times (as the number of sampled solutions in each iteration). The updating procedure in the multi-node state representation requires $\mathcal{O}(N \cdot n^3)$ and the preprocessing stage – $\mathcal{O}(n^3)$ since we perform a constant number of operation on every measurements triplet. The overall procedure time requirements are summarized as

$$\mathcal{O}(n_{CE}(N \cdot n^3 + N \cdot T \cdot n^2)), \tag{17}$$

where n_{CE} is the number of CE/PME iterations till convergence.

5. SIMULATION RESULTS

5.1 General Simulation Setup

We consider the same simulation setup as in [9]. A rectangular region on a plane, $\mathcal{R} = [0, 1000] \times [0, 1000] \subset \mathbb{R}^2$ is taken to be the surveillance region. Each target has a 4-component state vector with position and velocity in the x and y directions. Namely,

$$x_t = [p_{xt}, v_{xt}, p_{yt}, v_{yt}]^T.$$

We have used the Discrete White Noise Acceleration (DWNA) model [3] for the targets dynamics. Namely,

$$x_{t+1} = Ax_t + Gw_t,$$

where

$$A = \text{diag}\,[F_1, F_1],\ F_1 = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}.$$

In addition, the vector process noise is $w_t = [w_{xt}\ w_{yt}]^T$ with covariance $Q = \text{diag}(\sigma_w^2, \sigma_w^2)$, and

$$\text{cov}(Gw_t) = \sigma_w^2 \cdot \text{diag}(Q_1, Q_1),\ Q_1 = \begin{bmatrix} \frac{1}{4}T_s^4 & \frac{1}{2}T_s^3 \\ \frac{1}{2}T_s^3 & T_s^2 \end{bmatrix}.$$

The measurement equation is

$$y_t = Cx_t + v_t,$$

where

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

and the (vector) measurement noise covariance matrix is $R = \text{diag}(\sigma_v^2, \sigma_v^2)$. We have used a surveillance duration of $T = 10$ scans. Targets appear at a uniformly chosen position from the left bottom or right bottom quadrants of $\mathcal{R}$ respectively. They all move diagonally (in straight lines) with constant velocity randomly chosen between $0.2v_{\max}$ and $0.9v_{\max}$. Each target's appearance and disappearance times are chosen uniformly from the first and last quarters of the surveillance interval respectively. An example of a typical scenario is shown in Fig. 2.

5.2 Performance Measures

To the best of our knowledge, definition of unified performance measures for evaluation of MTT algorithms is an open question in the tracking and information fusion community. Performance evaluation of single-target tracking algorithms (both with and without measurement origin uncertainty) may be quantified by means of the Mean Square Error (MSE). Such criteria are problematic when dealing with association algorithms, since the MSE may provide meaningful insight on the performance only provided the data association is perfect. We thus adopt the following, rather intuitive, measures for performance evaluation suggested in [9].

1. The normalized correct associations (NCA), that is, the number of correct associations made by CEDA algorithm divided by the true number of associations.

2. The incorrect-to-correct association ratio (ICAR) which measures the ratio of incorrect to correct associations.

Mathematically, for each partition $\omega \in \Omega$, the set of all associations in ω is represented as $\text{SA}(\omega) = \{(\tau, t_i^\tau, t_{i+1}^\tau) : i = 1, \ldots, |\tau| - 1, \tau \in \omega\}$ where t_i^τ is the time at which the track τ is observed i times. The set of correct associations in ω relative to ω^*, which is the true partition, is $\text{CA}(\omega) = \{(\tau, t, s) \in \text{SA}(\omega) : \tau(t) = \tau^*(t), \tau(s) = \tau^*(s), \tau^* \in \omega^*\}$ The above measures now read:

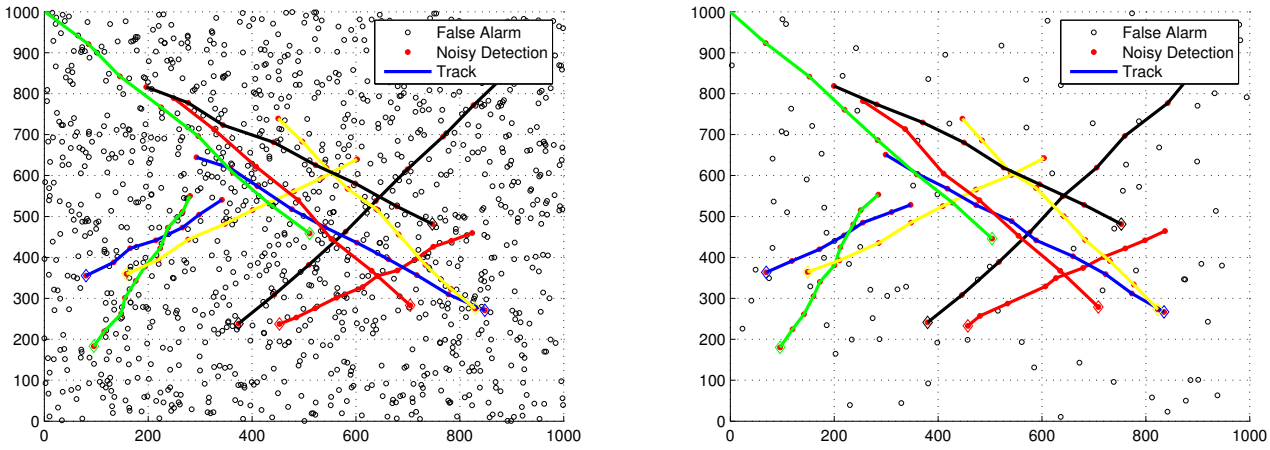
$$\text{NCA}(\omega) = \frac{|\text{CA}(\omega)|}{|\text{SA}(\omega^*)|}, \quad \text{ICAR}(\omega) = \frac{|\text{SA}(\omega)| - |\text{CA}(\omega)|}{|\text{CA}(\omega)|}.$$

In addition, we record the number of tracks estimated by the algorithms. From the definition, NCA varies between 0 and 1 and it provides a measure of the number of correct associations. In case of perfect associations - NCA = 1. It does not account, however, for false tracks. On the other hand, ICAR is a positive measure not bounded from above and, informally, it counts false associations – both false tracks and false continuations of true tracks. When no incorrect associations are made - ICAR = 0.

5.3 Experiments and Results

In this section we test the performance of the proposed algorithms by comparing them with the performance of the Multiple Hypotheses Tracker (MHT), the Greedy tracker [9] and the Markov Chain Monte Carlo Data Association (MCMCDA) as was reported in [9]. The MHT and MCMCDA methods were briefly described in section 1. The greedy tracker proposed in [9] is a batch-mode nearest neighbor multiple-target tracking algorithm. It generates candidate tracks by picking measurements nearest to the predicted states until there there are no unused measurements left. The algorithms are compared to each other using the performance measures NCA, ICAR and the estimated number of targets described in subsection 5.2.

Several ways exist to challenge a data association algorithm. The first is by intensifying the false alarm rate $\lambda_f V$. In addition, one can decrease the detection probability P_d and increase the density of tracks K. Closely moving targets, low detection probabilities and high false alarm rate make the problem more difficult. In the following we perform three different test sequences in which we modify different parameters to evaluate the performance of the algorithms – the false alarm rate λ_f, the density of tracks K,



(a) All Scans

(b) One Scan

Figure 2: An example of a typical scenario. Target detections and FAs from all 10 scans are shown (a), FAs from a single scan are shown (b). The FA rate is $\lambda V = 100$.

and the detection probability P_d. In all simulations presented below the results are averaged over 8 repeated runs.

5.3.1 *Number of Targets*

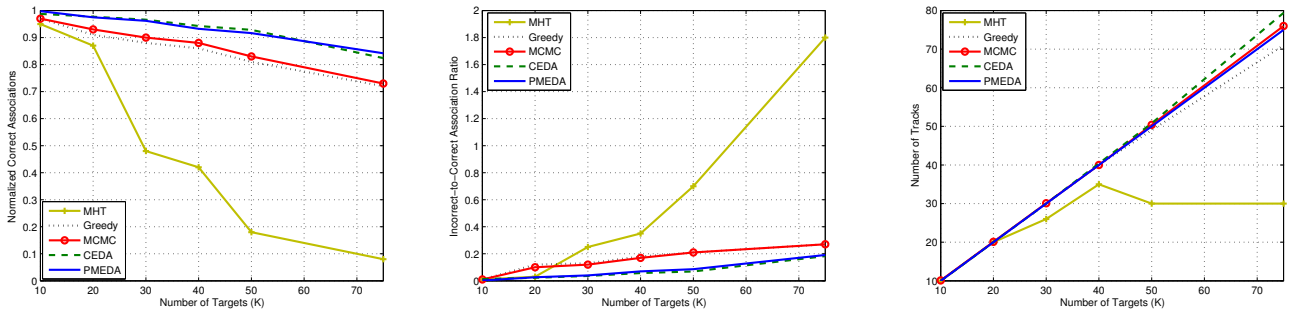
In this experiment we modify the track density. The number of targets K in the scenario is varied between 10 and 75 and keep the clutter rate low and the detection probability high. All other parameters are fixed as well, namely – $\lambda_f V = 1$, $P_d = 0.999$ and $v_{\max} = 140$ unit length per unit time, $p_z = 10^{-2}$ and $\lambda_b V = 1$ such that, on average, there is a single new target at each scan. The average NCAs, ICARs and number of tracks are presented in Fig. 3. It is readily seen that both cross-entropy based algorithms score higher than Greedy and MCMCDA in NCA and better in ICAR introducing no significant difference between CEDA and PMEDA. All algorithms outperform the MHT. Note that MCMCDA improves only a little the performance achieved by the Greedy algorithm. This is because the Greedy algorithm finds a solution, which is a strong local maximum of the problem, and MCMC, being a local search method, needs sufficiently many steps to escape from this strong local extremum. Although in theory the MCMC method is considered a global optimization method, it turns out in this experiment that at this level of problem complexity it quickly got trapped in local minima, from which it could not get out in any reasonable computation time. Thus, MCMC effectively reduces here to local search, while CE based algorithms maintain a more global flavor as witnessed by their performance.

5.3.2 *False Alarms Rate*

There is a constant number of $K = 10$ tracks, which move at constant velocity as described above. The clutter rate varies between $\lambda_f V = 1$ to $\lambda_f V = 100$. Namely, in this experiment we challenge our algorithms in heavily cluttered environment, keeping the detection probability high and the number of targets low. The results are depicted in Fig. 4. Clear superiority of PMEDA may easily be noticed with nearly 90% of correct associations. The CEDA algorithm behaves similarly at low and moderate clutter rates, but degrades in NCA performance at high clutter rates. However, false tracks are not produced keeping the ICAR relatively low. We may conclude that, in this application, the updating rule of the PMEDA algorithm, which uses all the samples obtained at a given iteration rather than the elite samples used by CEDA, is preferable. As reported in [9], the MHT algorithm does not make any associations when $\lambda_f V \geq 80$ resulting in zero NCA and unreported ICAR. MCMCDA being initialized with the output of the greedy algorithm, removes many of the false tracks found at the initialization stage, thus improving the ICAR, but at the same time it degrades the NCA performance by changing some of the correct associations. As a result, the obtained NCA is lower than that of the greedy algorithm by a few percent. Again, the inability of the MCMCDA to improve sufficiently the greedy solution is due to its local behavior, which limits the possibility to escape from the strong local extremum obtained by the greedy algorithm. The greedy algorithm achieves reasonable performance in terms of the NCA with more than 80% of correct associations, but results in unacceptably large ICAR due to generation of high number of false tracks which also increase the ICAR.

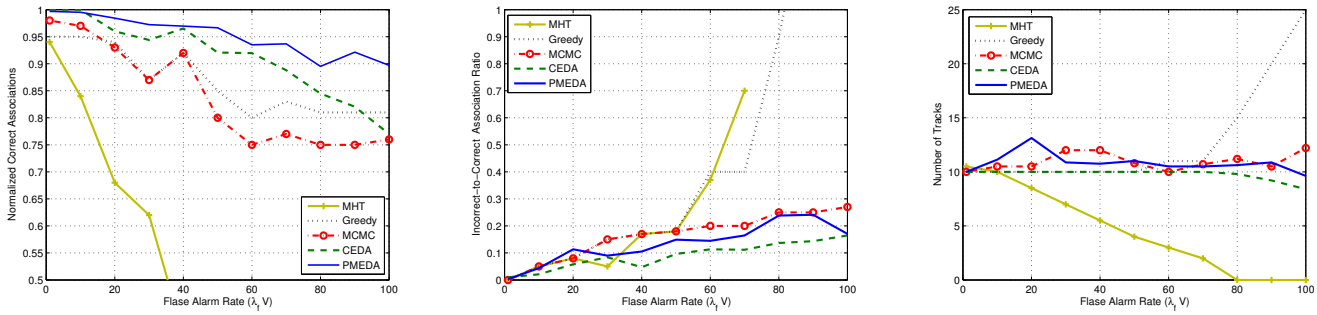
5.3.3 *Detection Probability*

There is a constant number of $K = 10$ tracks, which move at constant velocity each uniformly chosen between 30 and 120 unit lengths per unit time. As mentioned, each target's appearance and disappearance times are chosen uniformly from the first and last quarters of the surveillance interval respectively. The clutter rate is kept constant at $\lambda_f V = 1$. The probability of detection varies between $P_d = 0.3$ to $P_d = 0.9$. Due to lower detection probabilities we have set $d_{\max} = 5$ since many consecutive missed detections may occur. The probability of track termination is $p_z = 0.01$ and the appearance of new tracks is modeled by $\lambda_b V = 1$. Now the targets are not observed all the time. The results are depicted in Fig. 5. Both cross entropy based algorithms outperform all other algorithms with some superiority of



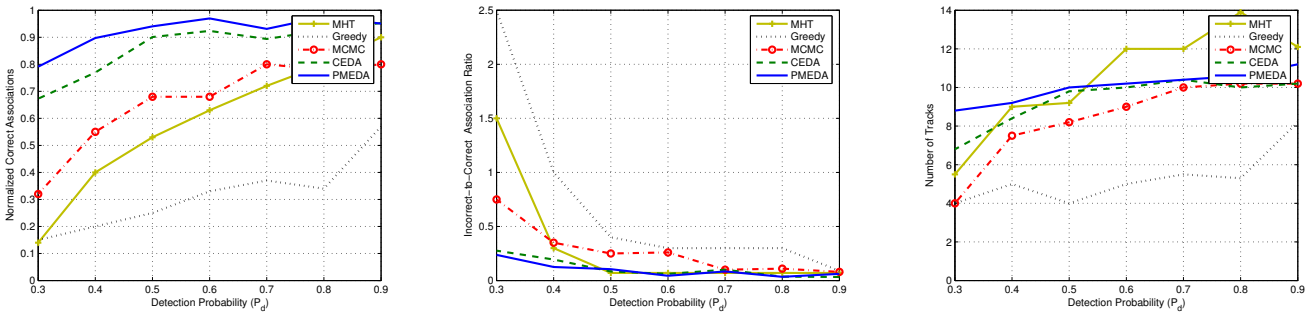
(a) NCA (b) ICAR (c) Number of Tracks

Figure 3: Simulation results for various values of K.



(a) NCA (b) ICAR (c) Number of Tracks

Figure 4: Simulation results for various values of $\lambda_f V$.



(a) NCA (b) ICAR (c) Number of Tracks

Figure 5: Simulation results for various values of P_d.

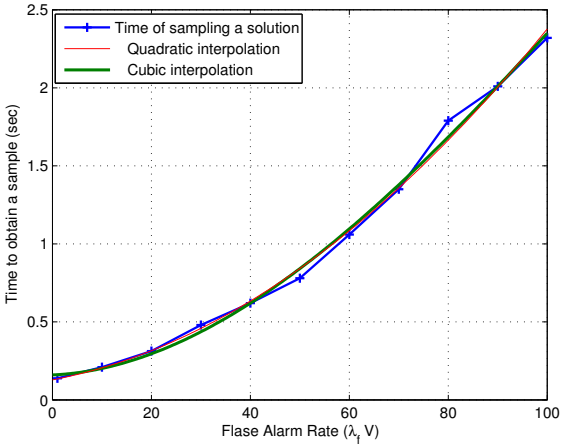


Figure 6: Time required to obtain a single sample.

the PMEDA over CEDA. Unlike in the previous tests, the greedy algorithm performs poorly with less than 50% of correct associations. MCMCDA scores better than MHT, but much worse than cross entropy based algorithms especially at very low detection probabilities. As mentioned before, although in theory MHT is an optimal solution in the MAP sense it performs poorly when the detection probability is low or the false alarm rate is high due to the necessary heuristics used in the MHT. These heuristics are required part of all practical implementations of the MHT since without them the number of hypotheses grows exponentially fast. MHT with such heuristics may work well when a few hypotheses carry most of the weight. However, when the detection probability is low or the false alarm rate is high, there are many hypotheses with low weight and there is set of dominating hypotheses, so MHT cannot perform well. This explains the above poor behavior of the algorithm when tested in the extreme cases.

5.3.4 *Computation Times*

All algorithms proposed in this paper were implemented in Matlab without any code optimizations and ran on a PC with 2.8GHz Intel processor. Recall that the overall performance is determined by the time required to obtain a single sample in the CE/PME procedures which requires $\mathcal{O}(T \cdot n^2)$ calculations. We present in Fig. 6 this empirically found time versus the clutter rate which is proportional to the average number of observations in the problem at hand. It is readily seen that this empirical evidence strongly supports the calculated complexity needed to obtain a single sample.

6. CONCLUSIONS

We have proposed two methods for the multi-scan multitarget data association problem based on the CE and PME heuristics. These schemes have been tested in simulation and show improved performance relative to the state-of-the-art algorithms. A major issue in the proposed algorithms is their computation time. Although polynomial in the problem parameters, the computation time is still considerable in challenging scenarios which involve a large number of measurements. In these cases the proposed algorithms are more suitable for off-line data analysis. Future research directions involve further reduction of the computation time and extension to multisensor cases. In addition, modification of the algorithms to handle long time intervals by applying a sliding window to the measurement set is of interest with preliminary results available in [15].

7. REFERENCES

[1] Y. Bar-Shalom and T. E. Fortmann. Tracking and Data Association. Academic Press, San Diego, 1988.

[2] Y. Bar-Shalom and X. Li. Multitarget-Multisensor Tracking: Principles and Techniques. 1995.

[3] Y. Bar-Shalom, X. Li, and T. Kirubarajan. Estimation with Applications to Tracking and Navigation. New York : Wiley, 2001.

[4] J. Collins and J. Uhlmann. Efficient gating in data association with multivariate distributed states. IEEE Trans. on Aerospace and Electronic Systems, 28(3):909–916, July 1992.

[5] S. Deb, M. Yeddanapudi, K. Pattipati, and Y. Bar-Shalom. A generalized s-d assignment algorithm for multisensor-multitarget state estimation. IEEE Trans. on Aerospace and Electronic Systems, 33(2):523–538, April 1997.

[6] M. Garey and D. Johnson. Computers and Intractability - A Guide to the Theory of NP-Completeness. Bell Telephone Laboratories, 1979.

[7] C. L. Morfield. Application of 0-1 integer programming to multitarget tracking problems. IEEE Trans. on Automatic Control, AC-22(3):302–312, June 1977.

[8] W. Ng, J. Li, S. Godsill, and J. Vermaak. A review of recent result in multiple target tracking. In Proc. of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.

[9] S. Oh, S. Russel, and S. Sastry. Markov chain monte carlo data association for general multiple-target tracking problems. In Proceedings of the IEEE Conf. on Decision and Control, 2004.

[10] D. B. Reid. An algorithm for tracking multiple targets. IEEE Trans. on Automatic Control, AC-24(6):843–854, December 1979.

[11] R. Rubinstein. A stochastic minimum cross-entropy method for combinatorial optimization and rare-event estimation. Methodology and Computing in Applied Probability, (1):1–46, 2005.

[12] R. Rubinstein. Semi-iterative minimum cross-entropy algorithms for rare-events, counting, combinatorial and integer programming. To appear in Methodology and Computing in Applied Probability, November 2007.

[13] R. Rubinstein and D. Kroese. The Cross-Entropy Method - A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning. Springer Science, 2004.

[14] R. Rubinstein and D. Kroese. Simulation and the Monte Carlo Method. John Wiley & Sons, Inc., 2007.

[15] D. Sigalov. Data association in multi target tracking using cross entropy based algorithms. Master's thesis, Technion–Israel Institute of Technology, 2008.

[16] J. Spall. Introduction to Stochastic Search and Optimization. John Wiley & Sons, Inc., 2003.