Multitarget Data Association with Higher-Order Motion Models

Robert T. Collins*
The Pennsylvania State University
University Park, PA 16802, USA

Abstract

We present an iterative approximate solution to the multidimensional assignment problem under general cost functions. The method maintains a feasible solution at every step, and is guaranteed to converge. It is similar to the iterated conditional modes (ICM) algorithm, but applied at each step to a block of variables representing correspondences between two adjacent frames, with the optimal conditional mode being calculated exactly as the solution to a two-frame linear assignment problem. Experiments with ground-truthed trajectory data show that the method outperforms both network-flow data association and greedy recursive filtering using a constant velocity motion model.

1. Introduction

The multi-target, multi-frame data association problem has a long history, with early works appearing in the target tracking [6] and computer vision [10] communities. It has seen a resurgence of interest in computer vision due to recent popularity of tracking-by-detection approaches, which apply a detector independently on every frame to find candidate objects that are then associated across frames [1].

Traditional data association problems consider point-like objects (e.g. radar blips), with trajectory quality measured by smoothness and continuity of object motion. Vision-based data association, on the other hand, involves objects of extended spatial extent in an image, from which discriminative appearance cues can be extracted to help disambiguate matches. We argue that recent vision-based approaches have begun to rely too heavily on these appearance cues, to the point of ignoring motion characteristics. One example is the recent network flow approach to data association, which formulates an objective function containing only pairwise costs, and for which a globally optimal solution can be found in polynomial time [22, 3, 17]. And yet, see Figure 1 for a simple example where network flow is unable to find the correct trajectories because it is unable to represent constant velocity motion constraints. These network flow formulations gain efficiency by limiting the cost functions they can handle. Our experiments show that lack of regularizing motion models has a detrimental effect on quality of the trajectories found, particularly when appearance constraints are weak and detection frame-rate is low.

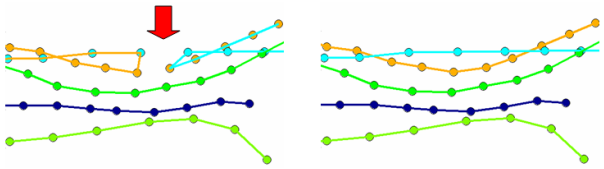


Figure 1. Comparison of trajectories computed by network-flow data association (left) versus our approach using a cost function that incorporates a constant-velocity smoothness term (right). Network flow approaches gain their efficiency by limiting the cost function to pairwise terms. Using higher-order motion models leads to smoother trajectories that reduce the number of mismatch errors, particularly at low sampling rates.

To focus our arguments, in this paper we do not use appearance terms – objects are described solely by their 2D point locations. We do not deny that when objects are easily distinguishable by appearance, appearance terms can (and probably should) do most of the work. Our point is that we need to retain the ability to leverage kinematic motion models in cases where objects are very similar, or when there are rapid appearance changes due to pose or lighting. Although this paper focuses on kinematic cost functions alone, we fully expect an improvement in data association performance when appearance terms are added back in.

From a combinatorial optimization standpoint, search for the best data association is governed by the form of the objective function, which requires two design decisions: how to represent the set of all trajectories that can be formed from raw target observations; and how to calculate the cost (or affinity score, if maximizing) of each trajectory. Our approach borrows from network flow ideas to represent each trajectory as a sequence of edges through a trellis graph, enabling efficient local update rules. However, network flow approaches also factor the cost function into pairwise costs,

*This work was funded by a grant from ObjectVideo and the AFOSR. Technical discussions with David Tolliver and Khurram Shafique of ObjectVideo were instrumental to the development of this work.

one cost per edge in the graph, while we retain general cost functions that are in principle defined over entire trajectories. This is a major difference, and has both pros and cons. On the upside is the increase in power of our cost functions to represent nontrivial motion constraints. On the downside, the use of cost functions defined over temporal windows longer than 2 frames yields data association problems that are NP-hard. However, this NP-hardness is nothing new to the multi-target tracking community, which has shown that approximate algorithms can produce multi-frame trajectories of high quality while working efficiently in practice.

Contributions.

• We present an iterative approximate solution to the multidimensional assignment problem under general cost functions. The method maintains a feasible solution at every step, and is guaranteed to converge to a (local) minimum. It is similar to the iterated conditional modes (ICM) algorithm, but is applied at each step to a block of variables representing possible correspondences between two adjacent frames. The block-optimal conditional mode at each step is calculated exactly and efficiently as the solution to a two-frame linear assignment problem.

• Our approach differs from both traditional multi-frame data assignment approaches as well as from more recent network flow approaches. Unlike traditional multi-frame data association, we factor the decision variable for each trajectory into a product of variables defined over edges in a trellis graph. Unlike recent network flow formulations, we retain the full power of general cost functions for describing kinematic motion models and long-range regularizers that improve the quality of estimated trajectories.

• We develop a novel higher-order cost function for data association that uses active contour ("snake") spline energy to measure the quality of a proposed trajectory. We show in our evaluation that this cost function compares favorably with network-flow solutions and greedy sequential filtering using a constant velocity motion model.

2. Background and Related Work

This section reviews different combinatorial formulations of the data association problem. Data association is the process of partitioning a set of observations into trajectories. Although the combinatorial formulations differ, all are based on the fundamental constraint that trajectories must be disjoint; that is, no two trajectories can claim the same observation. To keep the discussion at a high level, observations will be denoted as elements, and trajectories as subsets of elements.

In the weighted **Set Partition Problem (SPP)** [2], we are given a universe of elements $U = \{e_1, e_2, \ldots, e_n\}$, a list of allowable sets $S = \{S_1, S_2, \ldots, S_m | S_j \subseteq U\}$, and a cost c_j for each set. The goal is to choose a minimum cost collection of sets that form a partition of U, *i.e.* each element appears in one and only one set. Creating a solution vector of binary decision variables, $x = [x_1, x_2, \ldots, x_m]^T$, with $x_j = 1$ iff set S_j is in the solution, SPP can be written as the binary integer program

$$\min \sum_{j=1}^{m} c_j x_j \;\text{ s.t. } \begin{cases} Ax = 1 \\ x_j \in \{0,1\} \end{cases} \tag{1}$$

where A is an $n \times m$ constraint matrix with one row for each element, one column for each set, and a_{ij}=1 if element i appears in set j, 0 otherwise. The i-th constraint $\sum_j a_{ij} x_j = 1$ counts the number of sets in the solution that contain element i, and requires there to be only one.

The closely-related **Set Packing (SP)** problem is written as a maximization

$$\max \sum_{j=1}^{m} p_j x_j \;\text{ s.t. } \begin{cases} Ax \leq 1 \\ x_j \in \{0,1\} \end{cases} \tag{2}$$

where p is now a vector of scores to be maximized rather than costs to be minimized (for example, p_j=$-c_j$). The main difference is the $\leq$ relation in the constraints, making it possible to leave some elements unused in the solution. In the context of data association, this makes it easier to ignore false positive detections and spurious trajectory fragments, rather than explicitly accounting for them. The classic paper by Morefield on 0-1 programming for data association is an SP formulation [14]. Recent network flow algorithms for data association [22, 3, 17] can also be interpreted as solving SP, although limited to scores/costs having a decomposable structure (Section 3.3).

The **Maximum-Weight Independent Set (MWIS)** problem, aka vertex packing, is also closely related to SP and SPP. Consider the column intersection graph G(S,E) associated with constraint matrix A, with one vertex for each set S_j, and edge set $E = \{(i,j) | S_i \cap S_j \neq \emptyset\}$, that is, having an edge between two vertices if their underlying sets are not disjoint. As above, each set S_j has a score p_j and a binary decision variable x_j indicating whether set/vertex S_j is part of the solution. The MWIS problem solves for

$$\max \sum_{j=1}^{m} p_j x_j \;\text{ s.t. } \begin{cases} x_j + x_j \leq 1, \;\; \forall (i,j) \in E \\ x_j \in \{0,1\}\,. \end{cases} \tag{3}$$

The pros and cons of using MWIS for data association as compared to multi-hypothesis tracking (MHT) are discussed in [16]. A two-frame MWIS approach followed by hierarchical linking is presented in [7].

The **MultiDimensional Assignment (MDA)** problem is a specialization of SPP to the case of hypergraphs where elements are organized into a k-partite graph and the allowable sets are hyperedges containing one element per partite set. The SPP formulation is particularly well-suited to

data association for multi-target tracking, where elements are target observations in a series of frames, and allowable sets are paths connecting observations between frames.

For example, assume a sequence of k frames with n observations in each frame, from which an optimal partition of n trajectories of length k is to be formed (this is generalized in Section 3 to include varying numbers of observations and trajectory lengths). The k-partite structure of MDA enables enumeration over trajectories and cost function values by nested sums over observations and frames, *i.e.*

$$\min \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \cdots \sum_{i_k=1}^{n} c_{i_1 i_2 \ldots i_k} x_{i_1 i_2 \ldots i_k} \tag{4}$$

subject to

$$\sum \sum_{I \backslash i_f} \cdots \sum x_{i_1 i_2 \ldots i_k} = 1; \quad \begin{cases} f = 1, 2, \ldots, k \\ i_1 = 1, 2, \ldots, n \\ i_2 = 1, 2, \ldots, n \\ \vdots \\ i_k = 1, 2, \ldots, n \\ x_{i_1 i_2 \ldots i_k} \in \{0, 1\} \end{cases} \tag{5}$$

Here, $I \backslash i_f$ denotes all observations in all frames other than frame f. As in SPP, there is one constraint per observation, represented as a summation over all k-paths containing it.

MDA is known to be NP-hard for 3 or more frames. However, in the case of two frames it reduces to the 2D linear assignment problem on a bipartite graph

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \text{ s.t. } \begin{cases} \sum_j x_{ij} = 1; & i = 1, 2, \ldots, n \\ \sum_i x_{ij} = 1; & j = 1, 2, \ldots, n \\ x_{ij} \in \{0, 1\} \end{cases} \tag{6}$$

for which polynomial time exact solutions are known, *e.g.*, the Kuhn-Munkres (Hungarian) algorithm [8].

Data association problems formulated in an MDA framework have been widely studied, particularly target tracking applications performing probabilistic MAP estimation [6, 20, 18]. Solution methods include both deterministic (*e.g.* MHT [16]) and stochastic (*e.g.* MCMCDA [15]) search. Shafique presents a polynomial-time exact solution for pairwise cost functions and a greedy approximate solution for higher-order motion models [21]. We adopt a generalized MDA framework in Section 3.

Several recent vision papers show that data association can be formulated as a **Network Flow (NF)** problem for cost functions that decompose into a product of pairwise terms [22, 3, 17, 11]. Network flow can be solved in polynomial time using either linear programming [11], push-relabel methods [22], or successive shortest path algorithms [3, 17]. Efficient approximate solutions based on dynamic programming also can be applied.

However, as mentioned in the introduction, the network flow formulation is limited to relatively uninteresting motion models such as pairwise distance [22] or bounded velocity [17]. This is so because a static set of pairwise interframe edge costs is not able to represent kinematic constraints such as constant velocity that require data over three or more frames. We show in our experimental results (Section 4.2) that the inability to use higher-order motion models has a major detrimental effect on the quality of trajectories found using network flow.

All the previous approaches have been linear programming formulations. In contrast, data association can also be formulated as a **Quadratic Boolean Optimization (QBO)** problem [7, 13], including the problem of two frame association via graph matching [9]. The main benefit of quadratic programming is the ability to represent constraints between pairs of trajectories, such as coupling them to encourage similar motions [7].

3. Our Approach

We adopt a generalized multidimensional assignment (MDA) formalism that handles trajectory fragments and unassigned observations [18, 20], and present an iterative approximate solution for general cost functions. While retaining generality of the cost function, we factor the trajectory decision variables into pairwise edges between observations in adjacent frames, facilitating local trajectory updates. We present an iterative approach that cycles through the sequence, solving for decision variables between each subsequent pair of frames while holding the rest of the current solution fixed. These two-frame solutions are computed exactly and efficiently using the Hungarian algorithm. Each step of each iteration improves the value of the objective function (more precisely, does not increase it's value) and iterations continue until no further improvement is seen.

Our method is similar to the Iterated Conditional Modes (ICM) algorithm of Besag [5], except blocks of variables representing potential matches between pairs of adjacent frames are updated simultaneously, instead of a single variable at a time. This block update strategy is helpful because it maintains disjointness of trajectories. It is also expected that solution updates optimizing over multiple variables should be able to find stronger local optima than update schedules that solve for a single variable at a time.

In the next section we lay out the generalized MDA problem formulation. Section 3.2 presents our block-ICM iterative improvement strategy and sketches a proof (due to Besag) that it is guaranteed to converge. Section 3.3 shows that when the cost function can be factored into a product of pairwise costs between adjacent frames, our formalism reduces to the network flow approach. Finally, Section 3.4 discusses implementation issues related to initialization and termination of the algorithm.

3.1. Problem Formulation

Consider a sequence of k image frames where detections have been observed. In each frame $f = 1, \ldots, k$ we have n_f observations, labeled by an augmented index set I_f

$$I_f = \{0, 1, 2, \ldots, n_f\} \; ; \; 1 \leq f \leq k \tag{7}$$

where index 0 is a virtual or dummy index that allows us to reason over missed detections and partial trajectories [18]. The observations from frames 1 through k form a k-partite graph $G = (V, E)$ with

$$V = I_1 \cup I_2 \cup \cdots \cup I_k \tag{8}$$

$$E = (I_1 \mathrm{x} I_2) \cup (I_2 \mathrm{x} I_3) \cup \cdots \cup (I_{k-1} \mathrm{x}\, I_k) \; . \tag{9}$$

We treat G as an undirected graph.

Denote the set of all paths of length k through graph G by $\mathcal{T} = I_1 \times I_2 \times \cdots \times I_k$, and define a cost function $c : \mathcal{T} \rightarrow R$. Each path $t \in \mathcal{T}$ represents one hypothesized target trajectory, while $c(t)$ is a cost quantifying track quality. We write a k-path as an ordered list of its incident indices, $(i_1, i_2, \ldots, i_k)$, with $i_f \in I_f$ for $1 \leq f \leq k$.

The dummy index 0 plays an important role in increasing the flexibility of the k-path representation to handle partial trajectories. For example,

(0,0,a,b,c) : path that starts at frame 3
(a,b,c,0,0) : path that ends at frame 3
(0,0,a,0,0) : false positive detection in frame 3
(a,b,0,0,c) : missed detections or occlusion

Thus all trajectory hypotheses, including partial trajectories and false positives, are represented by complete paths of length k. The definition of trajectory disjointness is modified to exclude index 0 so that a variable number of missed detections can be represented in any given frame. We require each path to contain at least one real observation.

Finding the best set of disjoint trajectories can now be formulated as a multidimensional assignment problem, with binary decision variables $x_{i_1 i_2 \ldots i_k}$ for each k-path with associated costs $c_{i_1 i_2 \ldots i_k}$. We seek an optimal set of k-paths with respect to the following linear objective function and disjointness constraints

$$\min \sum_{i_1=0}^{n_1} \sum_{i_2=0}^{n_2} \cdots \sum_{i_k=0}^{n_k} c_{i_1 i_2 \ldots i_k} x_{i_1 i_2 \ldots i_k} \tag{10}$$

subject to

$$\sum_{i_2=0}^{n_2} \sum_{i_3=0}^{n_3} \cdots \sum_{i_k=0}^{n_k} x_{i_1 i_2 \ldots i_k} = 1; \quad i_1 = 1, 2, \ldots, n_1 \tag{11}$$

$$\sum_{i_1=0}^{n_1} \sum_{i_3=0}^{n_3} \cdots \sum_{i_k=0}^{n_k} x_{i_1 i_2 \ldots i_k} = 1; \quad i_2 = 1, 2, \ldots, n_2 \tag{12}$$

$$\vdots \qquad\qquad \vdots$$

$$\sum_{i_1=0}^{n_1} \sum_{i_2=0}^{n_2} \cdots \sum_{i_{k-1}=0}^{n_{k-1}} x_{i_1 i_2 \ldots i_k} = 1; \quad i_k = 1, 2, \ldots, n_k \tag{13}$$

Since the problem is NP-hard, it is infeasible to search for the exact optimal solution, and an approximate solution method is necessary.

Greedy forward sequential methods are one obvious approach. These are commonly found in the target tracking literature in the form of probabilistic recursive estimators, including single target filters such as the Kalman filter and PDAF, as well as multitarget filters such as JPDAF [19, 6]. These approaches require causal cost functions where computations at time t are based only on information observed up to time t. Because decisions, once made, are fixed, these methods are susceptible to making matches that are later revealed to be suboptimal. We use a greedy sequential method based on constant velocity motion prediction as a baseline for comparison in our experiments, as well as to initialize our iterative improvement approach.

Multi-hypothesis tracking (MHT) seeks stronger solutions through a deferred decision strategy [6]. Ambiguous matches are maintained until enough later information has been seen to disambiguate them. This leads to a combinatorially large, branching tree of hypotheses, and in practice suboptimal heuristic pruning decisions must be made [10]. Poore [18] and Deb *et al*. [20] present Lagrangian relaxation schemes for multidimensional assignment. These methods progressively relax the one-to-one matching constraints to generate a series of problems that are easier to solve, by inserting some of the matching constraints into the objective function as soft constraints using Lagrange multipliers. Both MHT and Lagrangean relaxation are complicated algorithms that are difficult to code and analyze. In the next section we propose a block-ICM iteration scheme that applies a series of optimal two-frame linear assignment decisions to monotonically improve an initial solution.

3.2. An Iterative Approximation

To introduce our solution strategy, first consider the notation for a simple 4-frame problem with observations indexed in the first frame by $I_a = (0, 1, \ldots, n_a)$, in the second frame by $I_b = (0, 1, \ldots, n_b)$, third frame $I_c = (0, 1, \ldots, n_c)$, and fourth frame $I_d = (0, 1, \ldots, n_d)$. Letting binary decision variable x_{abcd} be 1 if path (a,b,c,d) is in the solution, and zero otherwise, our multidimensional assignment problem of Eq. 10 can be written as

$$\min \quad \sum_{a=0}^{n_a} \sum_{b=0}^{n_b} \sum_{c=0}^{n_c} \sum_{d=0}^{n_d} c_{abcd} x_{abcd} \tag{14}$$

$$\text{s.t} \quad Ax = 1 \tag{15}$$

$$x \in \{0, 1\} \tag{16}$$

where A is a matrix representing the linear constraints in equations 11-13.

We further leverage the k-partite structure of the problem by noting that a path (a, b, c, d) is uniquely defined by its edge list $((a, b), (b, c), (c, d))$, so we can replace decision variables on the paths $\mathcal{T} = I_1 \times I_2 \times I_3 \times I_4$ with decision variables on the edges $E = (I_1 \times I_2) \cup (I_2 \times I_3) \cup (I_3 \times I_4)$. That is, we can factor x_{abcd} as $f_{ab} * g_{bc} * h_{cd}$, with $x_{abcd} = 1$ iff $f_{ab} = 1, g_{bc} = 1$, and $h_{cd} = 1$. This factoring is advantageous because the number of decision variables is now on the order of $(k-1) \times n^2$ instead of n^k. It also allows us to structure our objective function as

$$\sum_a \sum_b \sum_c \sum_d c_{abcd}\, x_{abcd} \tag{17}$$

$$= \sum_a \sum_b \sum_c \sum_d c_{abcd}\, f_{ab}\, g_{bc}\, h_{cd} \tag{18}$$

$$= \sum_a \sum_b f_{ab} \sum_c g_{bc} \sum_d h_{cd}\, c_{abcd} \tag{19}$$

$$\tag{20}$$

For a general k-frame problem with trajectory variables $x_{i_1 i_2 \ldots i_k}$ and associated costs $c_{i_1 i_2 \ldots i_k}$, we write the decision variable factorization as

$$x_{i_1 i_2 \ldots i_k} = \prod_{j=1}^{k-1} z_{i_j i_{j+1}}^{j\ j+1} \tag{21}$$

where notation $z_{i_j i_{j+1}}^{j\ j+1}$ is introduced to represent the factored decision variables on graph edges. The superscripts denote an edge that runs between frame j and frame $j+1$, and the superscripts denote that it is connecting observation i_j (in frame j) to observation i_{j+1} (in frame $j+1$). For example, $z_{3\ 4}^{1\ 2}$ represents an edge connecting the third observation in frame 1 with the fourth observation in frame 2. We then have

$$\sum_{i_1} \sum_{i_2} \cdots \sum_{i_k} c_{i_1 i_2 \ldots i_k} x_{i_1 i_2 \ldots i_k} \tag{22}$$

$$= \sum_{i_1} \sum_{i_2} \cdots \sum_{i_k} c_{i_1 i_2 \ldots i_k} \prod_{j=1}^{k-1} z_{i_j i_{j+1}}^{j\ j+1} \tag{23}$$

$$= \sum_{i_1} \sum_{i_2} z_{i_1 i_2}^{1\ 2} \sum_{i_3} z_{i_2 i_3}^{2\ 3} \cdots \sum_{i_k} z_{i_{k-1} i_k}^{k-1\ k} c_{i_1 i_2 \ldots i_k} \tag{24}$$

Note that the cost function $c_{i_1 i_2 \ldots i_k}$ is still a general function computed with respect to an entire trajectory.

The constraints 11-13 also need to be rewritten in terms of the new, factored decision variables, but this is easily achieved based on properties of the desired set partitioning. That is, for each node representing an observation, the sum of decision variables entering the node and the sum of decision variables exiting the node must be exactly one. Note that this is a stronger constraint than the typical flow conservation constraints in network flow, where the sum of flow into and out of a node can be either 0 or 1. The difference arises because our approach builds upon a set partitioning formulation rather than a set packing one.

Local improvement heuristic: We are now in a position to describe our local improvement heuristic, again illustrated with the 4-frame problem above. Assume we already have a feasible solution, meaning a binary labeling of decision variables f_{ab}, g_{bc} and h_{cd} satisfying all of the path disjointness constraints. Let the value of the objective function for this solution be $\mathcal{C}$. Without loss of generality, hold the labelings of all variables f_{ab} and h_{cd} fixed, and consider changes only to variables g_{bc} representing edges between pairs of observations in frames 2 and 3.

Define $a \to b_i$ to be the unique element $\{a | f_{ab_i} = 1\}$, and similarly $c_j \to d$ to be $\{d | h_{c_j d} = 1\}$. Then

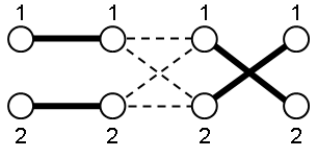
$$\sum_a \sum_b \sum_c \sum_d c_{abcd}\, f_{ab}\, g_{bc}\, h_{cd} \tag{25}$$

$$= \sum_b \sum_c g_{bc} \sum_a \sum_d f_{ab}\, h_{cd}\, c_{abcd} \tag{26}$$

$$= \sum_b \sum_c g_{bc}\, c_{(a \to b) bc (c \to d)} \tag{27}$$

$$= \sum_b \sum_c g_{bc}\, \omega(b, c) \tag{28}$$

Example: Consider two targets viewed through four frames as shown in the sketch below.



The thick lines represent variable values being held fixed, i.e. $f_{11} = f_{22} = h_{12} = h_{21} = 1$ and all other f and h decision variables are 0. The thin dashed lines represent the variables $g_{11}, g_{12}, g_{21}, g_{22}$ that we want to update. Applying the reduction from Eq 28 produces the two-frame cost matrix

$$\omega(b, c) = \begin{bmatrix} c_{1112} & c_{1121} \\ c_{2212} & c_{2221} \end{bmatrix}. \tag{29}$$

It is easily seen that Eq 28 is equivalent to a weighted maximum matching in a bipartite graph, aka the two-frame linear assignment problem. This subproblem can be solved exactly, in polynomial time, by a rectangular-matrix variant of the Kuhn-Munkres Hungarian algorithm [8].

Our proposed solution makes a series of iterations. At each iteration we step through pairs of adjacent frames

from 1-2 through (k-1)-k, updating the edge decision variables between them while holding all other decision variables fixed. An iteration thus consists of (k-1) two-frame linear assignment updates. In general notation, an update step involves solving for decision variables $z_{i_f i_{f+1}}^{f\ f+1}$ between frames f and $f+1$ holding all other variables fixed. Define $\{p \rightarrow i_f\}$ to be the current subpath from frame 1 to f ending in observation i_f. Similarly, define $\{i_{f+1} \rightarrow q\}$ to be the current subpath from observation i_{f+1} to frame k. The update step reduces to a two-frame assignment with objective function

$$\sum_{i_f} \sum_{i_{f+1}} z_{i_f i_{f+1}}^{f\ f+1}\ c_{\{p \rightarrow i_f\} i_f i_{f+1} \{i_{f+1} \rightarrow q\}}\ . \tag{30}$$

It is important to note that the objective function value $\mathcal{C}'$ after an update step can be no worse than the value $\mathcal{C}$ of the current solution, that is, $\mathcal{C}' \leq \mathcal{C}$. This is so because the current solution is among the set of solutions considered by each reduced two-frame assignment problem. As such, block-ICM inherits the convergence properties of regular ICM, in that it is guaranteed to eventually converge to a (local) optimum [5]. We also find experimentally that convergence occurs rapidly, usually within 5 iterations.

3.3. Decomposable Cost Functions

Although we have maintained the ability to use cost functions defined over arbitrarily long trajectories, many cost functions encountered in practice are defined over smaller temporal windows. In these situations, additional decomposition of the objective function may be possible.

For example, consider the case of cost functions that decompose into a product of pairwise costs

$$c_{i_1 i_2 \ldots i_k} = \prod_{j=1}^{k-1} c_{i_j i_{j+1}}^{j\ j+1}\ . \tag{31}$$

In this special case, Eq. 24 further simplifies to

$$\sum_{i_1} \sum_{i_2} z_{i_1 i_2}^{1\ 2} \sum_{i_3} z_{i_2 i_3}^{2\ 3} \cdots \sum_{i_k} z_{i_{k-1} i_k}^{k-1\ k} \prod_{j=1}^{k-1} c_{i_j i_{j+1}}^{j\ j+1} \tag{32}$$

$$= \sum_{i_1} \sum_{i_2} z_{i_1 i_2}^{1\ 2} c_{i_1 i_2}^{1\ 2} \sum_{i_3} z_{i_2 i_3}^{2\ 3} c_{i_2 i_3}^{2\ 3} \cdots \sum_{i_k} z_{i_{k-1} i_k}^{k-1\ k} c_{i_{k-1} i_k}^{k-1\ k} \tag{33}$$

and we see that each cost factor is now paired with an associated edge decision variable. The objective function therefore can be completely represented by a graph with weighted edges, as in the network flow formalism of [3]. This explains more precisely the relationship, as well as the limitations, of network flow data association with respect to the general data association problem. Network flow is a special case that only handles cost functions that can be factored into a product of costs computed in sequence between pairs of observations along a hypothesized trajectory. As we have already stated, this type of cost function has limited ability to represent motion models, as there is little that can be computed from pairs of locations other than distance.

3.4. Initialization and Termination

Our approach is an iterative improvement strategy, and therefore requires an initial feasible solution to get started. Because it is monotonically improving the solution (*i.e.* hill-climbing), starting with a good initial solution should yield convergence to a better local optimum. In our experiments we use a greedy baseline algorithm (Section 4) for initialization – this algorithm makes a series of bipartite assignments forward in time while using constant velocity motion prediction.

Typically, one runs an ICM-like algorithm until the value of the objective function stops improving. However, we also note that our approach maintains a feasible solution at every step, unlike, say, Lagrangean relaxation [6]. In time-critical applications, one could therefore use this approach as an "anytime" algorithm that can be terminated early while still providing a usable result.

4. Experimental Evaluation

In this section we evaluate our proposed iterative approximation approach against two baseline algorithms representing commonly-used alternatives. The first, called "Flow", is a network flow approach using pairwise distance as the cost of an edge between potential matches in adjacent frames. Since there is no appearance information being used, the objective function to be minimized reduces to finding the k shortest disjoint paths over all trajectories. The globally optimal solution to this objective is being found.

The second baseline algorithm, "Greedy", is a forward sequential filtering algorithm incorporating constant velocity motion prediction. For each current trajectory, the last two point locations define a velocity estimate that is used to predict a virtual point location in the next frame. Distances between predicted locations and observed targets form the cost matrix for a two-frame matching problem, solved by the Hungarian algorithm. Trajectories for which no matches are found are carried forward using the constant velocity prediction for a short period of time, but eventually die out. Observations for which no trajectory matches are found are used to start new trajectories. Once a decision has been made at a time step, it is fixed and cannot be undone.

4.1. Spline-based "Snake Energy" Cost Function

Motivated by the active contour model of Kass, Witkin and Terzopoulos[12], we define a spline-based cost function for our higher-order motion model :

$$cost(P) = \alpha E_{\text{cont}} + \beta E_{\text{curv}} \tag{34}$$

where

$$E_{\text{cont}} = \frac{1}{n-1}\sum_{i=2}^{n}\|p_i - p_{i-1}\| \quad (35)$$

$$E_{\text{curv}} = \sum_{i=2}^{n-1}\|p_{i+1} - 2p_i + p_{i-1}\|^2 . \quad (36)$$

This is a variant of the internal energy term of a "snake" active contour, and is applied to each hypothesized trajectory to compute the cost of that path. E_{cont} is the average distance between successive pairs of points, penalizing large jumps in position. E_{curv} is a sum of curvature terms over the length of the trajectory. In our experiments we set $\alpha = \beta = 1$. When there are only 2 points in the trajectory, $E_{\text{curv}} = 0$, and the cost reduces to distance between the points. We do not use any training data to tune cost function parameters, *e.g.* no distance or velocity thresholds and no knowledge of entry or exit regions.

Note that the curve of least energy with respect to E_{curve} is a natural cubic spline. Also, note that the curvature term is a finite difference computation of acceleration, and since we are minimizing, the objective function will automatically prefer piecewise constant velocity trajectories.

4.2. Evaluation

As a testset for evaluation, we have collected and groundtruthed two datasets of trajectories from pedestrians walking in an atrium[1]. One is a relatively "sparse" sequence, with an average of 5 observed people per frame. The second "dense" sequence is more challenging, with roughly 20 people observed per frame. The number of people in each frame is variable, since individuals may enter and exit the view at any time in the sequence.

Each sequence is 15 minutes long, and human-labeled annotations were used to generate ground truth locations of all people in every 10th frame, in a ground plane coordinate system. This trajectory data was then broken into 20 second sliding windows, each overlapping by 10 seconds, to provide a collection of smaller sequences for testing. To study the effect of sampling rate, we subsampled the data into test sets with 3 observations per second, 2 observations per second and 1 observation per second, expecting problems with a lower temporal sampling rate to be more difficult.

Table 1 shows a quantitative comparison of tracking performance. Algorithms evaluated are network flow (Flow), greedy sequential filtering (Greedy), and our block-ICM approach (Ours) using the snake energy cost function. Although both "Greedy" and our approach use constant velocity motion information, the snake energy model applies it to evaluate smoothness of entire trajectories, rather than to provide an inter-frame matching criterion.

[1]Dataset available from http://vision.cse.psu.edu/

Table 1. Mismatch error percentage for network flow (Flow), greedy forward sequential filtering (Greedy) and our approach (Ours) using the snake energy cost function. **Smaller numbers are better.** Approaches are compared on sparsely and densely populated sequences, for sampling rates ranging from 1 to 3 frames per second. Also shown in parentheses for our approach is the average number of iterations to convergence.

	Sparse Trajectories			Dense Trajectories		
	Flow	Greedy	Ours	Flow	Greedy	Ours
3fps	0.04	**0.00**	**0.00** (1)	0.23	**0.12**	0.13 (2)
2fps	0.28	**0.06**	0.12 (1)	1.36	0.34	**0.25** (2)
1fps	5.43	1.36	**0.80** (2)	21.35	6.83	**4.13** (5)

The error measure used is total mismatch error percentage (*mmep*), which is one component of the Multiple Object Tracking Accuracy (MOTA) error measure commonly used by the target tracking community [4]. It is computed as follows: let g(t) be the number of ground truth targets at time t, and let mme(t) be the number of mismatches (aka identity swaps) that occurred at time t within the estimated trajectories. Mismatch percentage *mmep* is then computed as $mmep = 100 \times (\sum_t mme(t) / \sum_t g(t))$. A higher mismatch percentage means a higher number of cases where an estimated trajectory incorrectly "jumps" from one individual to another during tracking. Sample images comparing network flow results with our results are shown in Figure 2.

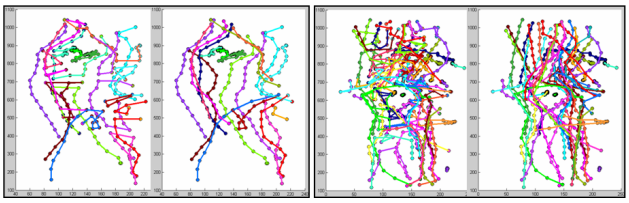


Figure 2. Sample trajectories. Far left: network flow results, with 22 ID swaps. Second from left: Our approach using snake energy on the same image yields 2 ID swaps. Second from right: network flow on a denser sequence, with 116 ID swaps. Far right: Our approach on the same image yields 16 ID swaps. All trajectories are color-coded with respect to ground truth; edges of good trajectories appear in the same color as the observations they connect.

Discussion: As expected, all methods perform better when the sampling rate is higher. However, at all sampling rates, even greedy forward selection outperforms the globally optimal shortest path solution computed by network flow, demonstrating the benefits of a constant velocity motion model for reducing mismatch errors. The improvement in performance is particularly notable for lower frame rates and for sequences with larger numbers of closely-spaced objects. Table 1 also shows average number of iterations until convergence for our iterative approach. These numbers confirm our observations that the block-ICM algorithm converges quickly.

5. Summary and Future Work

Starting with a generalized MDA framework [18], we factor the trajectory decision variables into a product of variables on edges in a k-partite trellis network representing multiframe observations. However, unlike network flow formalisms, we retain fully general cost functions that can use higher-order motion models to evaluate path quality. Although our problem formulation is NP-hard, we have proposed an iterative approximate solution method similar to ICM that cycles through pairs of adjacent frames, computing block-optimal two-frame linear assignment solutions while holding all other decision variables fixed. The method is guaranteed to converge, and usually converges rapidly. Using a "snake energy" trajectory cost function, our approach has been shown to outperform two common baseline algorithms for data association.

Our use of general cost functions allows evaluation of path quality over an entire trajectory of state vectors. In this paper we have only used object location as the state vector in order to isolate the effects of kinematic motion features from the confounding effects of appearance information. However, it is trivial to extend our approach to include shape and appearance information by augmenting each observation state vector with additional features such as bounding box width and height, detector confidence scores, normalized color histograms, or HOG descriptors. The cost function would then be able to compute additional quality measures based on average detector confidence, smoothness of variation of bounding box size/shape, and variance of appearance of the target with respect to its "mean" appearance over the whole trajectory. To properly incorporate this additional information, we will need to specify or learn relative weights for fusing different types of appearance, shape and motion cues into a single path quality score.

References

[1] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. *IEEE Conf on Computer Vision and Pattern Recognition*, June 2008. 1

[2] E. Balas and M. Padberg. Set partitioning: A survey. *SIAM Review*, 18:710–760, 1976. 2

[3] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:1806–1819, September 2011. 1, 2, 3, 6

[4] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: The CLEAR MOT metrics. *EURASIP Journal on Image and Video Processing, Special Issue on Video Tracking in Complex Scenes for Surveillance Applications*, 2008, May 2008. Article ID 246309. 7

[5] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 48(1):259–302, 1986. 3, 6

[6] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Sys*. Artech House, Norwood, MA, 1999. 1, 3, 4, 6

[7] W. Brendel, M. Amer, and S. Todorovic. Multiobject tracking as maximum weight independent set. In *IEEE Conf on Computer Vision and Pattern Recognition*, pages 1273–1280, 2011. 2, 3

[8] R. Burkard, M. Dell'Amico, and S. Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2009. 3, 5

[9] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. *European Conference on Computer Vision*, pages 492–505, 2010. 3

[10] I.J.Cox and S.L.Hingorani. An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Trans on Pat Analysis and Mach Intell*, 18(2):138–150, 1996. 1, 4

[11] H. Jiang, S. Fels, and J. J. Little. A linear programming approach for multiple object tracking. In *IEEE Computer Vision and Pattern Recognition*, pages 744–750, 2007. 3

[12] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987. 6

[13] B. Leibe, K. Schindler, and L. J. V. Gool. Coupled detection and trajectory estimation for multi-object tracking. *International Conference on Computer Vision*, pages 1–8, 2007. 3

[14] C. Morefield. Application of 0-1 integer programming to multitarget tracking problems. *IEEE Transactions on Automatic Control*, 22(3):302–312, June 1977. 2

[15] S. Oh, S. Russell, and S. Sastry. Markov chain monte carlo data association for multi-target tracking. *IEEE Trans on Automatic Control*, 54(3):481–497, March 2009. 3

[16] D. J. Papageorgiou and M. R. Salpukas. The maximum weight independent set problem for data association in multiple hypothesis tracking. *Optimization and Cooperative Control Strategies, Lecture Notes in Control and Information Sciences*, 381:235–255, 2009. 2, 3

[17] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. *IEEE Conf on Computer Vision and Pattern Recognition*, pages 1201–1208, June 2011. 1, 2, 3

[18] A. B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking. *Computational Optimization and Applications*, 3:27–57, March 1994. 3, 4, 8

[19] C. Rasmussen and G. Hager. Probabilistic data association methods for tracking complex visual objects. *IEEE Trans on Pat Analysis and Mach Intell*, 23(6):560–576, June 2001. 4

[20] S.Deb, M.Yeddanapudi, K.Pittipati, 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. 3, 4

[21] K. Shafique and M. Shah. A noniterative greedy algorithm for multiframe point correspondence. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 27(1):51–65, 2005. 3

[22] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. *IEEE Conf on Computer Vision and Pattern Recognition*, pages 523–538, June 2008. 1, 2, 3