A Noniterative Greedy Algorithm for Multiframe Point Correspondence

Khurram Shafique, *Student Member*, *IEEE*, and Mubarak Shah, *Fellow*, *IEEE*

Abstract—This paper presents a framework for finding point correspondences in monocular image sequences over multiple frames. The general problem of multiframe point correspondence is NP-hard for three or more frames. A polynomial time algorithm for a restriction of this problem is presented and is used as the basis of the proposed greedy algorithm for the general problem. The greedy nature of the proposed algorithm allows it to be used in real-time systems for tracking and surveillance, etc. In addition, the proposed algorithm deals with the problems of occlusion, missed detections, and false positives by using a single noniterative greedy optimization scheme and, hence, reduces the complexity of the overall algorithm as compared to most existing approaches where multiple heuristics are used for the same purpose. While most greedy algorithms for point tracking do not allow for entry and exit of the points from the scene, this is not a limitation for the proposed algorithm. Experiments with real and synthetic data over a wide range of scenarios and system parameters are presented to validate the claims about the performance of the proposed algorithm.

Index Terms—Point correspondence, target tracking, motion, occlusion, point trajectory, data association, bipartite graph matching, path cover of directed graph.

1 Introduction

In motion correspondence, given an image sequence, the problem is to find the correspondences between the feature points in the images that occur due to the same object in the real world at different time instants. Once these correspondences among all feature points are available, they can be used for many applications like object tracking, motion analysis, optical flow, and structure from motion [1], [15], [17], [26], [33], [34], [38].

The output of a correspondence algorithm is a set of tracks, where each track ideally corresponds to a unique point or an object in the real world and specifies its position in every frame from entry to exit in the scene. We assume that the only information available about the feature points is their position in the image and there is no other distinguishing feature among these points. This type of scenario occurs in applications like particle tracking or tracking of a large number of similar objects. Psychological experiments have shown that human vision is less sensitive to the form and appearance of objects as compared to their velocity and position [9]. It has also been shown that humans are capable of making inferences about the type of object and its motion by using the minimal information of velocity and position of a small number of otherwise indistinguishable points on the object [19].

We formulate the problem as follows. The same notation will be used throughout the rest of the paper. Let a sequence of n frames F_i (each of dimensions $S_x \times S_y$) correspond to n time instances $t_i, 1 \leq i \leq n$, and let $X_i = \{x_1^i, x_2^i, \ldots, x_r^i\}$ be the set of r points detected in the frame F_i (the number of points detected in each frame need not be the same). We define a *track* T of length m to be a sequence of m points $\langle x_{a_1}^{i_1}, x_{a_2}^{i_2}, \ldots, x_{a_m}^{i_m} \rangle$ such that $1 \leq i_1 < i_2 < \ldots \leq n$ and $1 \leq a_j \leq |X_{i_j}|$. The length of a track T is denoted by $|T|$. The *backward correspondence* of a point $x_{a_j}^{i_j}$ in track T is defined by the point preceding $x_{a_j}^{i_j}$, i.e., $x_{a_{j-1}}^{i_{j-1}}$, while the *forward correspondence* of a point is the point succeeding $x_{a_j}^{i_j}$, i.e., $x_{a_{j+1}}^{i_{j+1}}$. The first point of a track T has no backward correspondence and the last point has no forward correspondence. We assume that the detected points occur either due to sensory response of one or more real-world points or due to sensor noise.

The problem is to find a set of tracks $A = \{T_1, T_2, \ldots, T_m\}$ such that $\forall T_i \in A$, either one of the following is true:

- If $\exists x_j^k \in T_i$, such that x_j^k is a sensor response of *only* point Z_i in the real world, then every point in T_i is a sensor response of Z_i (or more points in occlusion with Z_i) and no other track T_r contains a sensor response of *only* Z_i.
- $\forall x_j^k \in T_i, x_j^k$ occurs due to sensor noise.

The first condition requires each real-world point to have exactly one track associated with it (element integrity principle [9]) and each track to be associated to exactly one world point. The second condition disallows noisy detections or false positives to be part of any track corresponding to a real-world point. Thus, by the above definition, a track is either composed of sensor responses of a single world point (may include responses due to the occlusion of that point by other points) or it is composed of points occurring from noise only. Hence, the problem is not only to find these tracks, but also to distinguish between these two types of tracks for noise removal. This distinction is usually done by higher level processes and is not in the scope of this paper.

We propose a look-ahead technique to solve the correspondence problem by using a sliding window over multiple

- *The authors are with the School of Computer Science, University of Central Florida, Computer Science Bldg. (No. 54), 4000 Central Florida Blvd. Orlando, FL 32816. E-mail: {khurram, shah}@cs.ucf.edu.*

Manuscript received 21 May 2003; revised 12 Apr. 2004; accepted 3 May 2004.
Recommended for acceptance by A. Rangarajan.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0095-0503.

frames. This information gathering over time for systems simulating the cognitive processes is supported by many researchers in both vision and psychology (e.g., [13], [21], [22]). Neisser [22] proposed a model according to which the perceptual processes continually interact with the incoming information to verify hypotheses formed on the basis of available information up to a given time instant. Marr's principle of least commitment [21] states that any inference in a cognitive process must be delayed as much as possible. The use of active information pickup by a human observer in an environment is supported by [13]. Todd [32] argued that most human observers require several frames to infer the structure of moving objects, even in simple experimental settings. Many existing algorithms use similar look-ahead strategies or information gathering over longer intervals of time (for example, by backtracking) [10], [23], [27], [30]. The comparison of these techniques with our work is presented in later sections.

The major contribution of this paper is the formulation of an efficient and robust solution to the multiframe correspondence problem as defined above. The proposed framework deals with the problems of occlusion handling, missed detections, and false positives by using a single greedy optimization scheme and, hence, reduces the complexity of the overall algorithm as compared to most existing approaches where different heuristics are used for the same purpose. The experimental results on both real and synthesized sequences show that the proposed algorithm outperforms the existing methods, is time-efficient, and is applicable in more general scenarios.

The organization of the paper is as follows: In the next section, we present a survey of the related work. In Section 3, we define the terminology and notation for this paper and provide a graph theoretical formulation of the correspondence problem and its solution in Section 4. We refine the solution of Section 4 and present details of the proposed algorithm in Section 5. The choice of gain function is discussed in Section 6. In Section 7, we demonstrate the results of the proposed approach on a variety of synthesized and real sequences and compare our results to the previous approaches. Section 8 concludes the paper.

2 Related Work

A large number of correspondence methods have been proposed in recent years. Most of these methods first define a motion model and use some optimization technique to maximize (minimize) a gain (cost) function based on that motion model. These methods differ by the choice of motion model, optimization technique, and/or gain function. Ullman [34] proposed a minimal mapping approach, where the probabilistic cost function was based on the distance between the points in consecutive frames. A linear programming approach was used to minimize the cost function. The cost function was further improved by Jenkin [18], who introduced the smoothness constraint along with the nearest neighbor relationship. He used a greedy approach for optimization. Barnard and Thompson [2] used a relaxation-based approach to solve this problem. All of the above algorithms use two frames for establishing the correspondences. Sethi and Jain [29] proposed an iterative greedy exchange algorithm using both nearest neighbor and smoothness constraints. The self initializing version of the algorithm repeats the optimization step in forward and backward directions until an equilibrium state is achieved. The algorithm however assumes that the points do not enter or exit the scene and that there is no occlusion and detection errors. The latter condition was relaxed by Salari and Sethi in [28].

Rangarajan and Shah [25] proposed the proximal uniformity constraint and a noniterative greedy algorithm that uses three frames to establish correspondences. The algorithm assumes a fixed number of real-world points, but allows for temporary occlusion or missed detections.

The most recent contribution in the area is by Veenman et al. [35], who proposed Hungarian search as an optimization tool for their GOA Tracker, along with the motion models defined in [4], [25], [29]. The basic algorithm assumes that the initial correspondence between the first two frames is known and establishes further correspondences based on consecutive frames only. An extension to the basic algorithm for self-initialization backtracks the correspondences over all frames once all correspondences are established by the basic algorithm. The algorithm also assumes that the number of world points remains the same over all frames, though it allows for occlusions and detection errors, which is done by using heuristics based on the analysis of the number of detected points in consecutive frames.

Apart from these methods, quite a few algorithms have been proposed in the statistical domain of which the most well-known are the Joint Probabilistic Data-Association Filter [11] and Multiple Hypothesis Tracking (MHT) [27]. The former is a greedy approach and it is more efficient than the latter, which suffers from combinatorial explosion. More efficient approximations of MHT have been presented. Some of these techniques use Murti's Algorithm to find the k best hypothesis to reduce the search space [5], [7], [6], [8], whereas others reduce the search space by using a limited temporal scope and a sliding window technique similar to ours. However, the problem remains intractable and further approximations are used for efficient implementation [10], [23], [24]. The major drawbacks of these methods are the large number of parameters and the assumptions about the probability distributions, which do not necessarily hold [35]. For a detailed review on statistical techniques for point correspondence, see [5].

Our contribution in this paper is the presentation of an optimization algorithm that optimizes the gain (cost) function over multiple frames and is general enough to be used for a large variety of motion models and cost functions (including statistical based functions) that satisfy the constraints as posed by the proposed framework. Our work is most closely related to Veenman et al. [35]. However, we present a solution to the "Multiframe" correspondence problem as opposed to the 2-frame correspondence problem in [35]. The latter is a special case of the former and is inherently an easy problem for which a polynomial time optimal solution exists. In addition, GOA assumes that the number of points in the scene remains constant, which is not a restriction for the proposed algorithm. Further, the

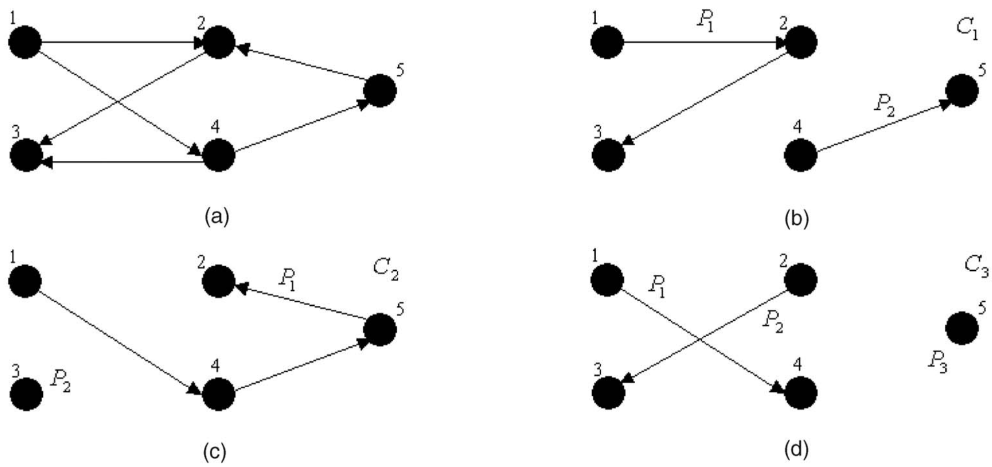


Fig. 1. (a) A sample digraph D. (b), (c), and (d) Three different vertex disjoint path covers C_1, C_2, and C_3 of digraph D.

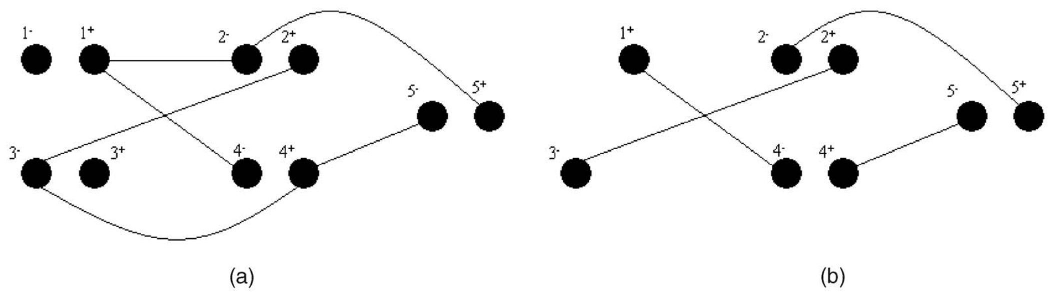


Fig. 2. (a) Split graph G of the sample digraph D shown in Fig. 1a. (b) A matching of the graph G.

self-initializing version of GOA is a two-pass algorithm compared to the proposed algorithm, which is a single pass algorithm and, hence, is applicable in real-time systems.

3 Definitions and Notations

We assume that the reader is familiar with the basic graph theory terminology. We follow the notation and terminology of [37]. Let $D = (V, E)$ be an edge weighted directed graph without self loops and multiple edges, where V and E are, respectively, the set of vertices and edges of the digraph D. A *directed path* P of length k in digraph D is a sequence of vertices $v_1, v_2, \ldots, v_{k+1}$, where $v_i \in V$, and, for every v_i, $1 \leq i \leq k$, there is a directed edge in E from v_i to v_{i+1}, for example, P_1, P_2, P_3 in Figs. 1b, 1c, and 1d are directed paths of the digraph shown in Fig. 1a. A *vertex disjoint path cover* C of D is a set $\{P_1, P_2, \ldots, P_k\}$ of directed paths P_i (of length ≥ 0), if $V = \bigcup_{i=1}^{k} V(P_i)$ and $V(P_i) \cap V(P_j) = \emptyset$ whenever $i \neq j$, where $V(P_i)$ is the set of vertices of directed path P_i. In other words, a vertex disjoint path cover C of a digraph D is a set of directed paths in D such that every vertex of D is in some path of C and no two paths in C have a common vertex. For simplicity of notation, we will refer to vertex disjoint path covers as path covers. Figs. 1b, 1c, and 1d show three different path covers, C_1, C_2, and C_3, of the digraph shown in Fig. 1a. Let $W(C)$ denote the weight of path cover C, where $W(C)$ is defined by the sum of weights of all the edges in the cover C. A *maximum weight path cover* of D is a path cover $C(G)$ such that $C(G) = \arg\max_{C_i} W(C_i)$, for all path covers C_i of D.

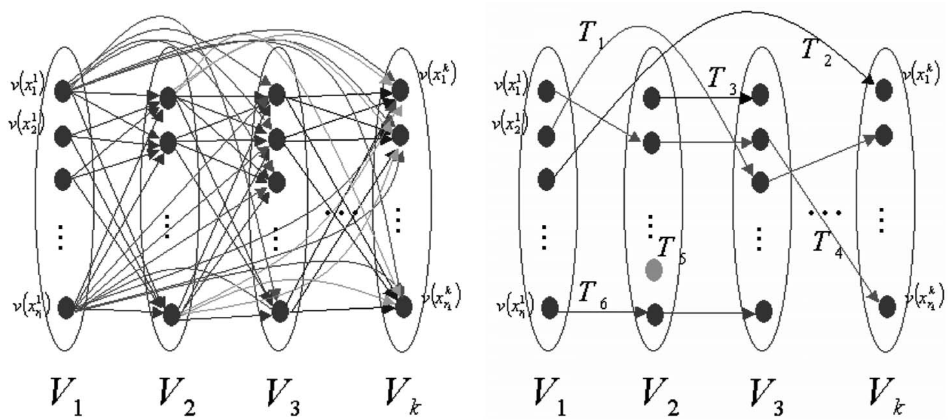
A *Split* of an edge weighted digraph D is an edge weighted bipartite graph G whose partite sets V^+, V^- are copies of $V(D)$. For each vertex $x \in V(D)$, there is one vertex $x^+ \in V^+$ and one vertex $x^- \in V^-$. For each edge e from u to v in D, there is a corresponding edge e' with endpoints u^+, v^- in G such that $w(e') = w(e)$. The split graph of the digraph in Fig. 1a is shown in Fig. 2a.

A *matching* in a graph G is a set of edges with no shared end-vertices. A *maximum (minimum) matching* in a weighted graph is a matching with maximum (minimum) weight among all matchings in the graph. A matching of the split graph in Fig. 2a is shown in Fig. 2b.

A graph $G' = (V', E')$ is a *subgraph* of a graph $G = (V, E)$, written $G' \subseteq G$ if $V' \subseteq V$ and $E' \subseteq E \cap V' \times V'$. If $S \subseteq E$, the subgraph induced by S is the graph $G[S] = (V', S)$ such that V' is a minimal subset of V containing the end vertices of all the edges of S. For example, the subgraph induced by edges $\{12, 14, 45\}$ in graph of Fig. 1a is the graph with vertex set $\{1, 2, 4, 5\}$ and edge set $\{12, 14, 45\}$.

4 Graph Theoretical Formulation

In this section, we present a graph theoretical formulation of the multiframe correspondence problem as defined in Section 1. There is an obvious graph theoretical formulation of the 2-frame correspondence problem, as has been observed in [34] and [35]. The problem can be viewed as finding a maximum matching of a bipartite graph G, where the partite sets V_1, V_2 correspond to the sets of points X_1 and X_2 detected in frames F_1 and F_2, respectively. We will denote the vertex in G corresponding to a point x by $v(x)$.



(a) (b)

Fig. 3. (a) An instance of digraph D as defined in Section 4 and (b) a candidate solution.

The edge $e = v(x_i^1)v(x_j^2)$ corresponds to a match hypothesis between the point x_i^1 in frame F_1 and the point x_j^2 in frame F_2, whereas the edge weight $w(e)$ is the gain $g(x_i^1, x_j^2)$ associated with this match. Hence, by finding the maximum matching of G, we find the match of every point x_i^1 in frame F_1 with some point x_j^2 in frame F_2 such that the total gain is maximized. Several efficient algorithms to find the maximum matching of a bipartite graph exist, for example, Hungarian search [20], which is $O(n^3)$ and an $O(n^{2.5})$ algorithm by Hopcroft and Karp [14], where n is the total number of vertices in graph G.

Since an efficient algorithm for the 2-frame correspondence problem exists, it seems natural to apply the same technique to the k-frame problem, $k \geq 3$. However, the k-dimensional matching problem is NP-Hard for $k \geq 3$ [12], i.e., all known solutions are exponential, and requires further approximations for all practical purposes [10], [24].

To avoid the high computational complexity involved, with the k-D matching, instead of using a k-partite hypergraph to model the k-frame problem, we construct a weighted digraph $D = (V, E)$ such that $\{V_1, V_2, \ldots, V_k\}$ partitions V, where each V_i corresponds to the set of points X_i detected in frame F_i. Further, $E = \{v(x_a^i)v(x_b^j) | v(x_a^i) \in V_i \wedge v(x_b^j) \in V_j, \forall i < j\}$, i.e., there is a directed edge from every vertex in set V_i to every vertex in set V_j whenever $i < j$. As in the 2-frame problem, each edge $e = v(x_a^i)v(x_b^j)$ corresponds to a match hypothesis of point x_a^i in frame F_i to point x_b^j in frame F_j, whereas the edge weight $w(e)$ is the gain $g(x_a^i, x_b^j)$ associated with this match. Let x_a^i be a point detected in frame F_i. The edge from $v(x_a^i)$ to vertex $v(x_m^{i+l}) \in V_{i+l}$ represents the possibility of x_a^i having no corresponding point in any frame from F_{i+1} to F_{i+l-1} and x_m^{i+l} being the forward correspondence of x_a^i. Thus, all the possibilities of detection errors and occlusions (or absence of a maximum of $k-2$ frames) are considered. A sample digraph formed in this way is shown in Fig. 3a.

By definition of the correspondence problem in Section 1, the task is to find a set of vertex disjoint directed paths (Tracks) of length 0 or more such that the total gain is maximum among all such paths, i.e., we want to find a maximum weight path cover of the directed graph D. A candidate solution to the problem is shown in Fig. 3b.

Let $T = \langle x_{a_1}^{i_1}, x_{a_2}^{i_2}, \ldots, x_{a_m}^{i_m} \rangle$ be a track corresponding to some real world point Z_i, we require that $\forall p, q,\ 1 < p+1 < q \leq m$, the gain function $g(x_{a_p}^{i_p}, x_{a_q}^{i_q})$ satisfies the following inequality:

$$g\left(x_{a_p}^{i_p}, x_{a_q}^{i_q}\right) < g\left(x_{a_p}^{i_p}, x_{a_{p+1}}^{i_{p+1}}\right) + g\left(x_{a_{q-1}}^{i_{q-1}}, x_{a_q}^{i_q}\right). \quad (1)$$

This condition guarantees that the total gain is maximized only if all the edges of T are in the path cover and penalizes the choice of a shorter track when a longer valid track is present. Note that the inequality reduces to strong triangular inequality when $m = 3$.

Once again, the problem of finding maximum path cover is NP-Hard, even in the case of unweighted graphs [3]. However, by the following theorem, a polynomial solution exists if the directed graph is acyclic:

Theorem 1. *The edges of maximum matching of the split graph G of an acyclic edge-weighted digraph D correspond to the edges of maximum path cover of D.*

Proof. Let D be an acyclic edge weighted digraph and G be its split graph, where $V^+,\ V^-$ form the partite sets of G. Let C^* be a maximum path cover of digraph D. Every vertex $y \in V(D)$ has indegree and outdegree at most one in digraph $D[C^*]$ induced by the edges in C^*. Since indegree and outdegree of a vertex $y \in V(D)$ correspond to the degrees of vertices $y^+ \in V^+$ and $y^- \in V^-$, respectively, if M^* is the set of corresponding edges of C^* in G, every vertex $x \in V(G)$ has degree at most one in $G[M^*]$ and, hence, M^* is a matching of graph G and $W(C^*) = W(M^*)$.

Now, let M be a maximum matching in G and let C be the set of corresponding edges in D. By an argument similar to the one above, every vertex in $V(D)$ has at most one edge in C coming into it and at most one edge in C going out of it. Thus, the directed subgraph $D[C]$ of D induced by the edges in C consists of vertex disjoint paths or cycles. Since D is acyclic, $D[C]$ consists only of vertex disjoint paths. Suppose that $W(C^*) > W(C)$, then $W(M^*) = W(C^*) > W(C) = W(M)$, which is contrary to M being a maximum matching. Hence, C is a maximum path cover of digraph D. □

By the construction of digraph D as described above, all the edges in D are in the direction of increasing time, thus D is acyclic. Hence, given the weighted directed graph D, the

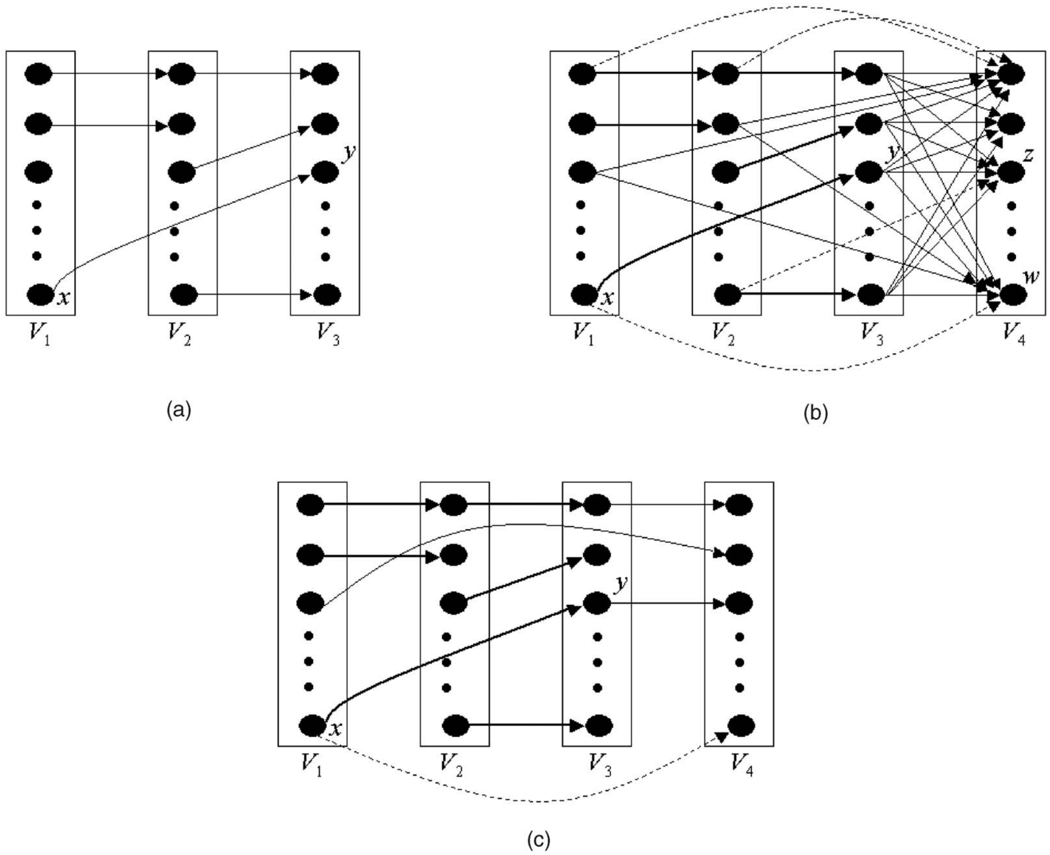


Fig. 4. (a) Initial correspondences. (b) The extension digraph D. (c) A maximum path cover of extension graph. The correction edges are shown as dotted lines, while the old edges are shown by bold lines. (Not all edges and vertices are shown.)

optimal set of tracks that maximizes the overall gain can be obtained in polynomial time. Note that, when the number of frames is two, the above problem reduces to the bipartite graph matching problem as described in the beginning of this section.

The above algorithm was successfully applied to the problem of establishing correspondences across multiple cameras with disjoint field of views (see [16]).

5 Greedy Algorithm

Thus far, we have assumed that the gain function g that is used for obtaining the edge weights of digraph is already given or is easily computable. This, however, is not always the case. It is easy to see that the construction of a digraph as mentioned in Section 4 assumes the gain $g(x_a^i, x_b^j)$ to be independent of the backward correspondences of x_a^i. For cases such as correspondence across multiple cameras [16] or gain functions based on the nearest neighborhood criteria [34], appearance, and/or color properties of the feature points, this condition is satisfiable. However, if the gain function, $g(x_a^i, x_b^j)$, requires velocity or acceleration of point x_a^i (which is computable only if the backward correspondence of x_a^i is known), this condition is not true. Similarly, gain functions based on smoothness of motion ([29]) do not satisfy this condition. Thus, the framework defined in Section 4 is not directly applicable when the gain function is dependent on backward correspondences. Since we are more interested here in gain functions based on motion information, we present a solution to this problem by proposing a greedy algorithm based on the framework of Section 4.

The algorithm assumes that the correspondences of points in $k-1$ frames, $F_1, F_2, \ldots, F_{k-1}$, $k > 2$, have been established. These correspondences in the previous $k-1$ frames were made by the information available up to time instant t_{k-1} and may be changed once more information is available. For every incoming frame, the proposed greedy algorithm modifies the established correspondences of the previous $k-1$ frames by either extending the tracks to the new frame (this includes the tracks of points occluded for one or more frames) or by correcting a previous mismatch by using the newly obtained information. Let F_k be the current frame and let C_{k-1} be the set of correspondences established up to frame F_{k-1}. The problem of extending the correspondences in C_{k-1} to the points in frame F_k can be represented by a digraph $D = (V = \bigcup_{i=1}^{k} V_i, E)$ such that, $V_1, V_2, \ldots, V_k$ are pairwise disjoint and each vertex $v(x) \in V_i$ corresponds to a point $x \in X_i$. Each track $T = \langle x_1, x_2, \ldots, x_p \rangle$ in C_{k-1} is modeled by the directed edges from vertex $v(x_i)$ to vertex $v(x_{i+1})$, $1 \leq i < p$. We refer to these edges as *old edges* (Fig. 4). The possible extension of every track to the new frame, F_k, is modeled by connecting the terminal vertex (vertex with no forward correspondence in C_{k-1}) of each track to every vertex in the vertex set V_k (i.e., set of vertices corresponding to frame F_k) by a directed edge. These edges are referred to as *extension edges*. The possibility of previous mismatches is represented by the directed edges from each nonterminal vertex of every track to the vertices in

Procedure FalseHypothesisReplacement()

for $i = 1$ *to* $k - 1$

begin

While there is a false hypothesis originating from some vertex in V_i

begin

Delete all the false hypotheses in graph D

Solve the $(k - i + 1)-$*frame correspondence problem for all the uncorresponded points*

in sets $F_i, F_{i+1}, \ldots, F_k$ *(Call Procedure* MultiframeCorrespondence($F_i', F_{i+1}', \ldots, F_k'$),

where F_j' *is the set of uncorresponded points in frame* F_j, $i \leq j \leq k$)

end

end

Fig. 5. Recursive algorithm for replacing false hypotheses.

Procedure NonRecursiveFalseHypothesisReplacement()

for $i = 1$ *to* $k - 1$

begin

Delete the false hypotheses originating from vertices in V_i

Solve the $2-$*frame correspondence problem for all the uncorresponded vertices in sets* V_i, V_{i+1}

end

Fig. 6. Nonrecursive heuristic for replacing false hypotheses.

the set V_k. We call these edges, *correction edges* (the old edges, extension edges, and correction edges are shown as bold, normal, and dotted arrows, respectively, in Figs. 4b and 4c). Formally, for every vertex pair $v(x) \in V_i,\ v(y) \in V_j$, there is an edge from $v(x)$ to $v(y)$ if either $i < j < k$ and there is a correspondence from x to y in C_{k-1} or $i < j = k$. The digraph D obtained in this way is called an *extension digraph* (Fig. 4b). Since all the backward correspondences except for the points in X_k have been established, all the edge weights in D can now be computed, regardless of the type of gain function used.

Once the extension digraph is constructed, we again seek a set of vertex disjoint paths (tracks) in D that maximizes the total gain. Thus, the same algorithm as presented in Section 4 can be applied to the digraph D. A candidate solution C_k may contain all three types of edges, i.e., old edges, correction edges, and extension edges. While an extension edge does not change any correspondence in C_{k-1}, a correction edge $e' = v(x)v(w)$ always replaces some existing edge or old edge $e = v(x)v(y)$. Suppose now that the point y has a forward correspondence z in C_k (Figs. 4b and 4c). Since this correspondence was obtained by assuming the correspondence xy in C_{k-1} and the correspondence xy is voided in C_k, the correspondence yz and all such forward correspondences must be removed from C_k and, if possible, be replaced with new edges. These edges will be referred to as *false hypotheses*. We define an edge to be a false hypothesis if it has a directed path from an edge replaced by a correction edge. The edge yz in Fig. 4c is a "false hypothesis." The replacement of false hypotheses with the new edges can be performed by the recursive scheme of Fig. 5. We have empirically determined that the nonrecursive heuristic of Fig. 6 also performs reasonably well in most cases and is more efficient than the recursive version (Section 7). After the false hypotheses are removed and replaced, we obtain a new correspondence set C_k of all the points up to frame F_k. These correspondences can be extended to any number of frames by adding one frame at a time in a similar fashion.

The initialization is done by first using the 2-frames algorithm (bipartite graph matching) to obtain the correspondences between the first two frames F_1 and F_2. These correspondences are then extended for each new frame by using the algorithm described above up to the kth frame. At this stage, a backtracking is performed by applying the same algorithm in the reverse direction, i.e., on frames $F_k, F_{k-1}, \ldots, F_1$, using the established correspondences. This takes care of any incorrect correspondence that was

Procedure MultiframeCorrespondence$(F_1, F_2, \ldots, F_n)$

Establish the initial correspondence between Frames F_1 *and* F_2 *by using the 2-frames algorithm (bipartite graph matching).*

$r \longleftarrow 2$

For each new frame F_i

begin

Construct the extension digraph D *using the frames* $F_{i-r}, F_{i-r+1}, \ldots, F_i$

Compute the edge weights $w(e)$ *using the given gain function* g

Find the maximum path cover of graph D

Replace false hypotheses by either FalseHypothesisReplacement() or NonRecursiveFalseHypothesisReplacement()

If $F_i = F_k$ *then perform backtracking using frames* $F_k, F_{k-1}, \ldots, F_1$

$r \longleftarrow r + 1$

if $r \geq k$ *then* $r \longleftarrow k - 1$

end

Fig. 7. Complete algorithm for the Multiframe Correspondence Problem.

made when either no or less motion information was available (i.e., first two frames) and is done just once for the first k frames. Once the correspondence for k frames is established, the correspondence is extended to each new frame F_i by using digraph D constructed from the last k frames $F_{i-k+1}, F_{i-k+2}, \ldots, F_i$. The complete algorithm for the window of size k is presented in Fig. 7.

6 Gain Functions

In this section, we address the following question: Given a partial track $T = \langle x^{i_1}_{a_1}, x^{i_2}_{a_2}, \ldots, x^{i_r}_{a_r} \rangle, 1 \leq i_1 < i_2 < \ldots < i_r$ and a point x^b_a in frame b, $b > i_r$, what is the gain $g(x^{i_j}_{a_j}, x^b_a)$ associated to assigning x^b_a as a forward correspondence of $x^{i_j}_{a_j}$ for all j, $1 \leq j \leq r$. Let $x^{i_j}_{a_j}x^b$ be the predicted motion vector for a point $x^{i_j}_{a_j}$ (this vector is predicted by making some assumption about the motion of point $x^{i_j}_{a_j}$, e.g., constant velocity or constant acceleration). The gain $g(x^{i_j}_{a_j}, x^b_a)$ is defined based on the match between $x^{i_j}_{a_j}x^b$ and the candidate motion vector $x^{i_j}_{a_j}x^b_a$. The simplest of these gain functions is based on the distance between the predicted and observed position of the point and is defined as follows:

$$g_1(x^{i_j}_{a_j}, x^b_a) = 1 - \frac{\| x^{i_j}_{a_j}x^b - x^{i_j}_{a_j}x^b_a \|_p}{\sqrt[p]{S^p_x + S^p_y}}. \tag{2}$$

Here, $\|.\|_p$ denotes the p-norm, whereas S_x and S_y are the x and y dimensions of the image, respectively.

The criteria defined by $g_1(x^{i_j}_{a_j}, x^b_a)$ prefers the match which is closest to the expected position of the point $x^{i_j}_{a_j}$. It reduces to the nearest neighborhood criteria when no motion information about point $x^{i_j}_{a_j}$ is available. However, the function ignores any information about the direction of motion and may allow very nonsmooth trajectories.

Sethi and Jain [29] proposed the gain function (also used in [35]) that prefers smooth changes in the magnitude and direction of motion. A slightly modified version of this function is as follows:

$$\begin{aligned} g_2(x^{i_j}_{a_j}, x^b_a) = \; & \alpha \left[\frac{1}{2} + \frac{x^{i_j}_{a_j}x^b \cdot x^{i_j}_{a_j}x^b_a}{2\|x^{i_j}_{a_j}x^b\| \|x^{i_j}_{a_j}x^b_a\|} \right] + \\ & (1-\alpha) \left[2 \frac{\sqrt{\|x^{i_j}_{a_j}x^b\| \|x^{i_j}_{a_j}x^b_a\|}}{\|x^{i_j}_{a_j}x^b\| + \|x^{i_j}_{a_j}x^b_a\|} \right], \; \alpha \in [0, 1]. \end{aligned} \tag{3}$$

The function is the convex combination of two terms called *directional coherence* and *speed coherence*, respectively. The first term penalizes large deviations in the direction of motion, while the second term prefers the match with less change in the magnitude of velocity (acceleration, etc.). The disadvantage of this scheme is its complete reliance on motion information, which is why it requires initialization of correspondence using some other criteria and does not support points entering the scene. We therefore, utilizing the advantages of both gain functions g_1 and g_2, define our gain function to be the convex combination of g_1 and the directional coherence term of g_2 as follows:

$$g_3(x_{a_j}^{i_j}, x_a^b) = \alpha\left[\frac{1}{2} + \frac{\overrightarrow{x_{a_j}^{i_j} x^b}\cdot\overrightarrow{x_{a_j}^{i_j} x_a^b}}{2\|\overrightarrow{x_{a_j}^{i_j} x^b}\|\|\overrightarrow{x_{a_j}^{i_j} x_a^b}\|}\right] + (1-\alpha)\left[1 - \frac{\|\overrightarrow{x_{a_j}^{i_j} x^b} - \overrightarrow{x_{a_j}^{i_j} x_a^b}\|_p}{\sqrt{S_x^2 + S_y^2}}\right],\ \alpha \in [0,1]. \quad (4)$$

When no velocity information is available, the function g_3 reduces to the nearest neighbor criteria. To satisfy the constraint of (1), we add a small constant penalty ϵ to the gain function $g_3(x_{a_j}^{i_j}, x_a^b)$ whenever $b > i_j + 1$.

7 Results

In this section, we present the results of the proposed algorithm on both synthetic and real sequences. We use the gain function g_3 of (4) in all of our experiments with the constant acceleration motion model, $\alpha = 0.1$ and $\epsilon = -10^{-3}$. Unless otherwise specified, a sliding window of size 5 is used. The procedure *NonRecursiveFalseHypothesisReplacement*() is used instead of its recursive counterpart. The results are compared with the self initializing version of the GOA tracker with the smooth motion model as defined in [35]. Since Veenman et al. have shown experimentally [35] that, in most situations, the GOA Tracker outperforms the algorithms in [4], [25], [27], [29], a comparison against the GOA Tracker implies a comparison against all these algorithms, which also includes MHT.

7.1 Results on Synthetic Data

The synthetic sequences in this section are generated by the data set generator called *Point Set Motion Generator (PSMG)* [36]. The generator provides control over the size of image space, number of points, number of frames, mean and variance of initial velocity, mean and variance of change in velocity, probability of occlusion, maximum absence, etc. For every experiment, we consider the following three scenarios separately: 1) Points are not allowed to enter or exit the scene, though they may be occluded or misdetected. 2) Points are allowed to exit the scene but new points may not enter. 3) Points are allowed to enter and exit the scene.

To analyze the performance of tracking and to compare the results, we use *track-based error*, E_T [36], defined as $E_T = 1 - \frac{T_c}{T_t}$, where T_t is the total number of true tracks and T_c is the number of completely correct tracks generated by the tracker. Since the GOA-tracker does not allow the points to enter or exit the scene, the output of GOA is only shown for the first scenario. To analyze the noise handling capability of the algorithms, we consider the scenario when the new points are generated in the middle of the sequence. We use a modified track-based error $E_T^c = 1 - \frac{T_c^c}{T_t^c}$ for both the GOA and the proposed tracker, where T_t^c is the total number of true tracks of points that were visible in both the first and last frame and T_c^c is the number of completely correct such tracks generated by the tracker. The points that enter or exit the image in these sequences are then considered as noise, while only the points that are visible in both the first and last frames are considered as valid tracks. The errors (and running times) reported in this section are estimated by averaging the errors (running times) of 100 sequences, where all the sequences are generated by using the same parameters of the PSMG.

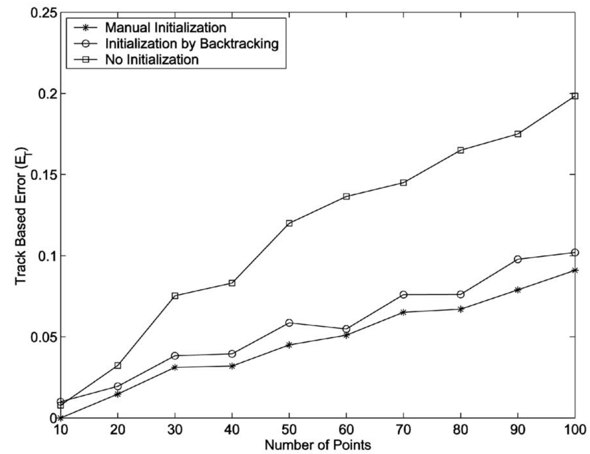


Fig. 8. Track errors with different modes of initialization. The upper curve is obtained by no initialization, while the middle and lower curves are the errors of backtracking and manual initialization, respectively.

7.1.1 Effectiveness of the Proposed Initialization Scheme

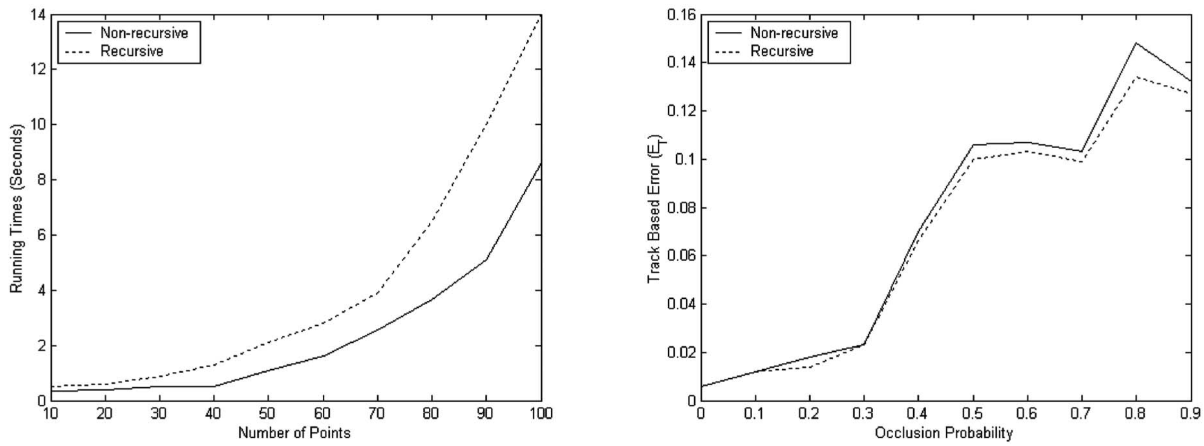
Our first experiment demonstrates the effectiveness of the proposed initialization scheme (i.e., backtracking after the first k frames). The experiments were run for varying numbers of points with three different modes, 1) manual initialization, 2) self-initialization by backtracking, and 3) no initialization. The track errors are shown in Fig. 8. The results show that the proposed initialization scheme is almost as good as the manual initialization and improves the results significantly as compared to no initialization.

7.1.2 Recursive versus Nonrecursive False Hypotheses Replacement

The second experiment analyzes the trade off between the use of recursive and nonrecursive routines for false hypotheses replacement. First, we show the running times of the algorithm with both recursive and nonrecursive schemes (Fig. 9a). The running times are computed over sequences generated by varying the number of points in the scene. Each sequence consists of 100 frames and the running times are reported for the whole sequence. In Fig. 9b, we show the track errors obtained by varying the probability of occlusion per frame (of each point). The number of points in the scene is kept constant over all these sequences. It can be seen that there is no significant difference between the performance of nonrecursive and recursive schemes in terms of track errors. As expected, the nonrecursive scheme is more efficient than the recursive one. For the rest of the experiments, we use the nonrecursive routine for false hypotheses replacement.

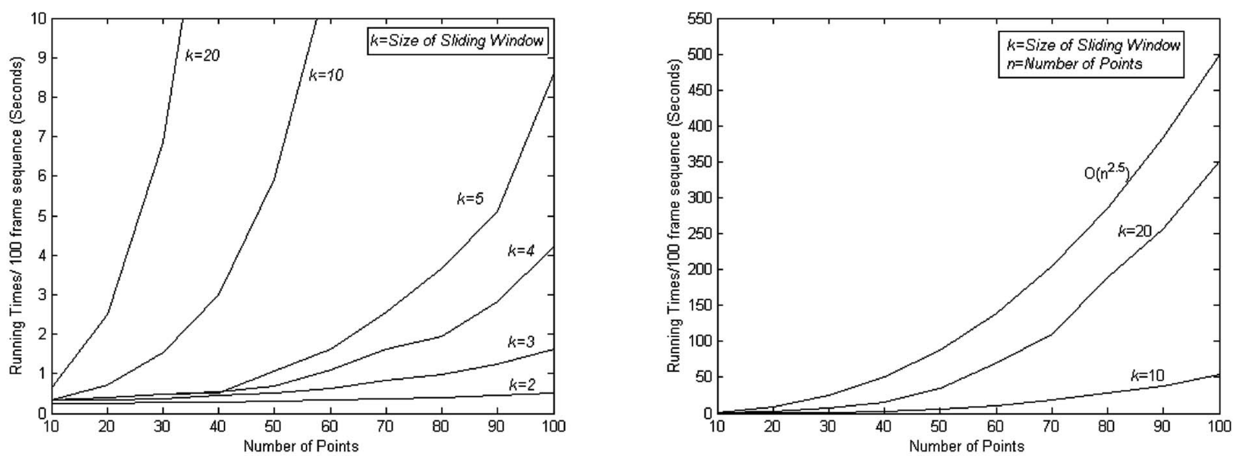
7.1.3 Computational Efficiency

The computational complexity of the algorithm is determined by the number of times the maximum matching of a graph is computed, which is an $O(n^{2.5})$ operation for a graph of order n. In the nonrecursive scheme with a sliding window of size k, this is done at most k times. Hence, the computational



(a) (b)

Fig. 9. The trade off between recursive and nonrecursive schemes for false hypotheses replacement: (a) Running times over varying number of points per frame. (b) Track-based errors E_T over varying occlusion probability.



(a) (b)

Fig. 10. Running times in seconds for 100-frame sequences with varying number of points per frames and sizes of the sliding window.

complexity of the proposed algorithm is bounded by $O(n^{2.5})$, where n is the number of points in the scene. We now verify the above claim by empirical analysis of the running times. As mentioned above, the computational complexity of the algorithm depends on two parameters, 1) the number of points in the scene and 2) the size of the sliding window. Fig. 10a shows the running times of the proposed algorithm with respect to both of these parameters. Fig. 10b shows the plots of running times with sliding windows of size 10 and 20 frames, respectively. It also shows a plot of $O(n^{2.5})$ (with proportionality constant 0.005), where n is the number of points in the scene. From the graph, it is evident that the running times are polynomial and are bounded by $O(n^{2.5})$. The proportionality constant is much lower for smaller sizes of the sliding window. Typically, the algorithm runs at 17 frames per second for 50 points with a sliding window of size 10 on a 2.4GHz Intel Pentium 4 CPU.

7.1.4 Size of the Sliding Window

Recall from Section 4 that the sliding window of size more than $k-1$ is required to handle the occlusion lasting for $k-2$ frames. Here, we validate this claim and also show that this size is sufficient even when the probability of occlusion (or misdetection) of each point in the sequence is very high. In Figs. 11a and 11b, we show the track-based errors E_T for the sequences of varying occlusion probabilities over different sizes of sliding windows. The maximum absence of an occluded point in the sequences of Figs. 11a and 11b is bounded by three frames and six frames, respectively. Notice that the track-based errors do not change significantly by increasing the size of sliding window beyond five frames and eight frames, respectively. Fig. 12 shows the same effect for the sequences with fixed occlusion probability and different number of points in the scene.

For the rest of the experiments, we fix the maximum absence of an occluding point to three frames and the window size to five frames.

7.1.5 Experiments with Variable Point Density, Velocity, and Occlusion Probability

In the next experiment, we analyze the performance of the proposed multiframe algorithm (MF) with respect to the point density. The experiments were performed by

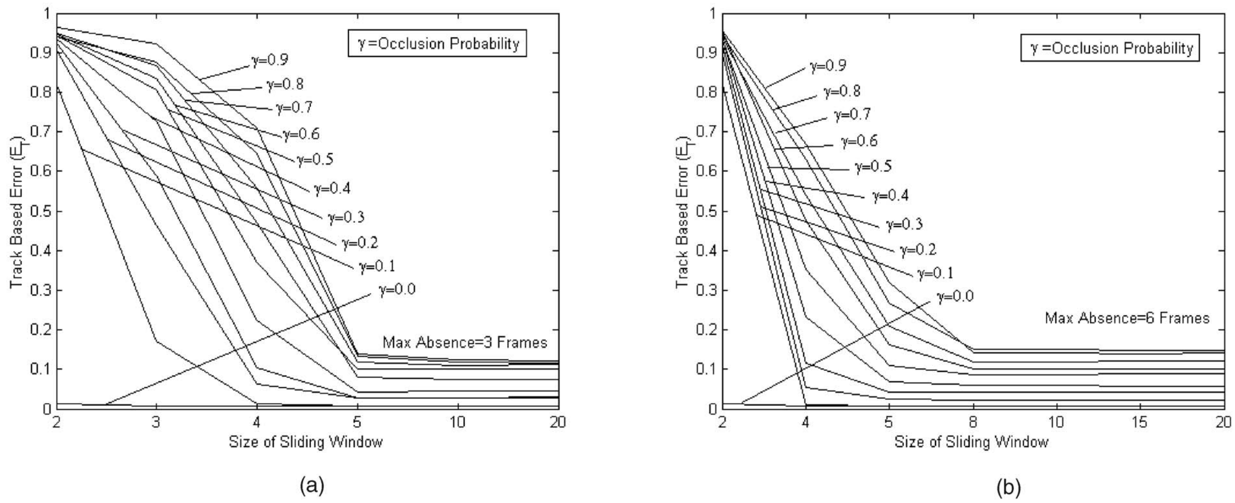


Fig. 11. Performance with respect to the size of sliding window: Track-based error E_T for different occlusion probabilities when the maximum absence of a point is bounded by (a) three frames and (b) six frames.

increasing the number of point tracks in a fixed image space. In Fig. 13a, the track-based errors, E_T, are shown for applying MF on three different types of sequences as described above. In addition, the track-based errors of the GOA tracker are also shown on the sequences where points are not allowed to enter or exit the scene. In Fig. 13b, we show the effect of noise on both trackers by using the modified track-based error, E_T^c, and allowing the points to enter and exit the scene.

Similar experiments were performed on occlusion handling (Fig. 14) and variable velocity performance (Fig. 15), where the probability of occlusion was varied in the former and the mean velocity was increased in the latter.

These results show that the proposed algorithm performs comparably to the GOA tracker when the points are not allowed to enter or exit the scene. However, the performance of the proposed tracker is unaltered when the points are allowed to exit the scene or when additional noise is introduced. The proposed tracker also performs reasonably well on sequences where points are allowed to enter and exit the scene simultaneously given the higher degree of ambiguity in such sequences. In addition, the results clearly show that the proposed algorithm outperforms the GOA tracker in the presence of noise.

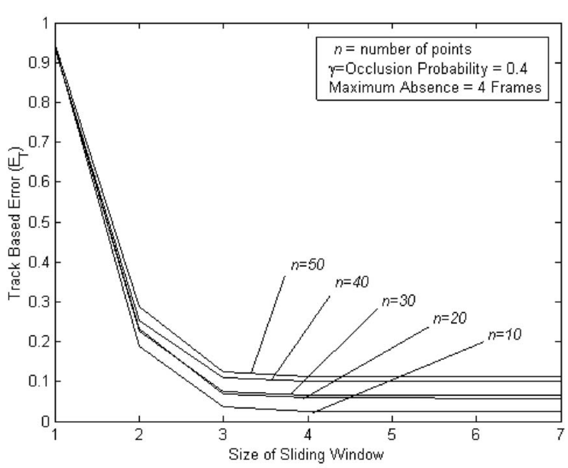


Fig. 12. Performance with respect to the size of sliding window: Track based error E_T for different number of points per frame with 0.4 occlusion probability and maximum absence bounded by four frames.

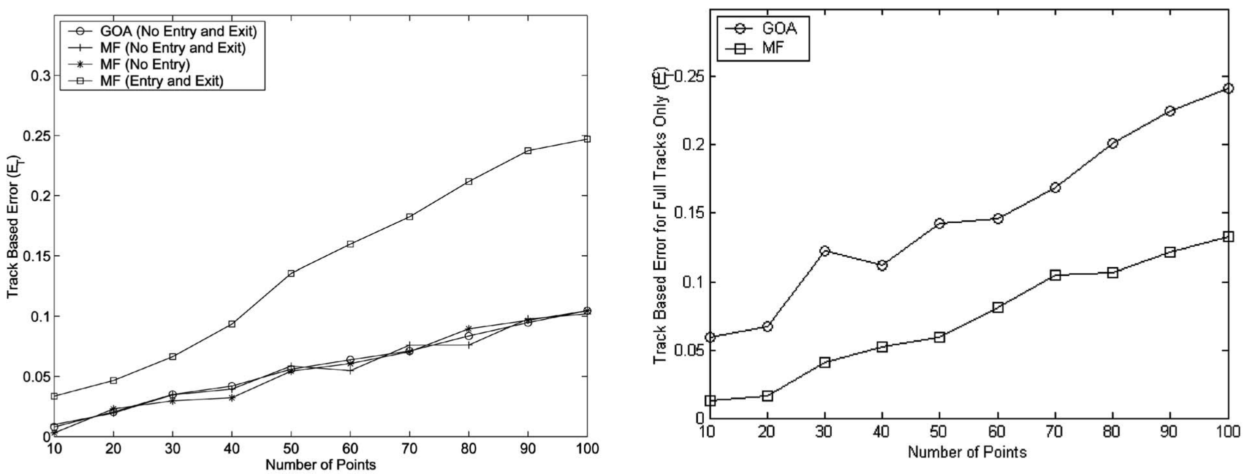
7.1.6 Experiments with Noise Density

To further analyze the noise handling capability of the proposed algorithm, we add random noise (up to 50 percent of the number of points in the scene) at every frame of the sequence. That is, if there are 50 points in the scene, then up to 25 random points per frame are added to the sequence. The tracking errors for different point and noise densities are shown in Fig. 16a. It can be seen from the results that, even with a large number of noisy points, more than 90 percent of the tracks are recovered correctly in most instances.

The presence of noise along with the high probability of occlusion presents a formidable challenge for any point correspondence algorithm. With a high probability of occlusion, any given point is very likely to be missing from a given frame (although we assume that, unless the point has left the scene, it is expected to reappear for at least one frame within a bounded interval of time, after which it may again disappear with the same probability as before). In this case, if the noise density is high, then every frame is likely to have many more spurious measurements than the valid points. The performance of the proposed algorithm with varying occlusion probabilities and noise density is presented in Fig. 16b.

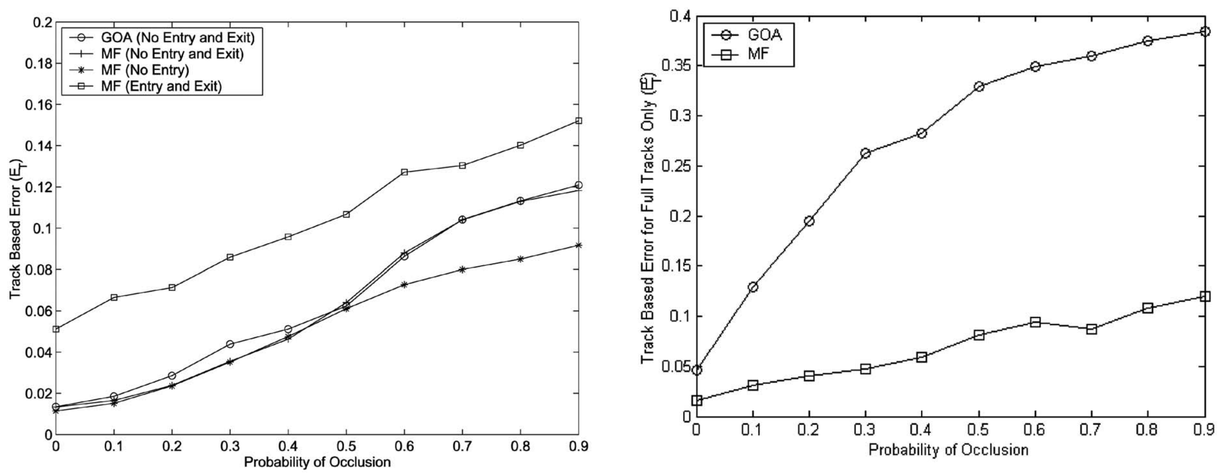
7.2 Results on Real Sequences

Next, we show the results of the proposed algorithm on real data sets. Our first set of experiments is based on the standard sequences in the point correspondence literature. In the first experiment, we use the sequence from [35], where 80 black seeds are placed on a rotating dish. This is an interesting sequence because of the high variance of speed among the points (from circumference to center of circle). Also note that the proposed algorithm does not make use of the synchronous motion of the points. Fig. 17 shows that all of the 80 seeds were correctly tracked over the sequence (this claim is also verified from the ground truth).



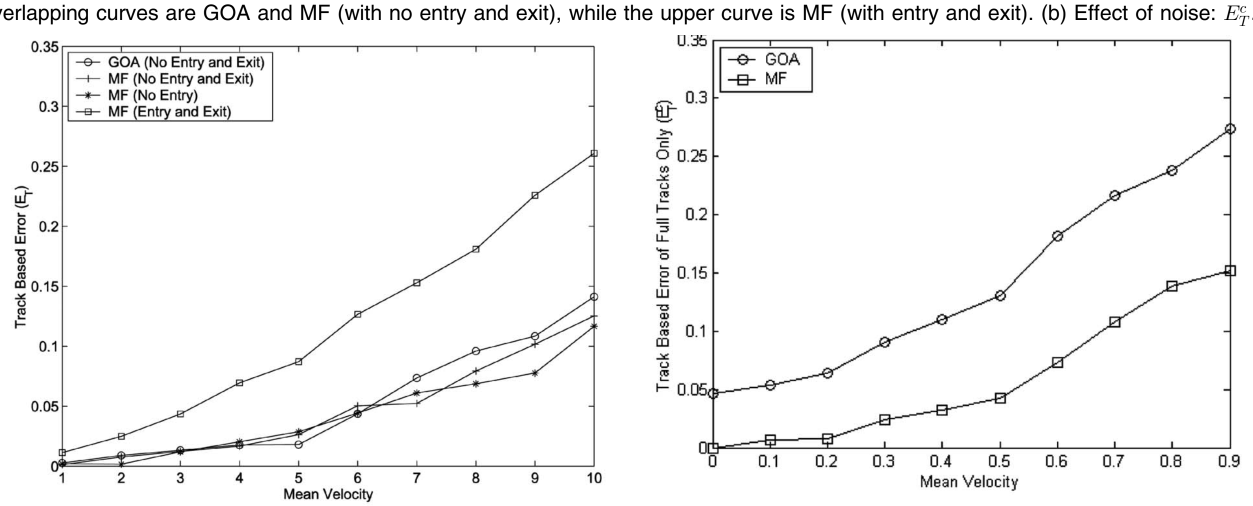
(a) (b)

Fig. 13. Variable point density performance: (a) E_T of proposed Multiframe algorithm (MF) and GOA algorithm. The lower three curves are the errors of GOA (with no entry and exit) and MF (with no entry and exit, and no exit), while the upper curve is the error of MF (with both entry and exit). (b) Effect of noise: E_T^c, when points are allowed to enter and exit the scene. The upper curve is the error of GOA and the lower curve is the error of MF.



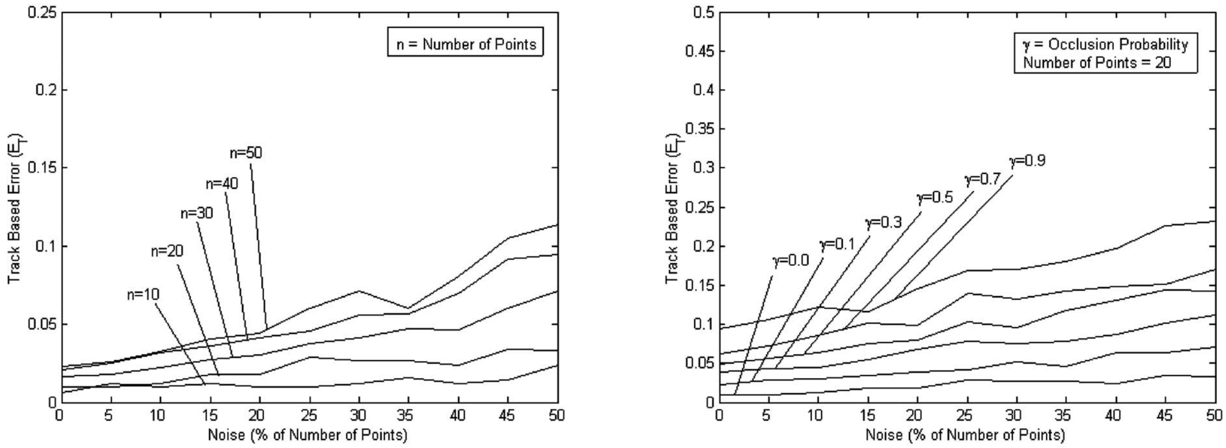
(a) (b)

Fig. 14. Occlusion handling: (a) E_T of proposed Multiframe algorithm (MF) and GOA algorithm. The lowermost curve is MF (with no entry), the middle two overlapping curves are GOA and MF (with no entry and exit), while the upper curve is MF (with entry and exit). (b) Effect of noise: E_T^c.



(a) (b)

Fig. 15. Variable velocity performance: (a) E_T of proposed Multiframe algorithm (MF) and GOA algorithm. The curves show the same pattern as in the previous two experiments. (b) Effect of noise: E_T^c.



(a) (b)

Fig. 16. Performance with respect to random noise density: Track-based errors (E_T) for (a) for varying point densities and (b) for varying occlusion probabilities.

In the next two standard real sequences, we used the KLT method [31] to only detect the feature points, then used the proposed algorithm to establish correspondences. The rotating golf ball sequence in Fig. 18a contains roughly 180 feature points per frame (which also enter and exit the scene in addition to noisy detection by KLT method) and the house sequence (where the camera is rotating about the house) in Fig. 18b contains about 100 feature points per frame. The visual analysis of both outputs show that most of the tracks were perfectly tracked throughout the sequence.

Our second set of experiments is based on natural sequences with a large number of feature points and high occlusion scenarios. The moving objects are detected by background subtraction and their centroids are used as the feature points for tracking. Our first example in this set is from particle tracking (Fig. 19a). It is a 10 frame sequence showing particles in a cylindrical reservoir containing liquid and a tubular heater which drives counterclockwise rotating convection cells. There are more than 100 particles in each frame (some of them are almost stationary, while others appear for one or two frames only). In Fig. 19b, we show some of the tracks of a flock of more than 150 fish in the sea (Fig. 20).[1] The next two examples (Fig. 21) show the tracking results for bird flocks (Figs. 22 and 23). The birds are at different altitudes and there are frequent occlusions. Some discontinuities in tracks at the corners of images (right edge of Fig. 21b) can be observed which are largely due to simultaneous entrances and exits of points at about the same location in the image. Other than that, the objects are tracked reasonably well throughout the sequences. The visual analysis of the results also provides a general feel of the collective motion, although the proposed algorithm does not model the group motion or neighborhood coherency in the scene and each point is tracked independently of the other points.

Fig. 17. Tracks generated for rotating dish sequence.

8 Conclusion

We have presented a framework for the efficient and robust solution of the multiframe point correspondence problem. The proposed framework provides an optimization algorithm that optimizes the gain (cost) function over multiple frames and it may be used for a large variety of motion models and cost functions (including statistical-based functions) that satisfy the constraints as posed therein. The presented algorithm is applicable in more general settings and is shown to perform well through extensive experimentation using synthetic data. Results on real data also support the experimental evaluation.

Acknowledgments

The authors would like to express their appreciation to the anonymous referees for their careful reading of the manuscript and their suggestions for the improvement of the paper. The authors also acknowledge Dr. Cor Veenman for providing the source code of GAO Tracker and Dr. Dmitry Chetverikov for providing the PSMG point generator. This research was partially funded by a grant from Lockheed Martin Corporation.

1. The complete sequences, along with the tracking results, can be accessed at http://www.cs.ucf.edu/~vision/projects/multiframetracking.

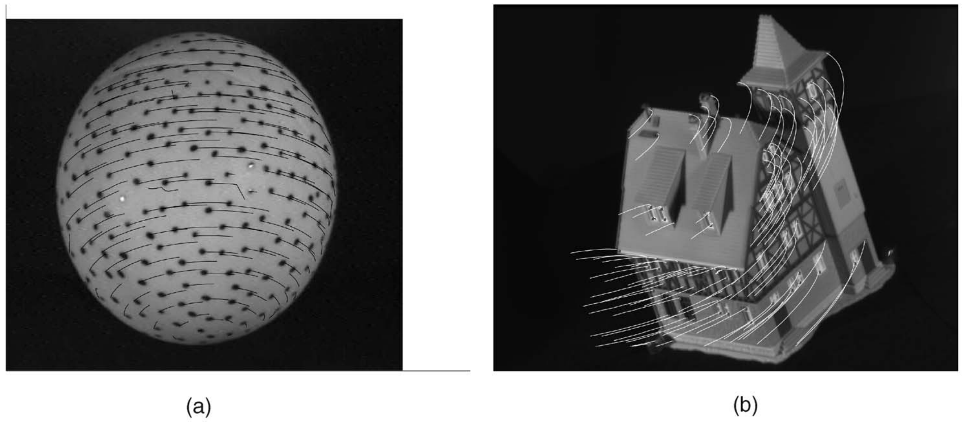


Fig. 18. Tracks generated for (a) rotating ball sequence, (b) house sequence.

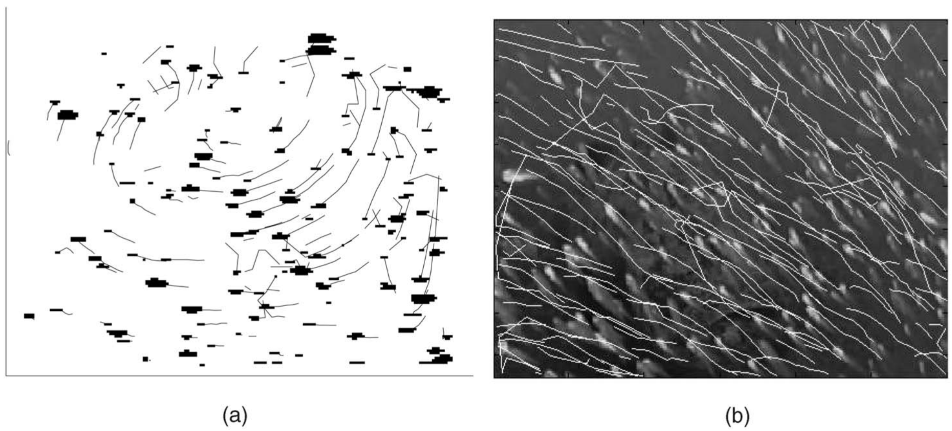


Fig. 19. Frames tracks generated for (a) cylindrical reservoir sequence, (b) flock of fish.

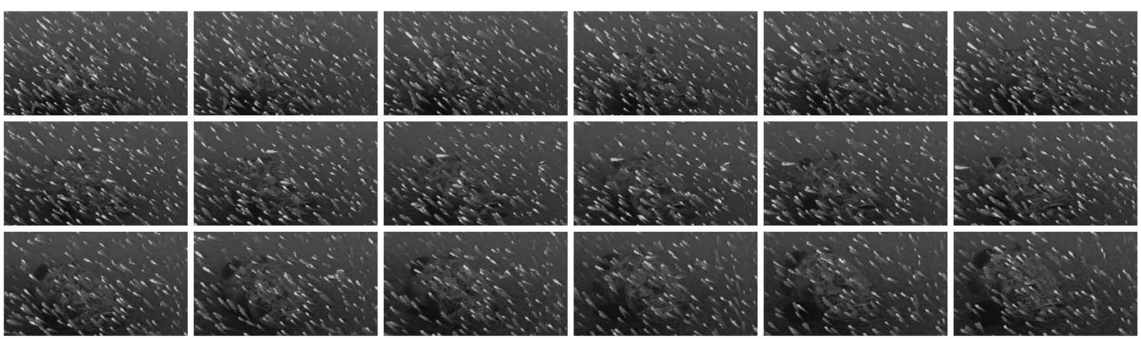


Fig. 20. Frames from fish sequence (every sixth frame).

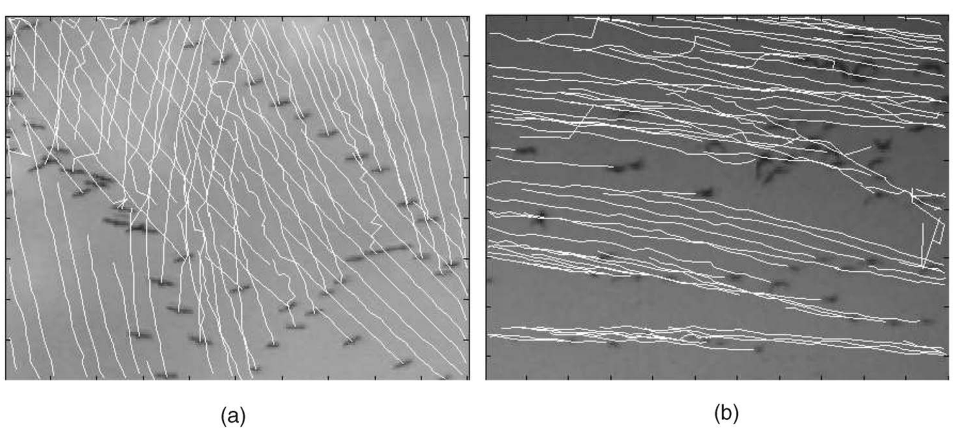


Fig. 21. Tracks generated for two sequences of bird flocks.

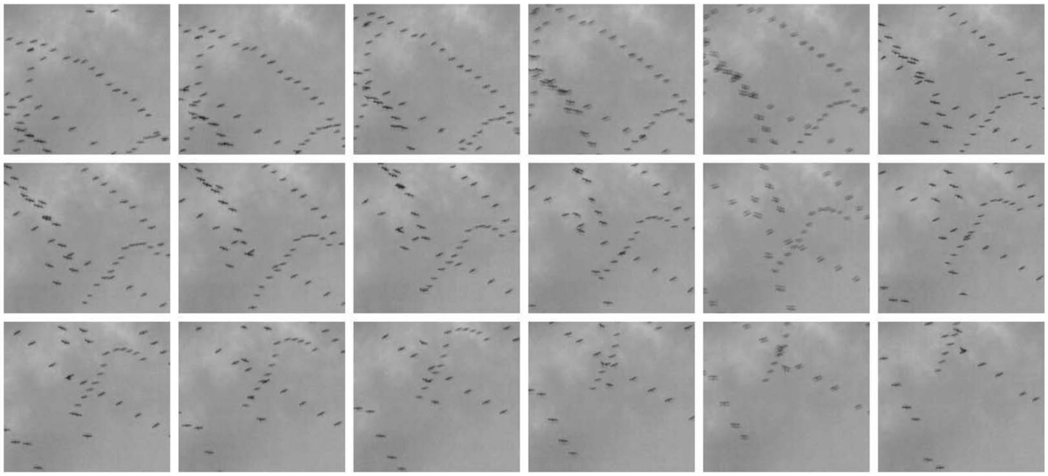


Fig. 22. Frames from first birds sequence (every eighth frame).

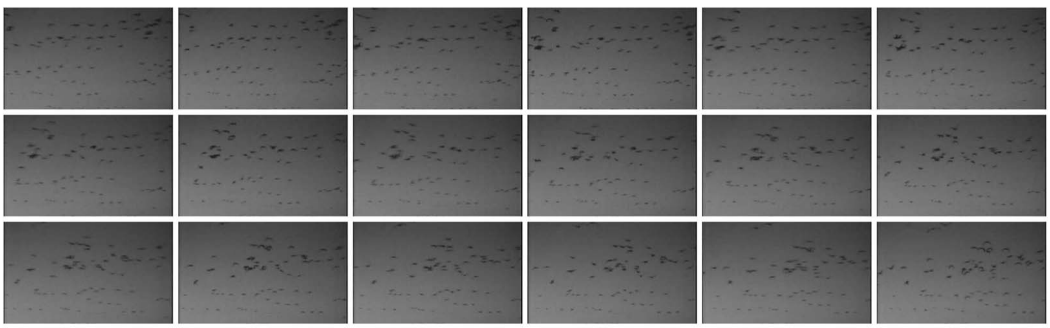


Fig. 23. Frames from second birds sequence (every fifth frame).

REFERENCES

[1] M. Asada and S. Tsuji, "Representation of Three-Dimensional Motion in Dynamic Scenes," *Computer Vision, Graphics, and Image Processing,* vol. 21, no. 1, pp. 118-144, Jan. 1983.

[2] S.T. Barnard and W.B. Thompson, "Disparity Analysis of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 2, no. 4, pp. 333-340, July 1980.

[3] F.T. Boesch and J.F. Gimpel, "Covering the Points of a Digraph with Point Disjoint Paths and Its Application to Code Optimization," *J. ACM,* vol. 24, no. 2, pp. 192-198, Apr. 1977.

[4] D. Chetverikov and J. Verestóy, "Feature Point Tracking for Incomplete Trajectories," *Computing, Devoted Issue on Digital Image Processing,* vol. 62, no. 4, pp. 321-338, 1999.

[5] I.J. Cox, "A Review of Statistical Data Association Techniques for Motion Correspondence," *Int'l J. Computer Vision,* vol. 10, no. 1, pp. 53-66, Feb. 1993.

[6] 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. Pattern Analysis and Machine Intelligence,* vol. 18, no. 2, pp. 138-150, Feb. 1996.

[7] I.J. Cox and M.L. Miller, "On Finding Ranked Assignments with Applications to Multi-Target Tracking and Motion Correspondence," *IEEE Trans. Aerospace and Electronic Systems,* vol. 32, no. 1, pp. 486-489, Jan. 1995.

[8] R. Danchick and G.E. Newnam, "A Fast Method for Finding the Exact N-Best Hypotheses for Multitarget Tracking," *IEEE Trans. Aerospace and Electronic Systems,* vol. 29, no. 2, pp. 555-560, Apr. 1993.

[9] M.R.W. Dawson, "The How and Why of What Went Where in Apparent Motion: Modeling Solutions to the Motion Correspondence Problem," *Psychological Rev.,* vol. 98, pp. 569-603, 1991.

[10] S. Deb, M. Yeddanapudi, K. Pattipati, and Y. Bar-Shalom, "A Generalized S-D Assignment Algorithm for Multisensor-Multitarget State Estimation," *IEEE Trans. Aerospace and Electronic Systems,* vol. 33, no. 2, pp. 523-538, 1997.

[11] T.E. Fortmann, Y. Bar-Shalom, and M. Sheffe, "Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association," *IEEE J. Oceanic Eng.,* vol. 8, no. 3, pp. 173-184, July 1983.

[12] M.R. Gary and D.S. Johnson, *Computers and Intractability.* New York: Freeman, 1979.

[13] J.J. Gibson, *The Ecological Approach to Visual Perception.* Boston: Houghton Mifflen, 1979.

[14] J.E. Hopcroft and R.M. Karp, "An $n^{2.5}$ Algorithm for Maximum Matchings in Bipartite Graphs," *SIAM J. Computing,* vol. 2, no. 4, pp. 225-230, Dec. 1973.

[15] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligence,* vol. 17, pp. 185-203, 1981.

[16] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, "Tracking Across Multiple Cameras with Disjoint Views," *Proc. Ninth IEEE Int'l Conf. Computer Vision,* 2003.

[17] O. Javed and M. Shah, "Tracking and Object Classification for Automated Surveillance," *Proc. European Conf. Computer Vision,* pp. IV: 343 ff., 2002.

[18] M.R.M. Jenkin, "Tracking Three Dimensional Moving Light Displays," *Proc., Workshop Motion: Representation Contr.,* pp. 66-70, 1983.

[19] G. Johansson, "Spatio-Temporal Differentiation and Integration in Visual Motion Perception," *Psychological Research,* vol. 38, pp. 379-393, 1976.

[20] H.W. Kuhn, "The Hungarian method for Solving the Assignment Problem," *Naval Reseach Logistics Quarterly,* vol. 2, pp. 83-97, 1955.

[21] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* New York: W.H. Freeman, 1982.

[22] U. Neisser, *Cognition and Reality: Principles and Implications of Cognitive Psychology.* San Francisco: W.H. Freeman, 1976.

[23] A.B. Poore, "Multidimensional Assignments and Multitarget Tracking," *Proc. Partitioning Data Sets; DIMACS Workshop,* pp. 169-196, 1995.

[24] A.B. Poore and X. Yan, "Data Association in Multi-Frame Processing," *Proc. Second Int'l Conf. Information Fusion,* pp. 1037-1044, 1999.

[25] K. Rangarajan and M. Shah, "Establishing Motion Correspondence," *Computer Vision, Graphics, and Image Processing,* vol. 54, no. 1, pp. 56-73, July 1991.
[26] R.F. Rashid, "Towards a System for the Interpretation of Moving Light Display," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 2, no. 1, pp. 574-581, Jan. 1980.
[27] D.B. Reid, "An Algorithm for Tracking Multiple Targets," *IEEE Trans. Automatic Control,* vol. 24, no. 6, pp. 843-854, Dec. 1979.
[28] V. Salari and I.K. Sethi, "Feature Point Correspondence in the Presence of Occlusion," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, no. 1, pp. 87-91, Jan 1990.
[29] I.K. Sethi and R. Jain, "Finding Trajectories of Feature Points in a Monocular Image Sequence," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 9, no. 1, pp. 56-73, Jan. 1987.
[30] P.J. Shea and A.B. Poore, "Computational Experiences with Hot Starts for a Moving Window Implementation of Track Maintenance," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, no. 1, pp. 87-91, Jan. 1990.
[31] J. Shi and C. Tomasi, "Good Features to Track," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 593-600, 1994.
[32] J.T. Todd, "Visual Information about Rigid and Nonrigid Motion: A Geometric Analysis," *J. Experimental Psychology: Human Perception and Performance,* vol. 8, pp. 238-252, 1982.
[33] R.Y. Tsai and T.S. Huang, "Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, no. 1, pp. 13-27, Jan. 1984.
[34] S. Ullman, *The Interpretation of Visual Motion.* Cambridge, Mass.: MIT Press, 1979.
[35] C.J. Veenman, M.J.T. Reinders, and E. Backer, "Resolving Motion Correspondence for Densely Moving Points," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 1, pp. 54-72, Jan. 2001.
[36] J. Verestóy and D. Chetverikov, "Experimental Comparative Evaluation of Feature Point Tracking Algorithms," *Proc. Workshop Evaluation and Validation of Computer Vision Algorithms,* pp. 183-194, 2000.
[37] D.B. West, *Introduction to Graph Theory.* Prentice Hall, 2001.
[38] M. Yachida, M. Asada, and S. Tsuji, "Automatic Analysis of Moving Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 3, no. 1, pp. 12-20, Jan. 1981.

Khurram Shafique received the BE degree in computer systems engineering from NED University of Engineering and Technology, Pakistan, in 1998 and the MS degree in computer science from the University of Central Florida in 2001. He is currently a PhD candidate at the University of Central Florida and was a recipient of Hillman Fellowship award for excellence in research in computer science in 2003. His research interests include graph theory, discrete optimization, tracking, correspondence across multiple cameras, and real-time surveillance systems. He is a student member of the IEEE.

Mubarak Shah received the BE degree in 1979 in electronics from Dawood College of Engineering and Technology, Karachi, Pakistan and was awarded a five year Quaid-e-Azam (Father of Nation) scholarship for the PhD degree. He spent 1980 at Philips International Institute of Technology, Eindhoven, The Netherlands, where he completed the EDE diploma. He received the MS and PhD degrees, both in computer engineering, from Wayne State University, Detroit, Michigan in 1982 and 1986, respectively. Since 1986, he has been with the University of Central Florida, where he is currently a professor of computer science and the director of the Computer Vision Lab. He is a coauthor of the books *Video Registration* (2003) and *Motion-Based Recognition* (1997), both by Kluwer Academic. He has published more than 150 papers in leading journals and conferences on topics including visual motion, tracking, video registration, and activity and gesture recognition. He has served as a project director for the national site for REU (Research Experience for Undergraduates) in Computer Vision, funded by the US National Science Foundation since 1987. He is a fellow of the IEEE, was an IEEE Distinguished Visitor speaker for 1997-2000. He received the Harris Corporation Engineering Achievement Award in 1999, the TOKTEN awards from UNDP in 1995, 1997, and 2000; Teaching Incentive Program award in 1995 and 2003, Research Incentive Award in 2003, and IEEE Outstanding Engineering Educator Award in 1997. He is an editor of an international book series on video computing, editor-in-chief of *Machine Vision and Applications*, and an associate editor *Pattern Recognition*. He was an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* and a guest editor of the special issue of the *International Journal of Computer Vision* on video computing.