Parallel Distributed Detection of Feature Trajectories in Multiple Discontinuous Motion Image Sequences

Srikanth Thirumalai, Narendra Ahuja

Abstract— **This paper is concerned with three-dimensional interpretation of image sequences showing multiple objects in motion. Each object exhibits smooth motion except at certain time instants when a motion discontinuity may occur. The objects are assumed to contain point features which are detected as the images are acquired. Estimating feature trajectories in the first two frames amounts to feature matching. As more images are acquired, existing trajectories are extended. Both initial detection and extension of trajectories are done by enforcing pertinent constraints from among the following : similarity of image plane arrangement of neighboring features, smoothness of three dimensional motion and smoothness of image plane motion. The constraints are incorporated into energy functions which are minimized using 2-D Hopfield networks. Wrong matches that result from convergence to local minima are eliminated using a 1-D Hopfield like network. Experimental results on several image sequences are shown.**

Keywords— **Feature correspondence, Trajectories, Motion, Structure, Hopfield networks.**

I. Introduction

THIS paper is concerned with three-dimensional interpretation of image sequences showing multiple objects in motion. Each object exhibits smooth motion except at certain time instants when a motion discontinuity may occur. A common type of motion discontinuity is in motion direction, e.g. when an object undergoes a collision. Between such instants of temporal discontinuity each object exhibits a smooth motion.

The objects are assumed to contain point features which are detected as images are acquired. Trajectory detection begins with the first two frames wherein it amounts to the standard problem of feature matching. As more images are acquired, existing trajectories are extended. There is little information available for the initial matching of features in the two frames. So any matching constraints are potentially error prone and must be further confirmed against subsequent frames. Such initial matching of features is done based on the similarity of the image plane arrangements of their neighbors. Clearly, this only holds for those neighbors which are detected in both frames, and provided the neighbors do not belong to another neighboring object. Once the first two frames are matched, the initial segment of each feature trajectory is found. There is now more information available for matching of the features in the second frame with those in the third, i.e., for trajectory extension. The extension of trajectories must be consistent with continuation of the three-dimensional motion of the object. However, this is only true when the object is not undergoing a motion discontinuity. Further, this constraint can be applied only if features have been segmented into objects so that three-dimensional motion of an object can be estimated. When the smoothness of the three-dimensional motion cannot be enforced, trajectories are constrained to have only two-dimensional smoothness which is possible once the initial trajectories are detected, i.e., beyond the second frame, and which is correct except across temporal discontinuities in motion. Thus, several different constraints are used to detect and extend trajectories, but each of these constraints must be used when it is applicable.

As the images are acquired and trajectories are estimated, they are segmented into subsets each corresponding to a different moving object. This is done by identifying neighboring correspondences and breaking these neighbor relationships if they are found to be very dissimilar.

The problem of detecting the feature trajectories is formulated as a series of cost minimization problems that are solved using Hopfield networks. The costs are defined in terms of the constraints mentioned above, such that only appropriate constraints are used at any given image location at any given time. The process of detecting the feature trajectories proceeds in an incremental fashion over time with information from each frame being integrated as it becomes available. At each time step in this process three Hopfield networks are used sequentially, one to compute the geometrical cost associated with every match hypothesis, a second to perform the feature matching and a third to eliminate all the wrong matches. Although the structure of the Hopfield networks (in terms of connectivity) does not change throughout the process of detecting the trajectories, it must be mentioned that some of the weights associated with the networks change with each time step. This is because these weights are calculated from the costs associated with the various match hypotheses.

The advantage of using Hopfield networks is that they offer a simple and elegant framework for optimization problems that can be suitably formulated. In addition, the regular structure of the computation and communication in a Hopfield network allows easy implementation on scalable parallel computers. Tasks involving extensive communication such as updating the network at each step may be

Srikanth Thirumalai is with Cray Research Inc., Eagan, MN. E-mail: Srikanth.Thirumalai@cray.com

Narendra Ahuja is with the Beckman Institute for Advanced Science and Technology, University of Illinois, Urbana, IL. E-mail: ahuja@stereo.csl.uiuc.edu

performed asynchronously like in several iterative linear algebra solvers. It is for these reasons that we have chosen this computational framework for the optimization problem.

In the past there have been attempts by researchers [1], [2], [3], [4] and [5] to use the Hopfield network to solve the feature correspondence problem. There have been no attempts though, to integrate cost measures from several constraints, as discussed above, to solve more complex scenes with multiple moving objects exhibiting temporal discontinuities in their motion. Also, none of the previous approaches offer any scheme to eliminate wrong matches which result from settling down to a local minimum while performing energy minimization. This makes the approach more robust to variations in scene parameters such as inter-frame motion of objects which can vary from 0 to 20 pixels.

Section II gives the details of the formulation of feature matching and trajectory detection problems. Sections III and IV describe the mapping of the problems of feature correspondence and trajectory finding onto the Hopfield network. Section V describes the algorithm to eliminate wrong correspondences that result from descending to a local energy minimum. Section VI describes a method to segment the correspondences into groups representing rigid objects. Finally, section VII contains results on several image sequences.

II. Feature Matching and Trajectory Detection

A. Feature Matching

The problem of feature correspondence deals with finding a match in the current frame, if it exists, for every point in the previous frame. In order to find the correct match, constraints such as uniqueness and image plane similarity in the arrangement of neighbors around the points are imposed. The uniqueness constraint implies that a point in the previous frame can match at most one point in the current frame and vice versa. The other constraint implies that the point in the current frame which is the correct match for a certain point in the previous frame should have a similar geometrical arrangement of neighbors around it. These constraints are then incorporated into an energy function in such a way that the energy function attains a minimum when the points are either correctly matched or not matched at all. This energy minimization is done using the gradient descent approach that is implemented using a Hopfield network.

The first task is to apply a feature detector to a frame when it becomes available. In this discussion, a feature detector is defined as one that extracts features such as intensity maxima, minima and saddle points and sharp corners of objects in the image. Let the first frame have N_1 points and the second frame N_2 points. We have a matrix of $N_1 \times N_2$ possible match hypotheses where element (i, j) indicates that the i^{th} feature in the first frame matches the j^{th} feature in the second frame. Only a subset of these are correct and these represent the correct matches. A cost is associated with every hypothesis such that a low cost is assigned to a correct hypothesis and a high cost is assigned to a wrong hypothesis. Since the motion of the feature points is bounded by a maximum possible motion, we compute the costs for only those hypotheses where the point in the second frame is within a *circle of interest* of the point in the first frame. All other hypotheses are assigned a fixed high cost. The costs are computed based on the image plane similarity in the arrangement of *neighbors* around the two points constituting the hypothesis. The neighbors discussed here are the Delaunay neighbors of the points which may be obtained from the Voronoi diagram [6] of the set of feature points in the image. To determine the cost associated with a match hypothesis, we must try to find a subset of neighbors of the point in the first frame that match a subset of neighbors of the point in the second frame. The similarity in the image plane arrangement of these subsets is used to compute the cost associated with the match. We look for similarity in the subsets rather than the entire set of neighbors because missing feature points and object boundaries cause distortions in the set of neighbors. This cost computation scheme is explained in section IV-A.

The costs associated with all the possible $N_1 \times N_2$ hypotheses are then incorporated into a cost function. Let us say that the process of minimizing the cost function has resulted in N_{12} correct matches. There are $N_2 - N_{12}$ points in the second frame that haven't been matched to points in the first frame. When the third frame becomes available, the first task is to apply the feature detector. Assume that there are N_3 points in the third frame. The correspondences between the N_{12} trajectories and the N_3 points in the third frame have to be established. Doing so results in trajectories of length three. In addition to this there exists the problem of establishing the correspondences between the $N_2 - N_{12}$ points discussed earlier and the N_3 points in the third frame. The latter correspondence process results in trajectories that start off from the second frame. This problem is solved in exactly the same manner as the problem of establishing the point correspondences between the first two frames. The problem of extending the N_{12} trajectories to the third frame is slightly more involved.

B. Trajectory Detection

The trajectory detection problem is just an extension of the feature correspondence problem where the trajectories obtained until the previous frame are to be matched to the points in the current frame if such a match does exist. In addition to constraints such as uniqueness of a match and image plane similarity in the arrangement of neighbors around points corresponding to a correct match, other constraints such as 2-D (image plane) continuity of trajectories and 3-D motion continuity can be imposed.

Consider the problem of extending the N_{12} trajectories obtained thus far to the third frame. The problem can be restated as that of having to match the N_{12} trajectories to the N_3 points in the third frame. In this situation, we have a matrix of $N_{12} \times N_3$ hypotheses. Of these hypotheses, there will be a subset of hypotheses that represent matches between trajectories and points in the third frame that lie

within their circles of interest. The costs to be associated with each of these hypotheses have to be determined. All of the other hypotheses (those involving trajectories and points outside their circles of interest) will be assigned very high costs. The costs associated with each hypothesis are representative of the following three constraints,

2-D arrangement of neighbors: Similarity between the arrangement of neighbors around the last point of the trajectory and the point in the third frame. This is exactly the same as the constraint imposed on point correspondences.

Continuity of 3-D motion: Continuity of the 3–D motion computed from the trajectories already known.

Continuity of trajectories: 2–D continuity of the trajectories.

The extent to which the hypotheses satisfy each of the above constraints is determined separately. This results in three different cost measures that need to be merged before being incorporated into the energy function. The second method is now explained. Having obtained the trajectories until the previous frame, any applicable motion and structure algorithm [7] can be used to estimate the 3-D motion and structure of the objects and these estimates can be used to predict the positions of the feature points in the current frame. Based on the predicted positions, a cost can be associated with every hypothesis that a trajectory extending up to the previous frame matches a point in the current frame. Before applying any motion model one has to segment the trajectories into groups that correspond to rigid objects. This cost computation scheme is described in detail in section IV-B.

The third method to compute the cost to be associated with a trajectory and a point in the third frame that lies in its circle of interest is based on the image plane continuity of the trajectory across the frames. This is done as follows. The time axis is collapsed so that the problem of fitting a function to the trajectory becomes a one-dimensional problem. A Lagrange interpolation is done between the points constituting the trajectory and the slope at the last point of the trajectory is computed. Then, the slope of the line segment joining the last point of the trajectory and the point in the current frame that is competing for the match is computed. The cost associated with the hypothesis is then based on the similarity of the two slopes. This cost computation scheme is described in detail in section IV-C.

We thus have three different cost measures that are associated with each competing hypothesis. A method to merge these three costs and associate a single cost measure with each competing hypothesis is devised.

C. Cost Merge Algorithm

Three cost computation schemes have been outlined above. Each method works well in some cases but fails in others. The 2-D geometrical cost computation method might not yield good results at points close to object boundaries because of differences in the motions of the two objects. It could also happen that though a certain point in the current frame is the correct match for a point in the previous frame, there might be no subset of neighbors that match for the two points. The cost computed using the assumption of 3-D motion continuity across frames yields better results than the 2-D method but again fails at object boundaries because the rigidity assumption is violated across boundaries. This method also fails when there is a temporal discontinuity in the motion of objects. Finally, the cost computation scheme based on the 2–D continuity of trajectories gives good results at object boundaries but fails when there is a temporal discontinuity of motion. In a situation in which there is a temporal discontinuity in the motion, only the cost based on the 2–D arrangement of neighbors can be used. Table I lists the cases in which the three methods fail or apply.

TABLE I

Comparative analysis of the effectiveness of the three cost computation schemes.

Cost computation scheme	Fails	Works
2-D arrangement of neighbors	When no subsets of neighbors match	All other times
Continuity of of 3-D motion	At object boundaries and temporal motion discontinuities	All other times
Continuity of trajectories	Temporal motion discontinuities	All other times

This suggests that there is a need to merge these costs and determine a single cost measure for a certain match that does not fail at either object boundaries or motion discontinuities. One could suggest many methods to achieve this. In this system we have taken the simple approach of choosing the least of the three costs. In our experiments we have seen that this method works pretty well, and there is no need for a more complex cost merge scheme. Once the merged cost associated with every hypothesis is computed, they are incorporated into an energy function along with the uniqueness constraints. This energy function is minimized using the Hopfield network.

III. Mapping the feature correspondence and trajectory finding problems onto the Hopfield Network

In this section we briefly describe how the problems of feature correspondence and trajectory finding are mapped onto the Hopfield network.

A. Mapping the feature matching problem onto the network

Consider the case in which there are two images of the scene taken at two different time instants, t_1 and t_2. The

feature detector is run on the two images and two sets of feature points are obtained. Let the first image have N_1 points and the second image have N_2 points. The problem at hand is to find a subset of points, N_{12} in number, in the first frame that has matches in the second frame.

The Hopfield network [8] used for the above problem consists of a two-dimensional array of processing elements (PEs) having N_1 rows corresponding to the N_1 points in the first frame and N_2 columns corresponding to the N_2 points in the second frame. Each PE is essentially a nonlinear amplifier that produces an output v_i which is related to its input u_i by the equation

$$v_i = g(\lambda u_i) = \frac{1}{2}(1 + \tanh(\lambda u_i)) \tag{1}$$

where λ is called the gain parameter. The input u_i to the i^{th} PE is the weighted sum of the outputs of the PEs that are connected to it. The processing element (i,j) represents the hypothesis that the i^{th} point in frame 1 matches the j^{th} point in frame 2. Each processing element has a potential associated with it. This potential corresponds to the quantity v discussed previously and can take on a continuum of values between 0 and 1. The value 1 represents a sure match between the corresponding points in the two frames and the value 0 represents a nonmatch. Any value between 0 and 1 signifies the level of confidence in the match between the corresponding points.

The connections between the processing elements and the weights associated with them depend on the energy function that has to be minimized. The energy function (EF) used in this problem has the form

$$\begin{aligned} EF &= -\frac{1}{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_1}\sum_{l=1}^{N_2} T_{ij;kl}\, v_{ij}\, v_{kl} - \\ &\quad \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} I_{ij}\, v_{ij} + \\ &\quad \frac{1}{\lambda}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\frac{1}{R_{ij}}\int_0^{v_{ij}} g_i^{-1}(v)dv \end{aligned} \tag{2}$$

where, v_{ij} represents the potential of the $(i,j)^{th}$ PE, $T_{ij;kl}$ represents the weight of the link from the $(k,l)^{th}$ PE to the $(i,j)^{th}$ PE, I_{ij} represents the bias input to the $(i,j)^{th}$ element and R_{ij} represents the resistance seen at the input of the $(i,j)^{th}$ PE. The last term in the above equation is the gain function term.

The energy function (EF) can also be written, specifically for the problem of feature correspondence, as

$$\begin{aligned} EF &= \frac{A}{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1,k\neq j}^{N_2} v_{ij}\, v_{ik} + \\ &\quad \frac{B}{2}\sum_{j=1}^{N_2}\sum_{i=1}^{N_1}\sum_{k=1,k\neq i}^{N_1} v_{ij}\, v_{kj} + \\ &\quad \frac{C}{2}(\sum_{i=1}^{N_1}\sum_{j=1}^{N_2} v_{ij} - N_{12})^2 + \\ &\quad \frac{D}{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1,k\neq j}^{N_2} (Cost(i,j) - Cost(i,k))v_{ij}\, v_{ik} + \\ &\quad \frac{E}{2}\sum_{j=1}^{N_2}\sum_{i=1}^{N_1}\sum_{k=1,k\neq i}^{N_1} (Cost(i,j) - Cost(k,j))v_{ij}\, v_{kj} + \\ &\quad \frac{1}{\lambda}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\frac{1}{R_{ij}}\int_0^{v_{ij}} g_i^{-1}(v)dv \end{aligned} \tag{3}$$

Each term in the above equation has a physical explanation that is outlined below. The first term deals with row inhibition. This term ensures that when the network stabilizes, there is at most one PE in each row that has a potential of 1 whereas all of the other elements have value 0. The constant A determines the relative importance this term is given w.r.t. the other terms. The second term deals with column inhibition. This is the column analog of the first term. When the network stabilizes, at most one PE in each column has a potential of 1. Again the constant B decides the relative importance this term is given w.r.t. the other terms. The first two terms in the above equation enforce the concept of uniqueness of a match, i.e., a point in the first frame can match at most one point in the second frame and vice versa. Since both of these terms are equivalent, we generally give them equal importance, i.e., we have $A = B$. The third term in the energy equation deals with global inhibition. This term is minimum, i.e., 0, only when the total number of $1's$ in the array is N_{12}. This term ensures that there are approximately N_{12} matches obtained when the network stabilizes. At the energy minimum, in most cases, we will not have exactly N_{12} matches but some number that is close to it. In this problem we set N_{12} to $\min(N_1, N_2)$. The fourth term deals with cost based row inhibition. For a certain point in the left frame, all of the points in the second frame compete for a match. If a certain point in the second frame has a lower cost associated with it than another point, then it tries to reduce the potential of the other hypothesis by issuing an inhibitory signal. This reduces the potential of the other PE. In this term again D determines the relative weight that this term has in the final expression. In this discussion, the cost associated with a particular match between point i in the first frame and point j in the second frame (given by $Cost(i,j)$) is obtained by selecting the minimum of the various costs discussed in section II-C. Analogously, the last term deals with cost based column inhibition. Ideally the last two terms should also be 0 when the network stabilizes on a solution.

Comparing Equations (2) and (3) we obtain

$$\begin{aligned} T_{ij;kl} &= -A\delta_{ik}(1-\delta_{jl}) - B\delta_{jl}(1-\delta_{ik}) - C \\ &\quad -D(Cost(i,j) - Cost(k,l))\delta_{ik}(1-\delta_{jl}) \\ &\quad -E(Cost(i,j) - Cost(k,l))\delta_{jl}(1-\delta_{ik}) \end{aligned} \tag{4}$$

where $\delta_{mn} = 1$ if $m = n$ and $\delta_{mn} = 0$ if $m \neq n$. We also obtain

$$I_{ij} = CN_{12} \tag{5}$$

We use the gradient descent method to approach the energy minima. The equation for gradient descent can be written as

$$\frac{du_{ij}}{dt} = -\frac{\partial(Energy)}{\partial v_{ij}} \tag{6}$$

which yields

$$\frac{du_{ij}}{dt} = \sum_{k=1}^{N_1}\sum_{l=1}^{N_2} T_{ij;kl}v_{kl} - \frac{u_{ij}}{\tau_{ij}} + I_{ij} \tag{7}$$

where $\tau_{ij} = R_{ij}C$, the time constant. In our formulation of the problem, we have assumed that the time constant is the same for all processors. This does not affect the solution but only decides the rate of convergence.

A digital simulation of this system requires that we integrate these equations numerically. For a sufficiently small value of Δt, we can write

$$\Delta u_{ij} = (\sum_{k=1}^{N_1}\sum_{l=1}^{N_2} T_{ij;kl}v_{kl} - \frac{u_{ij}}{\tau} + I_{ij})\Delta t \tag{8}$$

The values of u_{ij} can be iteratively updated according to the following rule.

$$u_{ij}(t+1) = u_{ij}(t) + \Delta u_{ij} \tag{9}$$

The final output potential of the PE is given by

$$v_{ij} = g(u_{ij}) = \frac{1}{2}(1 + \tanh \lambda u_{ij}) \tag{10}$$

If we substitute Equations (4) and (5) into (8) we obtain the following result:

$$\begin{aligned}\frac{\Delta u_{ij}}{\Delta t} &= -\frac{u_{ij}}{\tau} - A\sum_{k=1,k\neq j}^{N_2} v_{ik} - B\sum_{k=1,k\neq i}^{N_1} v_{kj} \\ &\quad -C(\sum_{k=1}^{N_1}\sum_{l=1}^{N_2} v_{kl} - N_{12}) \\ &\quad -D\sum_{k=1,k\neq j}^{N_2}(Cost(i,j) - Cost(i,k))v_{ik} \\ &\quad -E\sum_{k=1,k\neq i}^{N_1}(Cost(i,j) - Cost(k,j))v_{kj}\end{aligned} \tag{11}$$

Equations (9) and (11) describe the dynamics of the network. Now only the initial values of the potentials of the PEs (v_{ij} for the $(i,j)^{th}$ PE) have to be specified. Having done this the network could be allowed to evolve in time until it attains a steady state, i.e., a stage where the outputs of the PEs do not change. The output of each PE is initialized to $v_{ij} = 1.0 - Cost(i,j)$. The initial output could then be viewed as the probability that the corresponding hypothesis is true. The network can then be allowed to evolve in time until it stabilizes. The advantages of this initialization scheme are that a fewer number of iteration steps are needed and the solution obtained is better than that obtained using random initialization. After the network stabilizes, all of the PEs have outputs equal to either 0 or 1. Those that have outputs 1 have been identified as the correct hypotheses while those that have outputs 0 have been identified as the wrong hypotheses.

B. Mapping the trajectory detection problem onto the network

The problem of establishing the correspondence between the trajectories computed until the previous frame and the points in the current frame, is similar, in formulation, to the problem of feature correspondence between two frames. The only difference is the method in which the costs associated with every competing hypothesis is calculated. In the case of feature correspondence between two frames, we rely only on the cost based on the image plane similarity in the arrangement of neighbors around the point pairs constituting the hypotheses. In the case of correspondence between the trajectories and points, there are three cost measures based on the image plane similarity in the arrangement of neighbors around the last point of each trajectory and the point in the current frame that constitute each hypothesis, continuity of 3-D motion and continuity of trajectories. The details of the implementation of the three cost computation schemes are explained in section IV.

Before proceeding with the discussion on the computation of the costs associated with the match hypotheses, we discuss the scheme used to select the various constants that determine the weights, A, B, C, D and E, associated with the Hopfield network. The weights A and B are associated with the constraint that a match should be unique. Since this criterion is very important, one must choose large values for both A and B. The parameter C dictates the total number of matches obtained after the Hopfield network stabilizes. Since the total number is not known ahead of time (some points in frame 1 may not have matches in frame 2 and vice-versa), one must not choose a very high value for C since this would impose artificial constraints on the problem. The parameters D and E determine the contribution of 2-D and 3-D motion constraints to the energy function. Since these are subject to problem noise, choosing a very high value for these constants is undesirable. In keeping with these guidelines and after some experiments on real images we have chosen the following values for the parameters - $A = 1.0$, $B = 1.0$, $C = 0.2$, $D = 0.4$ and $E = 0.4$. It was found that these values yield good results for most feature correspondence problems.

IV. Cost Computation

In the previous section, the mapping of the feature correspondence and trajectory finding processes onto the Hopfield network was discussed. The energy function that had to be minimized was a function of the costs associated with every possible hypothesis. In addition, uniqueness constraints were incorporated into the energy function. In this section, the details of the implementation of the three costs are discussed.

A. Two-Dimensional Geometrical Cost Computation

The first step towards computing this cost is to determine the neighbors of the points in both of the frames. This can be done by triangulating the point sets using Delaunay triangulation [6]. For each point in either frame, a set of points that are Delaunay neighbors to it is obtained. To compute the cost associated with a certain match, some method to evaluate the similarity between the sets of neighbors of the two points being considered for the match is necessary.

Consider two points, i in the first frame and j in the second frame. Let the set of points that are neighbors to i be (a, b, c, d, e) and the set of points that are neighbors to j be (a', b', c', d'). To compute the similarity of the 2–D arrangement of neighbors around the points i and j let us consider the sets of lines (ia, ib, ic, id, ie) and (ja', jb', jc', jd'). A subset of lines belonging to the first set that have matches with lines belonging to the second set has to be determined. This is done using another Hopfield network, which is again a two-dimensional network with each PE representing the hypothesis that a certain line in the first set matches another line in the second set. The formulation of this problem is exactly the same as that described above. The problem of computing the cost associated with a match between pairs of lines still remains. This is a simple problem. To compute the cost associated with a match between a pair of lines, first, the similarity in their positions, lengths and orientations is computed. The similarity function used is

$$\begin{aligned} simil \;=\; & F(\, C_1 - C_2\,(shift_{actual} - SHIFT)^2 \\ & -C_3\,(\Delta length_{actual} - \Delta LENGTH) \\ & -C_4\,(\Delta orientation_{actual} \\ & \qquad -\Delta ORIENTATION)\,) \end{aligned} \tag{12}$$

where $F()$ is a nonlinear function, C_1, C_2, C_3 and C_4 are positive constants, $\Delta length_{actual}$ is the absolute value of the actual difference in the lengths of the lines and $\Delta orientation_{actual}$ is the absolute value of the actual difference in the orientations of the lines and $SHIFT$, $\Delta LENGTH$ and $\Delta ORIENTATION$ are constants that have to be provided a priori. From Equation (12) it can be seen that the argument of the similarity function is maximum when $shift_{actual} = SHIFT$, $\Delta length_{actual} = 0$ and $\Delta orientation = 0$, the maximum value of the argument being $C_1 + C_3 \Delta LENGTH + C_4 \Delta ORIENTATION$. The function $F()$ is now defined to be

$$\begin{aligned} F(x) \;&=\; x, && x \geq C_5 \\ &=\; C_5, && x < C_5 \end{aligned} \tag{13}$$

where

$$C_5 = -(C_1 + C_3 \Delta LENGTH + C_4 \Delta ORIENTATION)$$

The values of the parameters $SHIFT = 2$ pixels, $\Delta LENGTH = 6$ pixels, $\Delta ORIENTATION = 6$ degrees, $C_1 = 1.0$, $C_2 = 0.025$, $C_3 = 0.16$, $C_4 = 0.16$ and $C_5 = 3.0$ were selected after several experiments on real images.

The similarity measures for all pairs of lines in the two sets, i.e., for each line match hypothesis are computed. They are then normalized so that they take on values from 0 to 1. The cost associated with each hypothesis is then calculated using the following equation

$$cost = 1.0 - simil_{normalized} \tag{14}$$

Having computed the cost associated with every line match hypothesis, a Hopfield network, similar to the one we described earlier on in section III-A, is used. It is allowed to evolve in time until it stabilizes. Let us say that after stabilizing, the network produced the following solution : line ia matches line ja', line ib matches line jb', and line ic matches line jd', where a, b, c, a', b', d' are the points taken from the example considered at the beginning of this section. This means that points d and e in the first set and point c' in the second set did not have matches. To compute a cost to be associated with the hypothesis that point i in the first frame matches point j in the second frame, the following algorithm is used.

1. For every matched line pair (ia and ja', for e.g.) compute the quantity
   1.0 − dot product between the vectors $\vec{ij}$ and $\vec{aa'}$.
2. Take the minimum of the above quantity over all of the matched pairs. This is the cost.

Once the cost for every point pair is calculated using the above method, the Hopfield network is initialized and allowed to evolve in time according to Equations (9) and (11). When the network stabilizes, all of the PEs that have potential values equal to 1 are those that represent correct matches.

B. Motion Model Based Cost Computation

Let us consider a situation wherein the third frame becomes available to us. At this stage, a set of correspondences between the first two frames is available. Let us assume that there are N_1 points in the first frame, N_2 points in the second frame and N_3 points in the third frame. Using information about the 2-D arrangement of neighbors around the points in the first two frames, a set of N_{12} correspondences can be obtained. These correspond to trajectories of length two. Knowing these correspondences, one could compute the 3-D motion and structure of these points and predict their positions in the third frame. If one wants to use a motion model that assumes rotation and translation of the objects, then one must know the segmentation a priori to be able to compute the motion accurately. We discuss the segmentation technique in section VI. Let us at this point assume that the segmentation of the trajectories until the previous frame is known. To estimate the 3-D motion and structure of objects comprising the image, one can use any of the existing motion and structure algorithms described in [7]. These algorithms formulate the problem of obtaining the motion and structure as a least squares problem and require a minimum number of trajectories of a certain length. The formulation of the motion model based cost computation problem does not change with the kind of motion model applied and hence it is not central to

the feature correspondence problem. In this paper, we use the translation model only because, to a first order, if the sampling in time is dense, the motion of the objects can be approximated by a translational model. Having computed the motion and structure for every segment, we could use that to predict the position of the feature points in the current frame. With every trajectory there is a corresponding point which represents the predicted position of the feature point in the current frame. In order to compute the cost associated with a match between a trajectory and an actual point in the current frame, we could compute the similarity in the arrangement of neighbors around the predicted point associated with the trajectory and the actual point in the third frame. This could be done as explained in section IV-A . As more information becomes available about the segmentation and more trajectories accumulate over more frames, one could use more complex motion models to predict more accurately the positions of the feature points in the current frame.

C. Cost Based on the Continuity of Trajectories

The third method to compute the cost to be associated with a match between a trajectory and the points in the current frame that lie within its circle of interest depends on the the continuity of the trajectory. Consider a trajectory of length $n+1$ extending from the first frame to the $(n+1)^{th}$ frame. Let the image plane coordinates of the points constituting the trajectory be $(X_0, Y_0), (X_1, Y_1), ..., (X_n, Y_n)$. A function has to be fit through these points. While trying to fit a function to these points, one has to consider some specific situations. If the point is stationary, then a function cannot be fit to the point set. It is also important to determine whether the abscissa or the ordinate is to be the independent variable. We use Lagrange interpolation to fit a polynomial of degree three or four to the data because the error with higher degree fits is high. When the trajectory has a length greater than 4, only the last four points of the trajectory are considered. The next step is to compute the slope of the trajectory at the last point of the trajectory. At any time instant, this is the point that belongs to the previous frame. The slope of the lines joining the last point of the trajectory and the points in the current frame that lie within its circle of interest are computed. The cost associated with a match is related to the dot product of the two slope vectors.

In this section, the ways in which the cost associated with various hypotheses are calculated have been discussed. In the case in which the correspondence between trajectories and points in the current frame is sought, there are three different methods to evaluate the costs. As discussed in section II-C, the minimum of the three cost values is selected and associated with each hypothesis.

Once the costs associated with every hypothesis are computed, the output potentials of the PEs of the Hopfield network can be initialized as

$$v_{ij} = 1.0 - Cost(i,j) \tag{15}$$

where the subscript ij denotes the hypothesis that the i^{th} point in the previous frame or the i^{th} trajectory until the previous frame matches with the j^{th} point in the current frame. Once the output potential of every PE is initialized, the network is allowed to evolve according to Equations (9) and (11). The output potential of some of the PEs will grow to 1.0 while that of others will fall to 0.0. The PEs whose output potentials converge to 1.0 are those that represent the correct hypotheses while those that have their output potentials converging to 0.0 represent wrong hypotheses.

V. Eliminating wrong correspondences

Any energy minimization technique suffers from the problem of getting trapped in local minima. In our problem, this results in wrong correspondences. One way to overcome this problem is to use simulated annealing. This is computationally very intensive and cannot guarantee a global minimum in finite time. Another way to overcome this problem is to use heuristic techniques to remove wrong correspondences. We propose another network that is very similar to the Hopfield network described earlier to solve this problem. The algorithm is outlined below.

We assign one correspondence to a processing element that is exactly like the PEs of the Hopfield network. For every correspondence obtained using the correspondence network

1. Compute its Voronoi neighbors. This is done by computing the Voronoi neighbors of the point in the first frame of every correspondence.
2. Determine that neighbor that is most similar to it. The similarity measure used is based on the difference in the lengths and orientations of the correspondences. For the two correspondences, compute $\Delta length$ and $\Delta angle$. Using this calculate

   $$lengthcost = 0.5[1.0 + \tanh(1.5(\Delta length - 2.0))]$$

   $$angcost = 0.5[1.0 + \tanh(0.15(\Delta angle - 20.0))]$$

   Finally, these costs are merged to get one similarity measure by the following equation,

   $$cost = 1.0 - simil = (1.0 - angcost)lengthcost + angcost$$

   This formula has been used because there is evidence [9], [10] to believe that humans consider direction of motion continuity to be very important in all their visual tasks.
3. Establish a connection between the correspondence under consideration and the most similar neighbor with a weight equal to the cost computed in the previous step.
4. Initialize the output of the PE associated with the correspondence to the similarity computed earlier. This is the probability that the correspondence is correct.
5. Simulate every unit according to the equations

   $$\frac{du_i}{dt} = -u_i + I_i + T_{i,j(i)} v_{j(i)}$$

   $$v_i = 0.5[1.0 + \tanh(\lambda u_i)]$$

where $j(i)$ is the unit that is most similar to the i^{th} unit, I is the bias input chosen to be 50 in all our experiments and $T_{i,j(i)} = -100\ cost_{i,j(i)}$, and u_i and v_i are the input and output of the i^{th} PE.

The network is allowed to evolve until the outputs of all the PEs stabilize. Those PEs that approach 1.0 are kept while the others are eliminated. With this reduced set of correspondences, the entire process is repeated until none of the correspondences are removed. This process results in eliminating the wrong correspondences.

VI. Segmentation of trajectories

Once the correspondences between two frames are obtained, they must be grouped into segments representing rigid objects. The segmentation information available for every two frames must then be merged to cover more frames as they become available so as to produce long trajectories belonging to the different rigid objects. This allows more complex motion and structure algorithms to be used to get better estimates of 3–D motion and relative structure. The underlying assumption made in the segmentation procedure proposed in this paper is that the motion between two frames is small due to dense sampling in time of the scene. The segmentation algorithm is outlined below.

First, the correspondence network and the network used to eliminate the wrong trajectories are applied and correct correspondences between two frames are obtained. For every correspondence, its Voronoi neighbors are found. The lines linking every correspondence with its Voronoi neighbors are called edges. There are two types of edges in a scene – inter-object edges and within-the-object edges. For inter-object edges, the correspondences on either end of the edge vary in length and orientation to a large degree, whereas this variation is small for within-the-object edges. This is true because the scene is sampled densely in time. Given this premise, the steps of the algorithm are

1. Each edge is assigned to a processing element with the same input–output characteristics described previously, i.e., $v_i = g(u_i) = 0.5[1.0 + \tanh(\hat{\lambda} u_i)]$ where, u_i and v_i are the input and output of the i^{th} PE and $\hat{\lambda}$ is the gain parameter.
2. Construct an Energy function of the form

$$E = \sum_{i=1}^{\#of edges} \lambda(1.0 - v_i) cost_i + \alpha v_i + \int_0^{v_i} g^{-1}(v) dv$$

that needs to be minimized to give the correct segmentation. The cost function is again calculated like in the trajectory correction algorithm. In this case the two correspondences involved are those on either end of the i^{th} edge. In the above equation, v_i is the probability that the i^{th} edge is an inter-object edge, and λ and α are the constants weighing smoothness of motion versus discontinuities of motion in space. If the $cost_i$ is high, then there is a tendency for v_i to go to 1, whereas if it is low then v_i tends to 0 to reduce the energy.

3. The energy is minimized using the gradient descent approach where

$$\frac{du_i}{dt} = -\frac{\partial E}{\partial v_i} = -u_i + \lambda cost_i - \alpha$$

4. After allowing the network to stabilize, those PEs that have potentials going to 1 are discarded and only the within-the-object edges are retained.

This method fails when the motion between frames is very large or when the motion between neighboring objects is similar. The second case is very difficult to handle, whereas the first case can be handled by trying to find a motion model that fits unions of available segments.

VII. Results

In this section we demonstrate the performance of the algorithms on a variety of images. The examples have been chosen to demonstrate the ability of the system to deal with multiple objects moving in the scene and temporal discontinuity in the motion of objects in the scene. We also demonstrate the performance of the algorithm on some images taken from the set of images recommended for the general use of participants in the IEEE Workshop on Visual Motion 1991.

Fig. 1. First frame of a sequence in which 3 objects have translational motion.

Fig. 1 shows the first frame of a sequence of 10 images of a scene with 3 objects in translational motion. The motion of the 3 objects in this scene was between 4 and 10 pixels per frame. A point feature detector that selected approximately 200 of the strongest intensity maxima and minima was applied on all the images. The feature points detected in the first frame of the sequence with 3 moving objects are indicated using bright or dark points as shown in Fig. 1. Fig. 2 shows all the correct trajectories found between the first and last frames. For every frame, after the trajectories extending up to the previous frame were matched

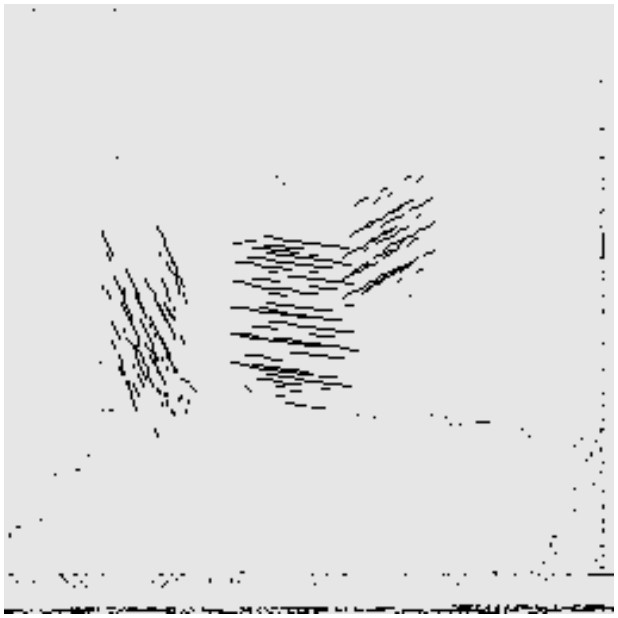

Fig. 2. Correct trajectories detected for the sequence in which three objects have translational motion.

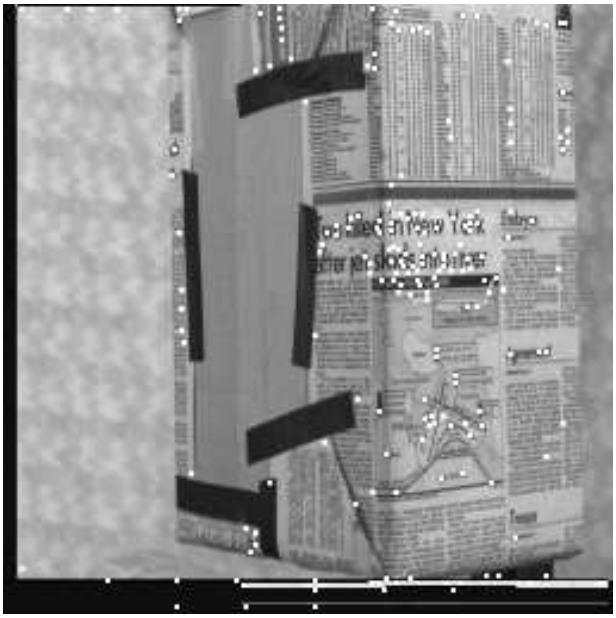

Fig. 4. First frame of a sequence in which the object undergoes a change in the direction of motion.

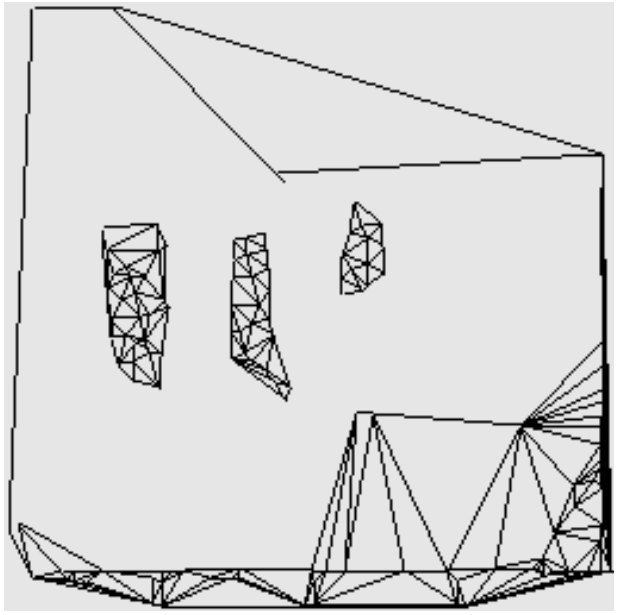

Fig. 3. Segmentation obtained for the sequence in which three objects have translational motion.

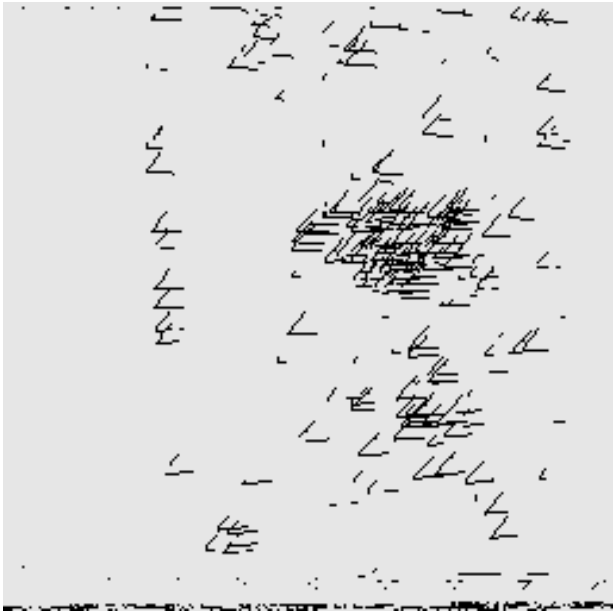

Fig. 5. Correct trajectories detected for the sequence in which the object undergoes a change in the direction of motion.

with the points in the current frame, the wrong extensions were eliminated using the algorithm described in section V. Fig. 3 shows the results of the segmentation algorithm applied to the correspondences between the first two frames of the sequence. There were 140 correspondences detected between the first two frames. It can be seen that these correspondences, indicated by their end points belonging to the first frame, have been grouped into four segments indicating the 3 rigid objects and the background. All the correspondences that belong to one object are connected to their Voronoi neighbors belonging to the same object.

To demonstrate the capability of the system to handle situations where the objects in the scene have temporal motion discontinuities, a sequence of 10 images of a scene with one object was taken. Fig. 4 shows the first frame of the sequence. The object shown in the figure moved to the left for the first 5 frames and then changed direction to move towards the upper right corner of the image. Here again, the motion of the object was of the order of 6 to 8 pixels. In every frame around 200 feature points were selected. The set of correct trajectories obtained between the first and last frames is shown in Fig. 5. The algorithm was able to handle this situation because of the way we define the cost function in this approach. When the object

changes its motion, the continuity of 3-D motion and the 2-D continuity of trajectories are lost and the costs computed using these two methods are high. In this case the algorithm uses the cost computed on the basis of geometrical similarity in the 2-D arrangement of neighbors around the feature points. Between the fifth and the sixth frame (when the object changed its direction of motion) the algorithm was able to detect around 130 correspondences.

Fig. 6. Second frame of a sequence taken by a camera mounted on a rotating robot arm.

The next two sets of experiments demonstrate the feature correspondence algorithm applied to two successive frames of image sequences taken from the set of images recommended for the general use of participants in the IEEE Workshop on Visual Motion 1991. Fig. 6 shows the second frame of one of the sequences. The camera was mounted on on a PUMA robot arm which was made to rotate, causing the entire scene to rotate about the optical axis of the camera. The motion of the feature points depends on their distance from the center of the image. This motion varies from 0 to 18 pixels. The feature detector detected around 190 feature points in the second and third frame of the sequence. The feature correspondence algorithm was used on the the two frames. The algorithm described in section V was used to eliminate the wrong correspondences. Fig. 7 shows the correct correspondences that were determined. In spite of the large variation in feature point motion, the system was able to detect 99 feature correspondences.

Fig. 8 shows the fourth frame of a sequence of 6 frames of an outdoor scene. The camera was made to move along the pathway seen in the figure. In this sequence, the feature points detected had a motion of around 10-15 pixels. Around 190 feature points were used in the fourth and fifth frames to find the correspondences. After eliminating the wrong correspondences, the correct matches are shown in Fig. 9 Again, in spite of the large motion of feature points, 79 matches were detected. The reason for such a low percentage of matches is that the feature detector failed to detect robust features in some areas of the image.

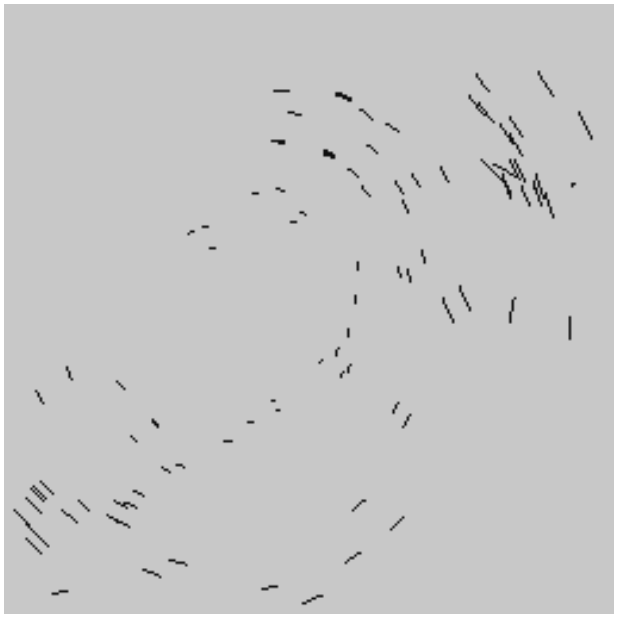

Fig. 7. Correct correspondences detected between the second and third frames of the sequence taken by a camera mounted on a rotating robot arm.

References

[1] A. L. Yuille, "Energy functions for early vision and analog networks", *Biological Cybernetics*, vol. 61, pp. 115–123, 1989.

[2] N. M. Grzywacz and A. L. Yuille, "Motion correspondence and analog networks", *Proceedings of the American Institute of Physics Conference on Neural Network Computing*, vol. 151, pp. 200–205, 1986.

[3] Y. T. Zhou and R. Chellappa, "Neural network algorithms for motion stereo", in *Proceedings of the International Joint Conference on Neural Networks*, 1989, pp. II–251 – II–258.

[4] Y. T. Zhou and R. Chellappa, "A network for motion perception", in *Proceedings of the International Joint Conference on Neural Networks*, 1990, pp. II–875 – II–884.

[5] P. Y. Zhu, T. Kasvand, and A. Krzyzak, "Motion estimation based on point correspondence using neural network", in *Proceedings of the International Joint Conference on Neural Networks*, 1990, pp. II–869 – II–874.

[6] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag.

[7] J. Weng, T. S. Huang, and N. Ahuja, "3-D motion estimation, understanding and prediction from moving image sequences", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 3, pp. 370–389, 1987.

[8] J. J. Hopfield and D. W. Tank, "Neural computation in optimization problems", *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[9] O. J. Braddick, "Low- and high- level processes in apparent motion", *Philosophical Transactions of the Royal Society of London*, vol. B 290, pp. 137–151, 1979.

[10] B. Julesz, *Foundations of Cyclopean Perception*, University of Chicago Press, 1971.

Fig. 8. Fourth frame of an outdoor sequence taken by a camera mounted on a robot traveling along the pathway.

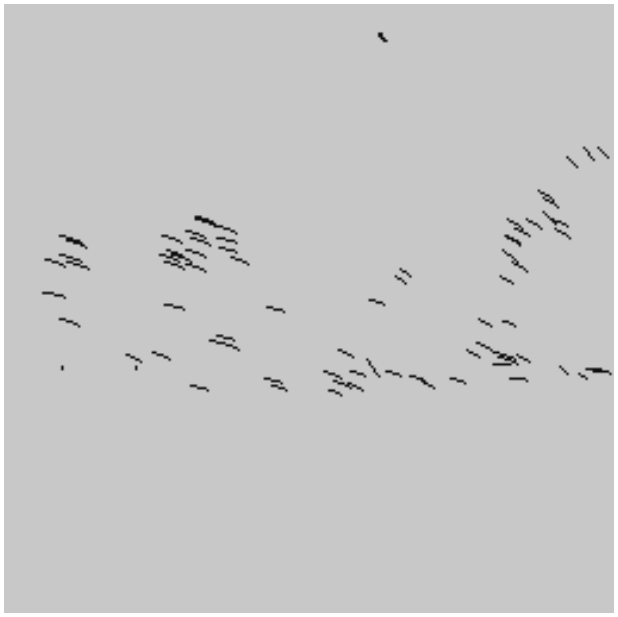

Fig. 9. Correct correspondences detected between the fourth and fifth frames of the outdoor sequence taken by a camera mounted on a robot traveling along the pathway.