COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING 41, 306–322 (1988)

Approximation of Displacement Fields Using Wavefront Region Growing

BIR BHANU* AND WILHELM BURGER†

Department of Computer Science, University of Utah, Salt Lake City, Utah 84112

Received July 10, 1986; accepted August 10, 1987

This paper presents a novel approach for the computation of displacement fields along contours which correspond to moving homogeneous regions in an image sequence. Individual frames of the image sequence are treated one at a time by performing segmentation and 2D motion analysis simultaneously. For the first frame, an original segmentation of the image into disjoint regions is assumed to be given in the form of pixel markings and the properties of these regions. The analysis of each new frame consists of (a) finding the new segmentation and (b) a set of displacement vectors that link corresponding points on the original and the new contour. The new region is assumed to overlap with the original region, such that their intersection is not empty. After finding the intersection, *wavefront region growing* is applied to obtain the new region and to compute a set of tentative displacement vectors. The final approximation is found by using a relaxation-type algorithm which "rotates" the mapping between the original and the new boundary until a correspondence with minimum deformation is found. The proposed algorithm is simple and lends itself to parallel implementation. Various examples are presented to illustrate the approach.

1. INTRODUCTION

Motion analysis is concerned with the reconstruction of an object's 3D motion parameters from a dynamic scene, given a series of two-dimensional projections [1, 9]. From the *apparent motion* of a sufficient number of points on each moving object its actual three-dimensional rotation and translation components can be determined, assuming that the objects involved are rigid. Here we address the problem of how to obtain the apparent motion from a sequence of two-dimensional images.

Two main approaches have been used to compute the optical flow field or displacement field from a given motion sequence of grey-level images. They are commonly referred to as the gradient method and the displacement method.

The *gradient method* [6] uses spatial and temporal grey-level variations to estimate the *instantaneous velocity* at each pixel in the image. It relies on sufficient object texture, continuous motion, and small displacements between subsequent frames. Since the magnitude of flow can only be determined in the direction of the spatial gradient (perpendicular to the tangent of the boundary), the flow vectors cannot be computed locally. Global optimization schemes which smooth the flow field must be applied. However, no scheme for smoothing optical flow fields has been presented yet that gives realistic motion estimates for a wide range of scenes and in the presence of noise. Simple smoothing of the flow field gives rise to problems at flow discontinuities such as object boundaries. Paradoxically, those are the locations where motion estimates should be most easily obtained.

*Present Address: Honeywell Systems and Research Center, 3660 Technology Drive, Minneapolis, MN 55418, USA.

†Present Address: Johannes Kepler University, Institute of Systems Sciences, Linz, Austria.

0734-189X/88 $3.00

The *displacement method* [7, 9] uses the parts of the image, where discontinuities in brightness or motion occur which create problems in the gradient method. Significant features such as line segments or distinguished ("interesting") points in two consecutive frames are selected and matched, thus giving a field of *displacement vectors* for the selected features. Two problems arise during this process: one is the selection and location of significant image features, especially when the images are noisy; the other problem is finding an optimal match between them, commonly referred to as the correspondence problem [3, 10].

We propose a solution which lies between these two methods, one that combines the *implicit matching* process of the gradient method and the *locality* of the displacement method and avoids the correspondence problem. One such approach has been suggested by Hildreth [5], where an approximation of the actual displacement field of moving closed contours is sought in two steps: first the displacement vectors perpendicular to the original contour are determined, and second the resulting flow fields along the contour are smoothed. Experiments indicate that the approximations are close to the motion *perceived* by the human visual system. However, there are two problems which arise in Hildreth's [5] approach.

First, since the perpendicular components of the flow vectors are obtained using the gradient method mentioned above, the resultant vectors will be susceptible to errors in direction as well as in length. Estimating the perpendicular *direction* is done by determining the direction of the edge encountered. This is difficult due to noise and limited spatial resolution. Estimation of *magnitude* of the flow vectors from the brightness gradients is inherently unreliable in cases of missing or too fine texture and/or displacement that exceeds the range of approximately linear gradient.

Second, the final result of this approach gives at best an approximation to the motion that *humans* would perceive, including some forms of illusionary motion. This is an important aspect for understanding human vision, but it is not necessarily the goal of *quantitative* motion analysis useful in machine vision.

The method proposed here is region-based and makes use of the fact that (assuming sufficient sampling in time) corresponding regions in two subsequent frames will overlap, thus giving the initial cue for correspondence. In contrast to finding corresponding boundary segments, the direction of search is implicitly given by the assumption of overlapping regions. The intersection of the old and the new region serves as the *seed* [8] for a *region growing* process, which produces a segmentation for the new image frame. Region growing is done layer-by-layer, very much like a *wavefront* to keep the overall region consistent. By propagating *shortest-distance* information an approximate mapping between the two boundaries is obtained. This mapping between boundary points is smoothed such that the distribution of the endpoints of the displacement vectors on the old and the new region boundary becomes approximately uniform. Subsequently, the mapping is *rotated* until a region-correspondence with *minimum deformation* is found. Experiments show that the final results approximate the actual displacement fields closely, even for extreme displacements.

This approach goes beyond pure motion analysis as it provides a *dynamic segmentation* scheme as well. Each frame is segmented by region-growing, using the previous segmentation as a starting point, while displacement data are computed simultaneously. This is an important benefit in dynamic scene analysis and understanding, where both segmentation *and* motion estimates are essential.

2. APPROACH

2.1. *Outline*

The input assumed to be given is a sequence of digitized images, representing a time-varying scene. The time interval between consecutive images is furthermore assumed to be sufficiently small, such that the condition of overlapping regions is met. This of course depends upon the granularity of the segmentation used and thus also upon the region properties that govern the segmentation process. An initial segmentation is supposed to be available, which is constantly updated while frames are processed successively as part of this *dynamic segmentation* scheme. At this point no attention is paid to the problem of how this initial segmentation is obtained. This could either be accomplished using standard segmentation techniques, or it could become an integral part of the proposed algorithm.

During dynamic segmentation it may occur, that certain regions vanish due to occlusion or when they move out of sight. Similarly new regions are created when objects move into the field of view. The case of occlusion does not pose a problem, since we can assume that the shape of a region will not change dramatically between two frames, and the vanishing of a region is easily detected. Newly created regions can be handled by the same process that provides the initial segmentation. Here we concentrate on the problem, how an established segmentation is *carried over* from one frame to the next while extracting motion data at the same time.

The suggested approach of approximating displacement vector fields consists in the following steps:

(1) Update the given segmentation by *growing* each individual region onto its corresponding region in the next frame.

(2) For every region in the scene compute a set of *displacement vectors* which links the boundary points of the original region to corresponding points on the boundary of this region in the next frame. This again is done in two steps:

(a) Get an initial estimate for the displacement vectors by establishing a *tentative correspondence* between the two contours. Here the *closest neighbors* on the opposing boundaries are selected.

(b) Improve the initial match by searching for an *optimal* correspondence that implies *minimal deformation* of the region between the two instances of time.

The segmentation and the search for the closest neighbors are accomplished simulatenously in one computational step, using a purely local technique. The result is the new segmentation and a relation in the form of pairs of coupled boundary points. The algorithm lends itself naturally to *pipelined* and *VLSI* implementations, making realtime operation feasible. The optimal correspondence is found by *rotating* one of the boundaries until the *minimum deformation* is observed. The sum of differences between corresponding *diameters* is used as the measure for deformation. Details are given in the remainder of this section.

2.2. *Wavefront Region Growing*

This first step of the algorithm operates on a given segmentation for the current frame and finds the new segmentation for the following frame. In addition to that, a

tentative correspondence between the contour points of the two related regions is determined. Both tasks are accomplished simultaneously by a *relaxation*-like algorithm, which scans the image iteratively and adds one new *layer* to every region during each iteration. The growth of the regions as well as the displacement data propagate similar to a *wavefront* during this process.

2.2.1. *Seed*

First the *intersection* between the old and the new region is determined, which is nonempty since we assume that regions overlap in successive frames. Pixels on the intersection are given the symbolic label **new**.

find_intersection:
  given:
    S1 . . . a segmentation into disjoint regions at time t
    I2 . . . the input frame at time $t + 1$
  return:
    S2 . . . a segmentation into disjoint regions at time $t + 1$
  for all image points (**P**) *do*
    if I2(**P**) is consistent with S1(**P**) *then*
      mark S2(**P**) as **new**
      end_if
    end_for

2.2.2. *Region Growing*

The intersection serves as the *seed* to grow onto the new image by acquiring points that are consistent with the properties of the region. Starting with the intersection, one new layer of consistent image points is added to the current region during each iteration. The region-growing process stops when no points could be added during an iteration (Fig. 1). At this point the new segment covers the corresponding image region in the new frame completely. During each iteration the

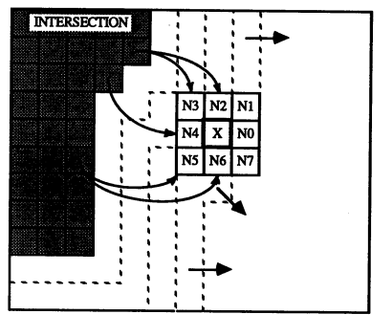

FIG. 1. The shortest distance to the original intersection (shaded area) is determined for every point from the displacements of its 8 neighbors. Assume that the shortest distance between point X and the intersection is to be found. For its neighbors N2 to N6 the shortest distances are already known, while N0, N1, N7 have no displacement values assigned yet. The shortest path from the intersection to X will thus go through N2, N3, . . . , or N6. The computation proceeds in "waves" which are shown as dashed lines.

following update is performed:

```
add_layer:
  for all image points P do
    if P is marked new then
      for all 8 neighbors N do
        if N is consistent with S2(P) then
          mark S2(N) ← S2(P)
          end_if
        end_for
      end_if
    end_for
```

2.2.3. *Displacement Propagation*

During the region-growing process, displacement data are propagated into the newly created parts of the region, such that each point in this region holds information about the location of the closest point in the original intersection (Fig. 1).

The entire image can be viewed as a *connected graph*, where each *node* corresponds to a pixel which is connected to all the neighboring points that are members of the same region. Each node in the graph holds information about the closest point on the original intersection. Those nodes lying on the intersection are initialized as referring to themselves, their displacement from the intersection is zero. The problem can thus be stated as finding the shortest distance from one node (on the intersection) to all other nodes of the graph. This is well known in graph theory as the *shortest path problem* [2] (Dijkstra algorithm). Here the only difference from the classical problem is that, due to the region-growing process, new neighborhoods (and thus links in the graph) are established successively. The graph becomes stable when no further changes in the nodes can be made. As a consequence, the entire image must be scanned and displacement data propagated until all displacements have settled (*relaxed*) to stable values. Although this might appear computationally expensive on a conventional (serial) computer, this technique is well suited for *pipelining* and *VLSI* implementation where high regularity of computation is an important requirement.

```
propagate_displacement:
  for all image points P do
    for all 8 neighbors N of P do
      if disp(N) + d(N, P) < disp(P) then
        disp(P) ← disp(N) + d(N, P)
        end_if
      end_for
    end_for
```

where

disp(**P**) . . . displacement of **P** from the closest point on the intersection (initially set to $+\infty$),

d(**N**, **P**) . . . Euclidean distance between points **N** and **P**.

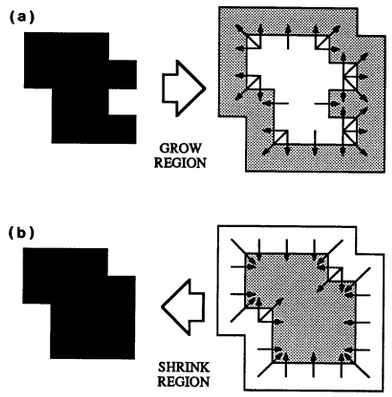

FIG. 2. Region growing and shrinking: (a) growing a region by adding a new layer of pixels; (b) shrinking the region obtained in (a) yields a region which is different from the original region.

2.2.4. *Region Shrinking*

Growing an original region onto the new region in the subsequent frame and propagating displacement information while the growth proceeds is a straightforward approach. The question is how to obtain displacement vectors for those parts of the new boundary which could not be grown, i.e., which lie on the intersection of the original and the new region.

One approach is to *shrink* this part of the original region until the intersection is reached, while propagating displacement data in the same fashion as on the growing side. Whenever a point gets *absorbed* into another point, it would propagate its data to the "absorber." At the end of this process, each point on the boundary of the new region would hold a valid set of displacement vectors from either *shrinking* or *growing*. This originally appealing idea suffers from three drawbacks:

First, shrinking a region is not necessarily *inverse* to growing a region (Fig. 2) which may lead to the creation of pathological intermediate shapes (Fig. 3). *Second*, on the shrinking side the old contour is generally *concave* as seen from the new contour, which results in a strongly nonuniform distribution of terminal points along the original boundary. *Third*, region shrinking does not contribute to the segmentation of the new region, since it only operates in areas which are already marked in the given segmentation and thus no additional information is gained.

Figure 3 shows five steps of the region-forming process, using both region growing and region shrinking. While the region develops quite regularly on the *growing side* (upper right), the region is narrowed down on the *shrinking side* and finally cut off. This behavior is not desirable, since one would expect both sides to develop similarly.

Thus region shrinking is not used in favor of applying only *displacement propagation* in this part of the region, but in backward direction, i.e., starting at the intersection of the original and the new region. This means that in areas where the intersection can grow onto the new region, region growing *and* displacement propagation are done simultaneously. In those areas of the original region which are

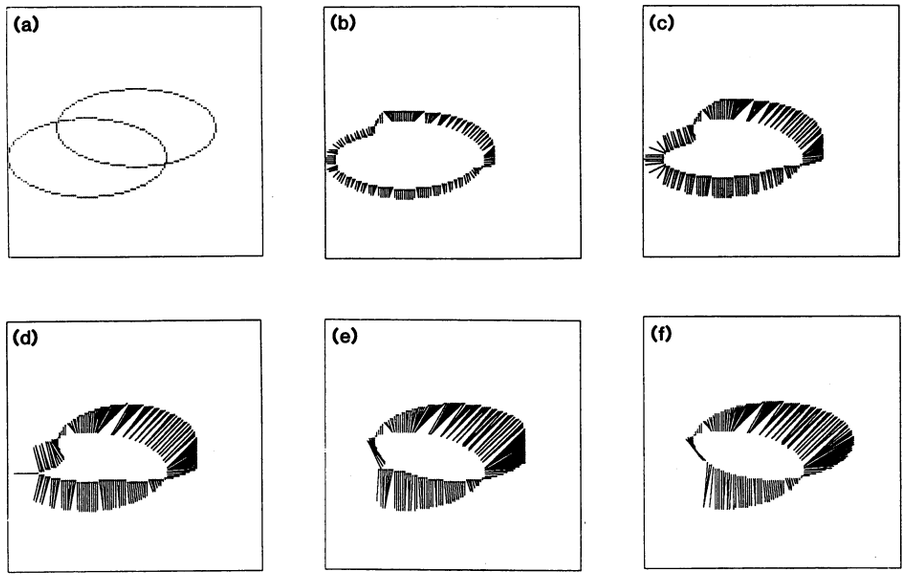

FIG. 3. Applying region growing and shrinking to a moving region: (a) the boundaries of the original region (left) and the new region; (b)–(f) the original region shrinks from the left towards the intersection, while it grows on the right onto the new region. Five layers are removed and added between each pair of ellipse on the shrinking and on the growing side.

not part of the intersection, displacement data are propagated *backwards* from the intersection until the original boundary is reached (Fig. 4).

2.3. *Correspondence Relation*

After the process of region growing and displacement propagation has terminated, the boundary points of the *union* of the old and the new region carry pointers to the closest points on the intersection. From this information a *correspondence relation* **C** is computed, consisting of pairs of boundary points:

Correspondence relation **C**(B1, B2):

$$\mathbf{C} = \{(\mathbf{P}, \mathbf{Q}) | \mathbf{P} = (x_p, y_p) \in \mathrm{B1}, \mathbf{Q} = (x_q, y_q) \in \mathrm{B2}\}, \tag{1}$$

where B1 is the original region boundary and B2 is the new region boundary.

This relation represents a *mapping* of the original boundary onto the new boundary of the region. Notice that one point in the original boundary may have several corresponding points on the new boundary and vice versa, while some points on either boundary are not linked to any other points at all. The finite spatial resolution exaggerates this fact, since corner points on a jagged boundary are likely to be closer to other points than their neighbors. *Smoothing* (see below) is applied to greatly reduce this effect and obtain a more uniform distribution of linked boundary points. Still this *shortest-distance approximation* is "well-behaved," in the sense that the displacement vectors do not cross over, a fact that will be useful in the following section.

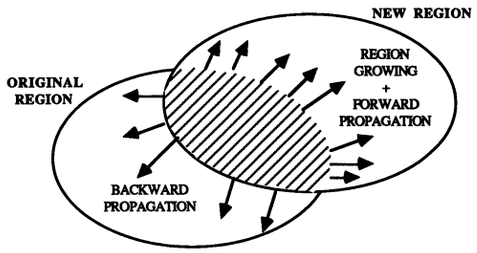

FIG. 4. Propagation of displacement data. From the intersection of the original region and the new region, displacement data are propagated in two directions: *forward*, where the intersection grows onto the new region and *backward* over the original region.

2.3.1. *Optimal Correspondence*

Given the tentative correspondences between the old and the new region boundaries (as described in the previous section), we try to obtain a more realistic set of point-to-point relations. In general, the initial approximation by selecting the *nearest neighbor* on the opposite contour is not a good estimate for the actual displacement vectors. For instance, displacement caused by translation in the direction of the boundary is not detected, because the estimated displacement vectors are zero at these points. This has been termed the *aperture problem* in motion analysis.

A better correspondence relation is obtained by *smoothing* to remove the effects of finite spatial resolution and to distribute matched points more uniformly along the boundaries. The smoothed correspondence is then modified by *rotating* the new boundary in order to find an optimal correspondence. An *error function* is used, which indicates the amount of *deformation* applied to the region under the given boundary mapping.

2.3.2. *Smoothing*

The initial shortest-distance mapping represents a set of displacement vectors, which link points on the original boundary (*origin points*) to points on the new boundary (*terminal points*). Due to the finite spatial resolution, however, *origin points* may have several corresponding *terminal points* and vice versa while other points in the neighborhood may not be matched at all. Ideally, we would like to see the endpoints of the displacement vectors uniformly distributed along both boundaries. The smoothing step tries to remove those clusters of endpoints using a technique similar to histogram equalization [4].

A local smoothing algorithm is applied, which iteratively traverses the correspondence relation and dissolves clusters of boundary points on both sides of the relation. The smoothed relation is available after a few iterations, in fact only *two* iterations were applied in the actual experiments. The following outline of the smoothing algorithm provides separate steps for smoothing each side of the correspondence relation for the sake of clarity. The two boundaries are assumed to be available as two cyclically linked lists of points, such that for each point its successor (succ) and its predecessor (pred) are defined.

```
smooth_correspondence:

  given:
    B1 ... original boundary (linked list of origin points)
    B2 ... new boundary (linked lists of terminal points)
    C(B1, B2) ... Correspondence Relation B1 → B2
    N ... number of tupels in C
    density(P) ...
      function that returns the number of displacement vectors meeting in boundary
      point P.

  repeat
    Smooth-B2 {new boundary}:
      for i from 0 to N − 1 do                    {traverse the relation C clockwise}
        (P_i, Q_i) ← C(i)
        (P_j, Q_j) ← C(i + 1)
        if Q_i ≠ Q_j then                         {terminal points Q_i, Q_j are not identical}
          if succ(Q_i) = Q_j then                 {terminal points Q_i, Q_j are neighbors}
            if density(Q_i) > density(Q_j) + 1 then
              Q_i ← Q_j
            else if density(Q_j) > density(Q_i) + 1 then
              Q_j ← Q_i
              end_if
          else                                    {terminal points P_i, Q_i are not neighbors}
            if density(Q_i) > 1 then              {points can be spread out clockwise}
              Q_i ← succ(Q_i)
              end_if
            if density(Q_j) > 1 then              {points can be spread out counterclockwise}
              Q_j ← pred(Q_i)
              end_if
            end_if
          end_if
        end_for

    Smooth-B1 {original boundary}:                {analogous to smoothing B2}
      for i from 0 to N − 1 do                    {traverse the relation C clockwise}
        (P_i, Q_i) ← C(i)
        (P_j, Q_j) ← C(i + 1)
        if P_i ≠ P_j then                         {origin points P_i, P_j are not identical}
        . . . . . .
      end_for
    until no more boundary points can be moved.
```

The two main steps of the algorithm could be done in parallel. The results of the smoothing algorithm applied to an initial approximation are shown in Fig. 5. The displacement field between the old (outlined) and the new (dashed) boundary is shown for an ellipse undergoing rotation. First the relation is smoothed by spreading points on the new boundary (Fig. 5a) and then the original boundary is smoothed (Fig. 5b).

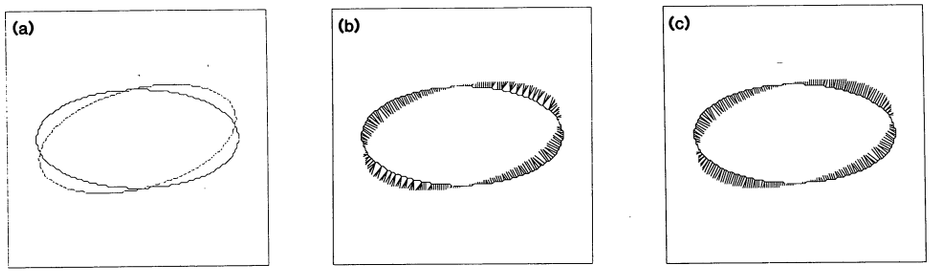

FIG. 5. Smoothing of the correspondence relation in two steps: (a) the boundaries of the original and the new region (dashed); (b) after equalizing the density of endpoints on the new boundary; (c) after equalizing the density of endpoints on both boundaries.

2.3.3. *Error Function*

A measure is required which indicates, how good the initial set of point-to-point correspondences represents the actual displacement vectors. The error function should furthermore be well behaved, e.g., the optimal value should be unique and distinct. Since the actual displacement vectors are *unknown*, we cannot expect to find an objective error function without applying additional restrictions. Previous approaches [5] have used the constraint of "*smoothness of flow*" either as a global restriction or along boundaries. As mentioned earlier, global smoothness of flow cannot cope with motion discontinuities which occur at object boundaries. Also a smooth vector field along a region boundary is not a good approximation in general, such as in the case of pure rotation as illustrated by the experimental results in Section 3.

The error function that we apply quantifies the *amount of deformation* that the region would undergo with a given set of point-to-point correspondences. The correspondence that results in the *least deformation* of the region is chosen as the closest approximation to the real situation. The measure of deformation is based on the differences of *diameters* across the region. If no deformation has occurred, then the distance between one point-pair on the old region should be the same as the distance between the corresponding point-pair on the new region. Pairs are selected such that their points lie approximately *opposite* to each other on the regions' boundaries. For all point-pairs that correspond on the old and the new boundary, the resulting diameters are compared. The sum of the squares of their differences is taken as the error measure.

Given a correspondence relation **C**(B1, B2) containing **N** tupels, the deformation error is defined as

$$E_d(\mathbf{C}) = \sum_{\substack{(\mathbf{P},\mathbf{Q}),(\mathbf{P}',\mathbf{Q}')\in\mathbf{C} \\ opposite(\mathbf{P},\mathbf{P}')}} [d(\mathbf{P},\mathbf{P}') - d(\mathbf{Q},\mathbf{Q}')]^2, \tag{2}$$

where *opposite*$(\mathbf{P},\mathbf{P}')$ means that $\mathbf{P}'$ and $\mathbf{P}$ are separated by approximately half the

circumference of boundary B1:

$$opposite(\mathbf{P},\mathbf{P}') \Leftarrow\Rightarrow \mathbf{P} = \mathbf{C}(i), \qquad \mathbf{P}' = \mathbf{C}(j), \qquad j = \left(i + \frac{M}{2}\right) \bmod (M-1),$$
$$M = |\mathrm{B1}|. \tag{3}$$

The diameter d is defined as

$$d(\mathbf{P},\mathbf{P}') = \left[(x_{p'} - x_p)^2 + (y_{p'} - y_p)^2\right]^{1/2}. \tag{4}$$

To obtain a *quantitative* estimate of the goodness of fit for the selected correspondence, this error measure could be normalized, e.g., by the sum of the diameters. Since here we are only interested in finding the optimal correspondence for each region individually, normalization is not required. Minimal deformation as an indicator for the optimal correspondence will produce satisfactory results as long as there are no dramatic changes in the shape of a region between two frames. This measure will fail when the shape of a region is undergoing dramatic changes, such as in the case of rapid occlusion. Since sufficient sampling in time is assumed, these changes will generally proceed gradually. The final error resulting from the optimal correspondence furthermore indicates whether the deformation is within acceptable bounds.

2.3.4. *Modification Rules*

After defining the criterion to guide the search for an optimal correspondence mapping, we define rules to select candidate mappings out of the many different mappings possible. It turns out that the search space of suitable mappings can be reduced considerably by making use of the implicit order of the set of boundary points. The initial *shortest-distance* approximation has the property, that displacement vectors do *not* crossover, which means that the order of pairs of points on boundary (for instance, clockwise) will be the same for the corresponding points on the other boundary. We term this property of a mapping between two closed boundaries as *radial*:

Given:

> B1, B2, two ordered sets of boundary points representing two closed boundaries. A mapping **C**: B1 → B2 is called *radial*, iff
>
> $$\text{for all } (P_i, Q_i), (P_j, Q_j), (P_k, Q_k) \in \mathbf{C}:$$
> $$ordered_{\mathrm{B1}}(P_i, P_j, P_k) \Rightarrow ordered_{\mathrm{B2}}(Q_i, Q_j, Q_k), \tag{5}$$

where $ordered_{\mathrm{B}}(P_i, P_j, P_k)$ means, that points P_i, P_j, P_k lie on boundary B in (clockwise) order.

This condition must hold for the optimal mapping as well, so we never need to investigate *permutations* of the initial mapping. Among all the other remaining possible mappings we select those that can be found by a *cyclic shift*. This means that the maximum number of mappings to be considered equals the number of boundary points. The optimization problem can thus be stated as:

Given:

B1, B2, two ordered sets of boundary points with approximately equal number of elements

$\mathbf{C}_0$, an initial *radial* mapping B1 → B2.

Find a *radial* mapping $\mathbf{C}_{opt}$: B1 → B2, such that

$$E_d(\mathbf{C}_{opt}) \text{ is a minimum for all } \textit{radial} \text{ mappings } \mathbf{C}'\text{: B1} \rightarrow \text{B2}.$$

Therefore, we only have to *rotate* the correspondence relation until the mapping of minimal deformation is found. In practice it is sufficient to restrict the amount of cyclic shifting to a limited neighborhood of the original estimate. A shift of $\pm \frac{1}{4}$ the length of the boundary (as used in the experiments) will include the optimal solution in most practical cases. For each cyclic displacement the deformation error of the corresponding mapping is evaluated. From all the inspected mappings the one that results in the minimal deformation is selected. The associated set of displacement vectors is taken as an estimate for the actual displacement field. Results of this algorithm obtained from simple (elliptical) moving regions are given in the following section.

3. RESULTS

The selection of the type of objects and motion used was influenced by the work of other researchers in the motion analysis community. All experiments were conducted with ellipses undergoing translation and rotation in 2D space, which allows a direct comparison with the results obtained by Hildreth [5].

3.1. *Experiments*

Programs were written in C and run on a VAX 11/785 under the UNIX operating system. The frames shown have a resolution of 128 × 128 pixels and the major axis of the elliptic objects is 80 pixels long. A *Grinnell* color display system was used to observe the region growing process visually. The results were printed on paper using the *plot*-library in C. Four different types of 2D motion were investigated on elliptic objects:

(1) *translation* only (Figs. 6a–g),

(2) *rotation* only (Figs. 7a–g),

(3) *translation and rotation* (Figs. 8a–f), and

(4) *extreme rotation* (Figs. 9a–f).

For experiments (1) and (2), the corresponding results from Hildreth [5] are included.

3.2. *Discussion*

Experiment 1 (Fig. 6). In this case of *pure translation*, the smoothed approximation is identical to the correspondence of minimum deformation, i.e., rotation of the new boundary does not improve the result in this particular case. One reason is that the terminal points are not spread uniformly over the old and the new boundaries by the local smoothing algorithm. Increasing the local support of the smoothing

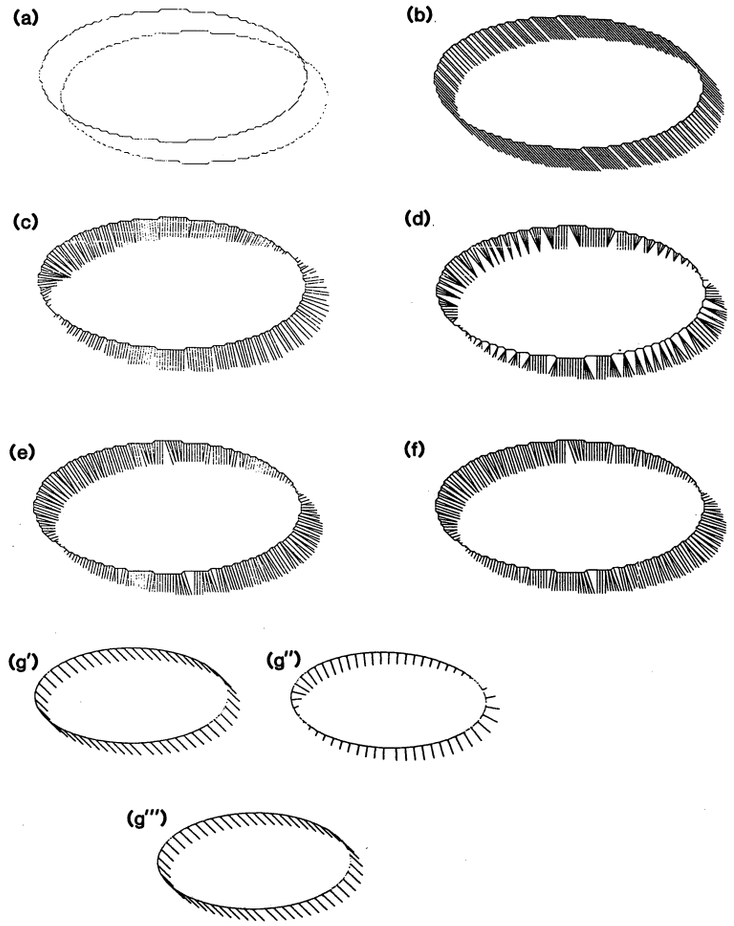

FIG. 6. Pure translation: (a) the boundaries of the original and the new region (dashed); (b) actual displacement vectors; (c) perpendicular components of displacement vectors; (d) shortest-distance approximation; (e) smoothed approximation; (f) final approximation; (g) comparable results from Hildreth: (g′) actual displacement vectors; (g″) perpendicular components; (g‴) smoothed displacement field.

algorithm to obtain a more uniform distribution of points along the boundaries would remedy this problem. Better approximations for pure translation can also be expected for regions which are not symmetric and do not have extremely smooth boundaries. Compared to the corresponding result from Hildreth [5, p. 50] (Fig. 6g) the approximation shown in Fig. 6f is inferior. This is no surprise, since in the case of pure translation the smoothest displacement field (i.e., with uniform orientation) obtained by Hildreth is *always* the correct solution. However, extremely uniform orientation of the displacement field is not optimal for other forms of motions and the type of motion is generally not known a priori.

Experiment 2 (Fig. 7). Here *pure rotation* of 15° about the center of the ellipse was applied. The initial approximation differs significantly from the actual displace-

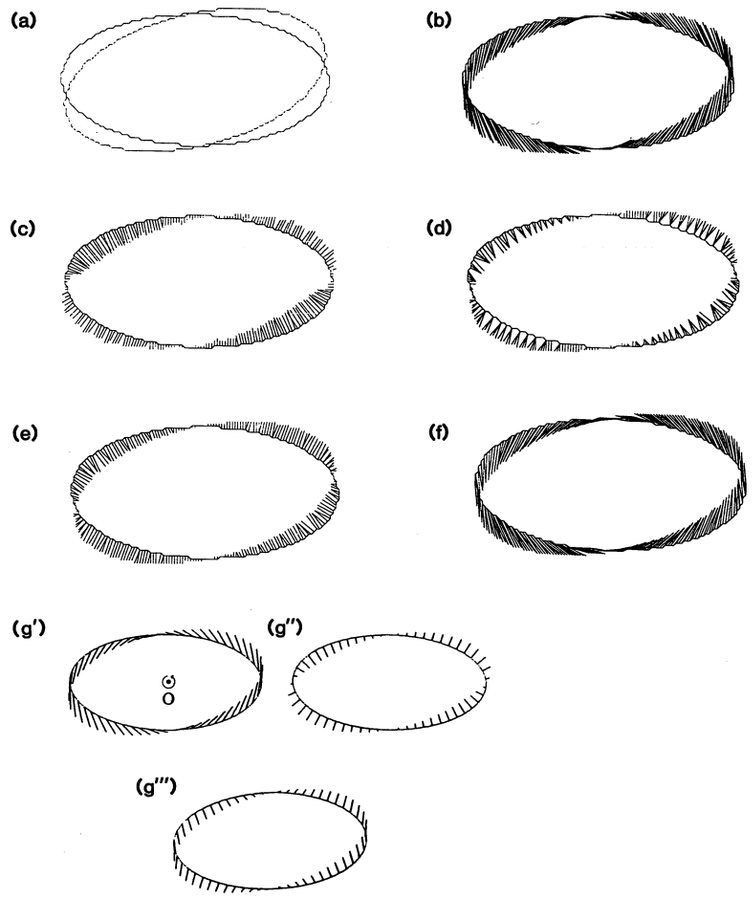

FIG. 7. Pure rotation: (a) the boundaries of the original and the new region (dashed); (b) actual displacement vectors; (c) perpendicular components of displacement vectors; (d) shortest-distance approximation; (e) smoothed approximation; (f) final approximation; (g) comparable results from Hildreth: (g′) actual displacement vectors; (g″) perpendicular components; (g‴) smoothed displacement field.

ment field. However, the final result after smoothing and rotating the new boundary represents a good estimate of the real situation. In this case extreme smoothing of the vector field around the boundary will *not* yield the optimal result and therefore the corresponding approximation by Hildreth [5, p. 52] (Fig. 7g) fails to come close to the real situation.

Experiment 3 (Fig. 8). Here *translation and rotation* were both applied. The initial approximation is already very good at points of small displacement, whereas areas of large displacement are not estimated well. This means that some areas along the boundary require rotation while other areas do not. Since rotation is applied uniformly to the entire boundary, no perfect mapping can be expected. Local smoothing improves the estimate significantly and the final mapping of minimum deformation is a very realistic approximation.

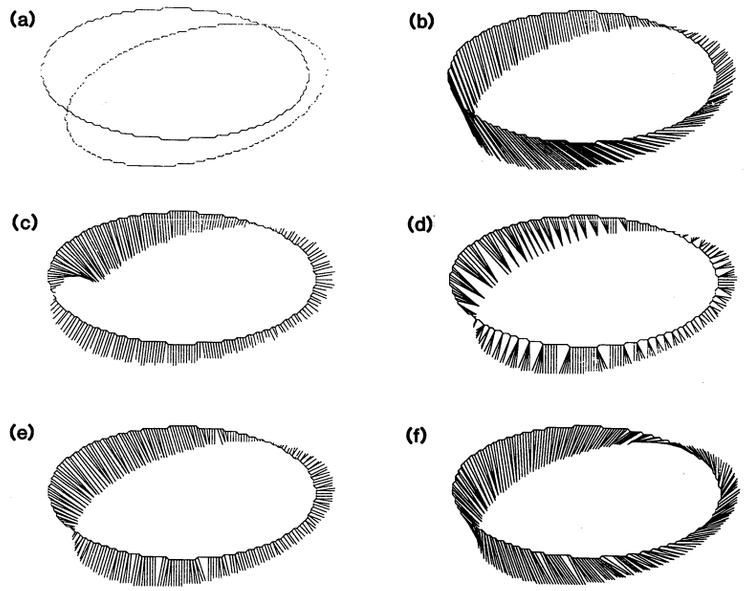

FIG. 8. Translation and rotation: (a) contours of the two regions (original contour is solid); (b) actual displacement vectors; (c) perpendicular components; (d) shortest-distance approximation; (e) smoothed approximation; (f) final approximation.

Experiment 4 (Fig. 9). This setup was originally chosen to demonstrate the limits of the approach in the presence of extremely wide rotation (60°). After looking at the initial approximation, one might well assume that the approach would fail in this situation. The final result, however, shows a very good estimate of the actual displacement field. Of course, in this case it applies even more that the *smoothest* displacement field as in Hildreth [5] would not yield a valid approximation.

4. CONCLUSION

A novel, region-oriented approach for estimating the displacement fields of moving objects has been devised and implemented, performing segmentation and two-dimensional motion analysis simultaneously. Corresponding regions in successive image frames are supposed to overlap, such that connectivity information is carried over from one frame to the next. *Region growing and wavefront propagation* of displacement data is used to obtain a new segmentation *and* a motion estimate simultaneously. Both processes can run in parallel, lending this technique to *pipelined* and *VLSI* implementations. For the initial *shortest distance* approximation, the solution is improved by *rotating* the mapping between contours until the correspondence of *least deformation* is found. Experiments conducted on elliptical regions show, that for the case of pure rotation almost perfect estimates are obtained. In the presence of translatory motion, the results depend crucially on how uniformly the terminal points of displacement vectors are distributed on the original and on the new boundaries. Here a local smoothing technique was applied to

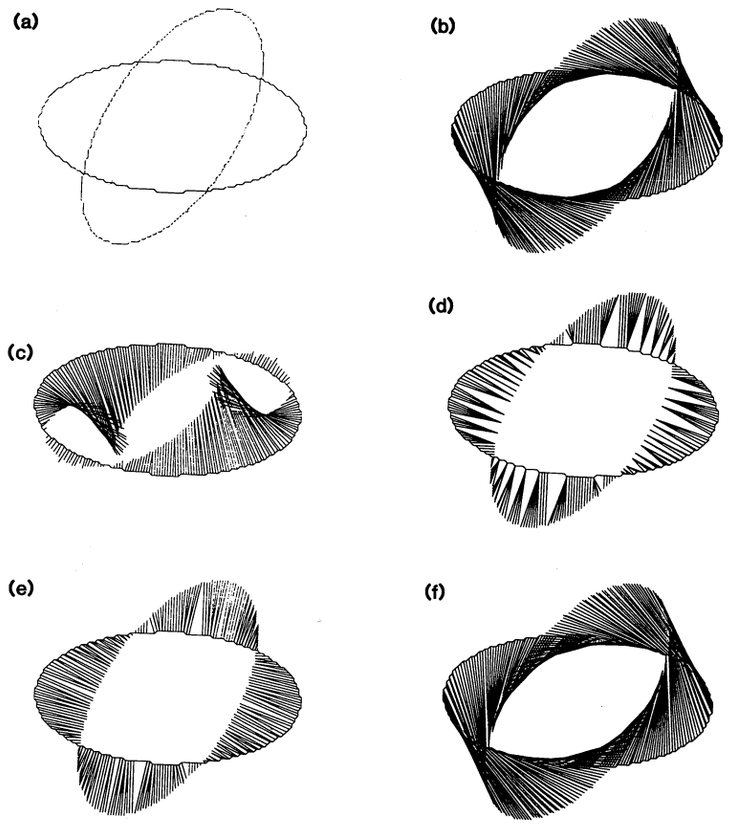

FIG. 9. Extreme rotation: (a) contours of the two regions (original contour is solid); (b) actual displacement vectors; (c) perpendicular components; (d) shortest-distance approximation; (e) smoothed approximation; (f) final approximation.

eliminate point clusters on either boundary. This method is fast and requires only local interaction between neighboring boundary elements. Improved results can be expected from increasing the size of the local support and the number of iterations for the local smoothing algorithm to distribute points more uniformly around the boundaries.

The experiments shown here were limited to regions of extremely simple shape and restricted motion. More work must be done to investigate the effects of region shape, especially nonconvex regions, deformation of regions due to arbitrary 3D object motion, and the effects of noise.

The wavefront region growing approach offers several advantages over other techniques commonly used in 2D motion analysis. *First*, it is based on region boundaries and therefore has the potential to supply denser displacement fields than tracking individual points. The extraction of regions is also more robust than the extraction of points or lines. *Second*, correspondence between features in successive images is easier to establish if overlapping regions can be assumed. *Third*, the direction of displacement vectors is not obtained from the local direction of a

boundary and is therefore less sensitive to noise. Finally, the described technique can be used for image segmentation itself, apart from its application in motion analysis.

ACKNOWLEDGMENTS

This work was supported in part by a grant from Ford Aerospace & Communications Corporation, Newport Beach, California. The authors are grateful to Rick Holben for his continued support.

REFERENCES

1. G. Adiv, Determining three-dimensional motion and structure from optical flow generated by several moving objects, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-7**, No. 4, 1985, 384–401.
2. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison–Wesley, Reading, MA, 1974.
3. S. T. Barnard and W. B. Thompson, Disparity analysis of images, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-2**, No. 4, 1980, 333–340.
4. R. C. Gonzalez and P. Wintz., *Digital Image Processing*, Addison–Wesley, Reading, MA, 1977.
5. E. C. Hildreth, *The Measurement of Visual Motion*, MIT Press, Cambridge, MA, 1984.
6. B. K. P. Horn and B. G. Schunck, Determining optical flow, *Artif. Intell.* **17**, 1981, 185–203.
7. H.-H. Nagel, Displacement vectors derived from second-order intensity variations in image sequences, *Comput. Vision Graphics Image Process.* **21**, 1983, 85–117.
8. T. Pavlidis, *Algorithms for Graphics and Image Processing*, Springer-Verlag, Berlin/Heidelberg, 1982.
9. J. W. Roach and J. K. Aggarwal, Determining the movements of objects from a sequence of images, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-2**, No. 6, 1980, 554–562.
10. S. Ullman, *The Interpretation of Visual Motion*, MIT Press, Cambridge, MA, 1979.