

# $\Psi$ -S Correlation and Dynamic Time Warping: Two Methods for Tracking Ice Floes in SAR Images

Ross McConnell, Ronald Kwok, *Member, IEEE*, John C. Curlander, W. Kober, and Shirley S. Pang

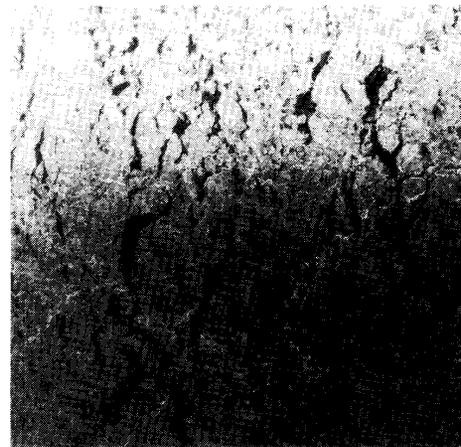
**Abstract**—In recent years, there has been an interest in automating the process of producing maps of the motion of ice floes from SAR images acquired a few days apart. A common approach is to correlate raw pixel values in order to find corresponding features in two images. The problem with this approach is that the search space is often prohibitive when ice floes rotate, as they frequently do. We present two algorithms for performing shape matching on ice floe boundaries in SAR images. These algorithms quickly produce a set of ice motion and rotation vectors that can be used to guide a pixel value correlator. The algorithms match a shape descriptor known as the  $\psi$ - $s$  curve. The first algorithm uses normalized correlation to match the  $\psi$ - $s$  curves, while the second uses dynamic programming to compute an elastic match that better accommodates ice floe deformation.

## I. INTRODUCTION

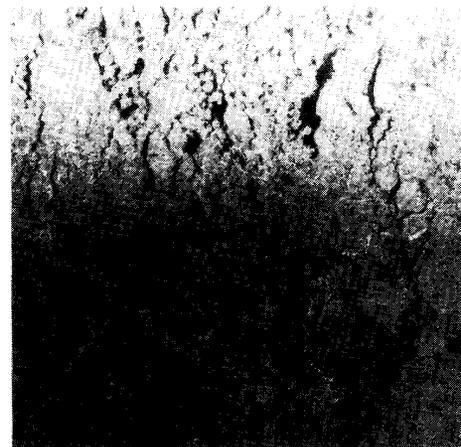
THE motion of ice floes plays a large role in the world's weather, because this motion exposes large expanses of unfrozen ocean water to the much more frigid Arctic air, and is therefore responsible for significant heat transfer between the ocean and the atmosphere. In addition, the movement of ice floes is of interest to ocean navigation and oil drilling. SAR imagery is suited for this task because it allows continuous coverage through clouds that prevail in the Arctic, as well as during the dark winter months. Figs. 1 and 2 are SAR image pairs that show substantial ice movement.

Because producing ice motion maps by hand from SAR image pairs is time consuming, and because the demand for ice motion data will be great, there has been an interest in developing algorithms for automating this process [1]–[6].

One way to address this problem is by correlating a patch of pixels from one image with the other image. Each position and rotation of the patch on the image defines a pairing of pixel values in the patch with a set of pixel values in a window of the image. The position and rotation of the patch that maximizes the correlation coefficient



(a)



(b)

Fig. 1. Seasat SAR images of ice floes, Beaufort Sea, revs. 1439 and 1482.

Manuscript received July 26, 1990; revised May 29, 1991.

R. McConnell and W. Kober are with the Voxel Corporation, Boulder, CO 80301.

R. Kwok, J. C. Curlander, and S. S. Pang are with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109.

IEEE Log Number 9102665.

is assumed to contain the same feature that is imaged in the patch.

This method is known as area correlation. This method is often computationally expensive. Computation of the correlation coefficient at each position takes time propor-

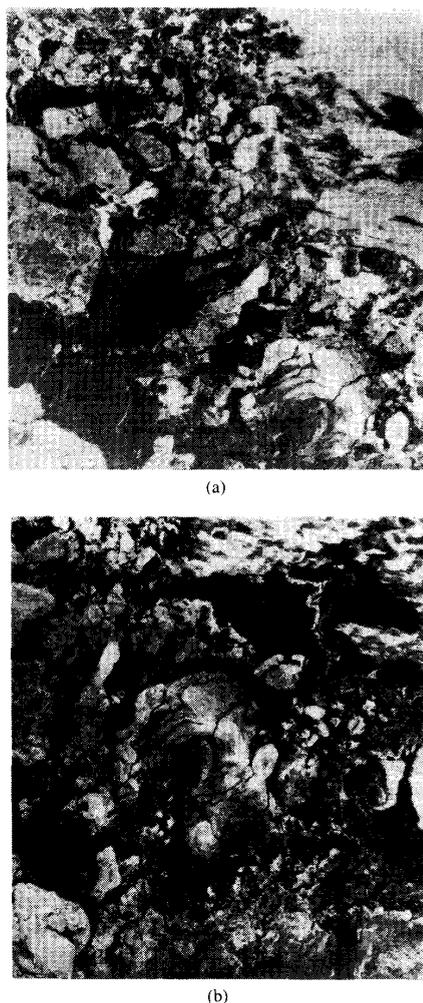


Fig. 2. Seasat SAR images, Beaufort Sea, revs 1409 and 1452.

tional to the area of the patch, and this must be computed at a number of locations proportional to the area of the region searched for a match. When features may rotate, as in the case of ice floe matching, the computation is multiplied again by the number of rotations tried at each potential match location. The amount of computation may be reduced by using a "resolution pyramid," whereby low resolution matches to find the approximate location and rotation of a match, and then the match is refined at higher resolutions. This approach still requires extensive computation when matches are ambiguous at a low resolution and for small patch sizes.

In addition, the ice pack frequently deforms. Area correlation is not robust to deformation of objects, since, if the object deforms, there is no effect of the patch that results in a correct pairing of a majority of the patch's pixels, and a correlation peak may fail to appear. In ice

imagery, this effect is often more pronounced at low resolution, hampering the effectiveness of a resolution pyramid.

What is needed for ice matching is computationally efficient matching methods that handle rotation and that are robust to deformation. These may then be used to produce a coarse ice motion and rotation map for a pair of images, which can be used to restrict the search space for area correlation.

We describe in this paper two approaches based on feature matching, one that handles rotation, and another that is robust to deformation. Use of these two methods in concert can produce the desired motion map with little computational expense. Both of the methods match extracted features by shape, using a shape descriptor known as the  $\psi$ - $s$  curve [7].

Section II presents the  $\psi$ - $s$  curve mathematically. Section III presents an approximation of the  $\psi$ - $s$  curve of a feature that is computable from the feature's digital representation and is appropriate for use in the matching algorithms. Section IV presents the first matching approach,  $\psi$ - $s$  correlation, while Section V presents the second approach, dynamic programming. Section VI addresses the problem of automatically discarding the incorrect matches that are invariably produced by these matching algorithms. Section VII presents some empirical data on the performance of the algorithms on Seasat SAR images.

## II. THE $\psi$ - $s$ CURVE

This section describes the  $\psi$ - $s$  curve, the shape descriptor used by the algorithms described in this paper for matching ice floe boundaries.

Let  $\vec{x}(s) = (x(s), y(s))$ ;  $0 \leq s \leq S$  be a continuous parametric curve that is parametrized by arc length. Further assume that  $\vec{x}(s)$  is differentiable except at isolated points.

Let the derivative  $\vec{x}'(s) = (x'(s), y'(s))$ , where

$$x'(s) = \begin{cases} dx(s)/ds, & \text{where } x \text{ is differentiable,} \\ \lim_{\epsilon \rightarrow 0^+} (x(s + \epsilon) - x(s))/\epsilon, & \text{where } x \text{ isn't differentiable.} \end{cases} \quad (1)$$

$$y'(s) = \begin{cases} dy(s)/ds, & \text{where } y \text{ is differentiable,} \\ \lim_{\epsilon \rightarrow 0^+} (y(s + \epsilon) - y(s))/\epsilon, & \text{where } y \text{ isn't differentiable.} \end{cases} \quad (2)$$

Let  $\theta_{\vec{x}}(s)$  be defined as follows over the interval  $0 \leq s \leq S$ :

$$\theta_{\vec{x}}(s) = \begin{cases} \cot^{-1}(x'(s)/y'(s)) & \text{if } y'(s) > 0, \\ \cot^{-1}(x'(s)/y'(s)) + \pi & \text{if } y'(s) < 0, \\ 0 & \text{if } y'(s) = 0 \text{ and } x'(s) > 0, \\ \pi & \text{if } y'(s) = 0 \text{ and } x'(s) < 0. \end{cases} \quad (3)$$

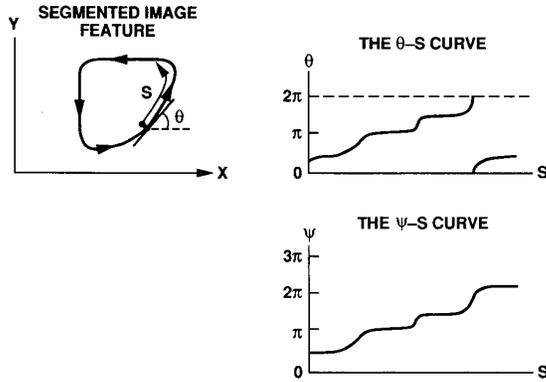


Fig. 3. Concept of a  $\psi$ - $s$  curve. The direction of the tangent to a shape is plotted as a function of arc length around the curve.

The value of  $\theta_{\vec{x}}(s)$  ranges over the interval  $[0, 2\pi)$ , and may have discontinuities whether or not  $\vec{x}(s)$  is continuously differentiable. This is because this definition is modulo  $2\pi$ , hence, there are discontinuities when the function reaches  $2\pi$  or  $0$ .

Suppose there are  $N$  continuous intervals on  $\theta_{\vec{x}}(s)$ . Let  $i_{\vec{x}}$  be an integer function such that  $i_{\vec{x}}(s_1)$  denotes the continuous interval on the  $\theta$ - $s$  curve of  $\vec{x}$  containing  $s_1$ , and  $1 \leq i_{\vec{x}}(s_1) \leq N$ , where  $N$  is the number of continuous intervals. Let  $u_{\vec{x}}(j)$  be the lowest value  $t$  such that  $i_{\vec{x}}(t) = j$ . Let  $k_i$ :  $1 \leq i \leq N$ , be an integer that is defined inductively as follows:

$$k_1 = 0$$

$$k_i = \begin{cases} k_{i-1}, & \text{if } -\pi \leq \lim_{\epsilon \rightarrow 0^+} \theta_{\vec{x}}(u_{\vec{x}}(i)) \\ & - \theta_{\vec{x}}(u_{\vec{x}}(i) - \epsilon) \leq \pi, \\ 1 + k_{i-1}, & \text{if } \lim_{\epsilon \rightarrow 0^+} \theta_{\vec{x}}(u_{\vec{x}}(i)) \\ & - \theta_{\vec{x}}(u_{\vec{x}}(i) - \epsilon) > \pi, \\ -1 + k_{i-1}, & \text{if } \lim_{\epsilon \rightarrow 0^+} \theta_{\vec{x}}(u_{\vec{x}}(i)) \\ & - \theta_{\vec{x}}(u_{\vec{x}}(i) - \epsilon) < -\pi. \end{cases} \quad (4)$$

The first case corresponds to a discontinuity in the  $\psi$ - $s$  curve where there is no phase wrapping, the second corresponds to a discontinuity where there is phase wrapping from  $2\pi$  to  $0$ , and the last case corresponds to a discontinuity where there is phase wrapping from  $0$  to  $2\pi$ . The value of each  $k_i$  gives the net phase wrapping that precedes each continuous interval of the curve.

Let the  $\psi$ - $s$  curve be defined as follows:

$$\psi_{\vec{x}}(s) = \theta_{\vec{x}}(s) + 2\pi k_{i(s)}. \quad (5)$$

Fig. 3 illustrates a shape and its  $\psi$ - $s$  curve. The  $\psi$ - $s$  curve modulo  $2\pi$  is the  $\theta$ - $s$  curve, and it eliminates the discontinuities associated with the  $\theta$ - $s$  curve's phase wrapping.

### III. EXTRACTING SHAPES FROM SAR ICE IMAGES AND DERIVING THEIR $\psi$ - $s$ CURVES

SAR images of ice floes may be segmented into regions with high backscatter (usually multiyear ice) and regions with low backscatter (usually new ice or open water). In general smoothing the image produces an image with a bimodal histogram, which can easily be used to determine a threshold that divides the pixels into a bright group and a dark group. Binarizing the smoothed SAR images with this threshold produces an image where multiyear ice shows up predominantly as one value, and new ice and open water show up predominantly as another value.

We have successfully used the Isodata clustering algorithm to cluster the pixel gray values and establish a threshold [6]. The images used are  $1024 \times 1024$  pixel SEASAT images with 100-m pixel spacing, where each pixel is generated by averaging 64 pixels to produce an image that approximates one that has sixty-four-look pixels. Each of the 100-m pixels is classified according to the mean value of 100-m pixels in the surrounding  $3 \times 3$  neighborhood. The histogram of these  $3 \times 3$  neighborhood averages usually shows a pronounced bimodal distribution that is suitable for clustering.

If pixels are considered to be square, and a feature is considered as a group of pixels, then the boundary of a segmented feature consists exclusively of vertically and horizontally oriented pixel boundary segments, or *crack edges*, that together make up a closed polygon. This precisely defines the boundary, and thus, arc length along the boundary.

Misclassified pixels usually appear after thresholding as scattered salt-and-pepper noise, and can be corrected with standard binary image cleaning techniques, such as inverting the classification of regions that have a circumference (perimeter arc length) that is smaller than a threshold. The result of thresholding one of the images of Fig. 1, vectorizing the region boundaries, and discarding all regions whose circumference is smaller than a threshold of 100 pixels, is shown in Fig. 4.

While these boundaries approximate the shapes of the features, the  $\psi$  values along these boundaries may assume only four values, hence, the  $\psi$ - $s$  curves suffer from strong aliasing. In addition, the arc length of these boundaries is affected by the orientation of a feature's boundary relative to the direction of tessellation, which introduces distortion of the  $s$  (arc length) parameter of the  $\psi$ - $s$  curve.

The interaction of a feature boundary with its raster segmentation produces "jags." Connecting the midpoints of the jags with straight lines, as illustrated in Fig. 5, produces a polygon that is a more plausible approximation of the true feature boundary. Arc-length is again well defined on this polygon, and less sensitive to the orientation of the tessellation of the raster image.

A close digital approximation of the feature's  $\theta$  curve can be obtained by sampling the orientation of the polygon's boundary at equal intervals of arc length around the polygon. The digital  $\psi$ - $s$  curve is then obtained from the

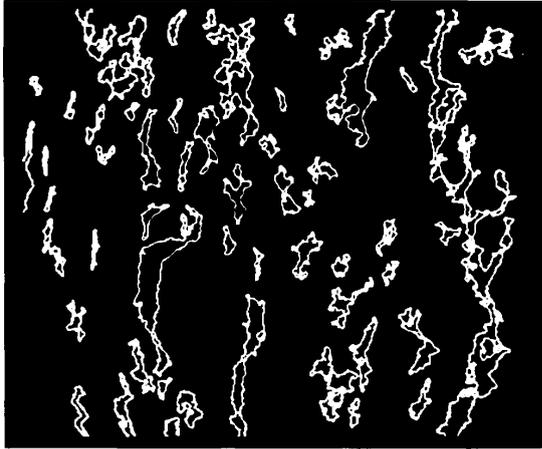


Fig. 4. Ice floe boundaries extracted from one of the images in Fig. 1. A gray-value threshold for distinguishing dark from bright regions is determined using the Isodata clustering algorithm. The image is thresholded, binary region boundaries are vectorized, and all boundaries whose circumference is less than 100 pixels are discarded.

$\theta$ - $s$  curve in the straightforward way, that is, by adding multiples of  $2\pi$  to remove the *modulo*  $2\pi$  wrap-around.

Henceforth, this approximation to the imaged feature's  $\psi$ - $s$  curve will be referred to as the *discretized  $\psi$ - $s$  curve*. The notation  $\psi_{\vec{x}}(i)$  denotes the  $i$ th element of a discretized  $\psi$ - $s$  curve of polygon  $\vec{x}$ .

#### IV. THE $\psi$ - $s$ CORRELATION APPROACH

Let  $\vec{x}_1(s)$  and  $\vec{x}_2(t)$  be the boundaries of a segmentable feature in two images, where the feature is free to translate and rotate in a plane parallel to the image plane.

To determine the translation and rotation, one may begin by deriving a discretized  $\psi$ - $s$  curve,  $\psi_{\vec{x}_1}$ , for a small segment of a boundary from one image, and the discretized  $\psi$ - $s$  curve,  $\psi_{\vec{x}_2}$ , for a whole boundary from the other image.

To find correspondences between the two curves, we correlate  $\psi_{\vec{x}_1}$  with each segment of length  $l$  in  $\psi_{\vec{x}_2}$ , using the following formula for normalized correlation:

$$r = \frac{N \sum_{i=1}^l \psi_{\vec{x}_1}(i) \psi_{\vec{x}_2}(k+i) - \left( \sum_{i=1}^l \psi_{\vec{x}_1}(i) \right) \left( \sum_{i=1}^l \psi_{\vec{x}_2}(k+i) \right)}{\sqrt{\left[ N \sum_{i=1}^l \psi_{\vec{x}_1}(i)^2 - \left( \sum_{i=1}^l \psi_{\vec{x}_1}(i) \right)^2 \right] \left[ N \sum_{i=1}^l \psi_{\vec{x}_2}(k+i)^2 - \left( \sum_{i=1}^l \psi_{\vec{x}_2}(k+i) \right)^2 \right]}} \quad (6)$$

The value of  $k$  that maximizes this expression gives the index of the beginning of the best-match segment in  $\psi_{\vec{x}_2}$ .

Part of the computational advantage of this approach comes from the fact that the correlation is a linear one rather than a two-dimensional one. A match is evaluated at each point along a boundary, rather than at each point in a region. The small segment of a boundary that is matched is described by a small number of  $\psi$  values rather

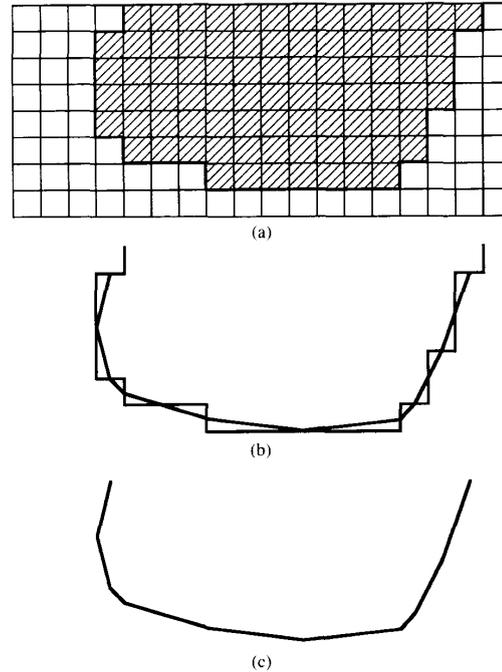


Fig. 5. Technique for deriving a boundary amenable to  $\psi$ - $s$  matching from a segmented region in a raster image. When the pixels are modeled as squares, the boundary of the segmented group of pixels consists exclusively of vertical and horizontal segments. In (A), a small portion of such a boundary is shown. A polygon is interpolated between the midpoints of the "jags" of the boundary, where it changes direction for one pixel and then resumes its course (B). This polygon (C) is a smoother representation of the boundary that has fewer artifacts of the raster representation of the region.

than by the much larger set of pixels in a two-dimensional patch, and thus, computation of the correlation coefficient at each position is less expensive. In addition, the method is unaffected by rotation of features; rotation of a feature adds a constant to its  $\psi$ - $s$  curve, which does not affect the correlation measure. While area correlation must be computed for a number of rotations at each potential match location,  $\psi$ - $s$  correlation does not.

#### V. THE DYNAMIC PROGRAMMING APPROACH

$\psi$ - $s$  correlation is a good model for matching in many cases. However, if one of the shapes is not identical to the other, the arc length is likely to be stretched or shrunk for portions of the boundary, and this corresponds to distortion in the  $s$  axis of the  $\psi$ - $s$  (Fig. 6). This frequently

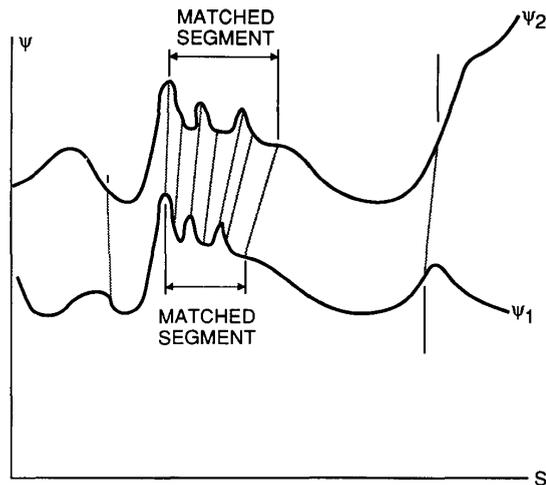


Fig. 6. Perturbation in the shape of a feature frequently leads to disturbance of the shape of the  $\psi$ - $s$  curve in both the  $\psi$  and  $s$  directions. Correlation is robust to disturbance in the  $\psi$  direction, but not to disturbance in the  $s$  direction.

happens with  $\psi$ - $s$  curves obtained from ice images. One cause is variability in the behavior of the algorithm for thresholding the image. Small changes in the length of a boundary also occur when small bits of floating ice adhere to the edge of an ice floe.

We solve this problem recursively, using dynamic programming. A simple recursive solution to many problems leads to recomputation of intermediate results in solving separate subproblems. Dynamic programming is the use of a table for storing these intermediate results so that they do not have to be recomputed. This device is frequently used to solve elastic matching problems, such as those arising in stereopsis [8], sequence comparison [9], and speech processing [10]. Its use in shape matching has been primarily restricted to handwriting analysis [11]. The subject is discussed in detail in a 1983 book on sequence comparison [12].

To illustrate how this approach works, we begin with the simple case where the end points of two segments of  $\psi$ - $s$  curves to be matched are known, and where the degree of match between them must be computed. In other words, in this example, we are not searching for a segment that best matches another; the segments are already given, and all we want to do is to find the elastic mapping between them.

Each of the  $\psi$ - $s$  curves is represented by a sequence of values. A "mapping" between the two curves pairs up values from the curves. Multiple elements from one sequence may be matched with a single element from the other sequence and *vice versa*, with the restriction that the mapping lines do not cross, as in the example in Fig. 7.

More formally, suppose sequence  $x$  has  $M$  elements, and sequence  $y$  has  $N$  elements, and a mapping between the two sequences contains  $P$  pairings, where  $P \geq \max(M, N)$ . Let the notation  $x_i$  stand for the  $i$ th element of  $x$ .

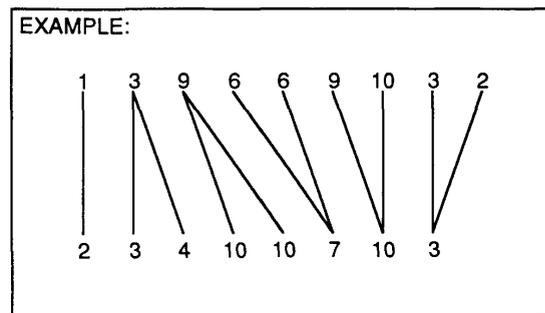


Fig. 7. Mapping between two sequences of numbers that minimizes the sum of differences of paired values. Finding this mapping for  $\psi$ - $s$  curves can be used to match  $\psi$ - $s$  curves that are perturbed in the  $s$  direction.

Each pairing has an index,  $k$ :  $1 \leq k \leq P$ . The mapping is specified with two integer functions  $i$  and  $j$ , where  $x_{i(k)}$  and  $y_{j(k)}$  give the elements paired by the  $k$ th pairing. Each of the integer functions increase by either 0 or 1 at each  $k$ , and at least one of them increases by 1 at each  $k$ .

Many mappings are possible between any two sequences, and one must have a "cost measure" for evaluating different mappings. A number of measures are possible, but for the purposes of this example, we will use the sum of absolute differences between paired values as the measure of the quality of the mapping. A smaller cost implies a better mapping. The *distance* between the two sequences is the minimum of the costs of all possible mappings between the two sequences.

The measure of the cost of a mapping is thus

$$c(i, j) = \sum_k |x_{i(k)} - y_{j(k)}|. \quad (7)$$

The distance between the two sequences is

$$D_{x,y} = \min(c(i, j)) \quad (8)$$

over all possible pairs of mapping functions  $(i(k), j(k))$ .

This distance can be computed recursively using the following rule. Let a *prefix* of a sequence denote the subsequence found in an interval that starts at the beginning of the sequence. Let  $d_{x,y}(m, n)$  denote the distance between the prefix of  $x$  of length  $m$  and the prefix of  $y$  of length  $n$ .

$$\begin{aligned} d_{x,y}(1, 1) &= |x_1 - y_1| \\ d_{x,y}(i, j) &= \min(d_{x,y}(i-1, j), d_{x,y}(i-1, j-1), \\ &\quad d_{x,y}(i, j-1)) + |x_i - y_j|. \end{aligned} \quad (9)$$

The solution of  $d_{x,y}(M, N)$  is  $D_{x,y}$ . An inductive proof of correctness of this algorithm is straightforward, and is a variant of that given in [11].

To solve this problem efficiently, one may construct a two-dimensional dynamic programming table, each of whose entries contains the cost of matching a single pair

	$x_1$	$x_2$	$\dots$	$x_{i-1}$	$x_i$	$\dots$
$y_1$	$d(1,1)$	$d(2,1)$	$\dots$	$d(i-1,1)$	$d(i,1)$	
$y_2$	$d(1,2)$	$d(2,2)$	$\dots$	$d(i-1,2)$	$d(i,2)$	
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	
$y_{j-1}$	$d(1,j-1)$	$d(2,j-1)$	$\dots$	$d(i-1,j-1)$	$d(i,j-1)$	
$y_j$	$d(1,j)$	$d(2,j)$	$\dots$	$d(i-1,j)$	$d(i,j)$	
$\vdots$						

Fig. 8. Use of dynamic programming table for computing (11) efficiently. The value of  $d_{x,y}(i, j)$  is filled into position  $(i, j)$  of the table. The values of  $d_{x,y}(i - 1, j)$ ,  $d_{x,y}(i - 1, j - 1)$ , and  $d_{x,y}(i - 1, j)$ , which are needed to evaluate (11) are contained in three neighboring elements. If the table is filled in row-by-row, the neighboring elements are filled in first, and the whole table can be filled in in constant time per entry.

of prefixes of the sequences. The table has size  $(M \times N)$  (Fig. 8). The distance between the prefix of  $x$  of length  $i$  and the prefix of  $y$  of length  $j$  goes in position  $(i, j)$  of the table. Note that the recursive cases needed to compute an entry correspond to the entries in the immediate left, upper-left, and upper neighbors of the entry. The table can be filled in row by row so that the left, upper-left, and upper neighbors of an entry are already complete when it is time to process the entry. The number of steps required to fill in each entry is constant for any size of table, and therefore the running time is proportional to the product of the lengths of the sequences to be compared.

When the process is completed, the lower-right entry contains  $d_{x,y}(M, N) = D_{x,y}$ . The mapping can be reconstructed by backtracking through the elements of the array giving rise to the solution. This may be accomplished by starting at the lower right entry, and at each entry proceeding to the upper, left, or upper-left neighbor that contains the minimum entry. For each entry  $(i, j)$  traversed by the path, the mapping has a pairing between element  $i$  of the first sequence and element  $j$  of the second sequence. Fig. 9 shows this process for the example sequences in Fig. 7. The highlighted backtracking path through the array yields the mapping depicted in Fig. 7. Fig. 10 shows an actual dynamic programming match for two  $\psi$ - $s$  curves derived from the boundaries of ice floes in segmented SAR images.

For tracking ice floes, however, the object is to *locate* the feature in one image that best matches a given feature in the other image. This can be accomplished by selecting a subinterval of the  $\psi$ - $s$  curve of a floe boundary from one image, and finding the subinterval of the  $\psi$ - $s$  curves of the other image most similar to it. This allows a match even if only short portions of an ice floe boundary match.

The subinterval of one sequence that is most similar to another sequence can be found by generalizing an algorithm developed for finding best-match subintervals of alphabetic sequences that was developed by Erickson and Sellers [9]. The sequence to be searched for the subinterval is put along the top of the dynamic programming table, and the sequence to be matched is put along the left.

	1	3	9	6	6	9	10	3	2
2	1	2	9	13	17	24	32	33	33
3	3	1	7	10	13	19	26	26	27
4	6	2	6	8	10	15	21	22	24
10	15	9	3	7	11	11	11	18	26
10	24	16	4	7	11	12	11	18	26
7	30	20	6	5	6	8	11	15	20
10	39	27	7	9	9	7	7	14	22
3	41	27	13	10	12	13	14	7	8

Fig. 9. Dynamic programming match of the two sequences of Fig. 7. The measure of similarity is recorded in the lower right-hand corner. The mapping between the sequences is recovered by following the course of the recursion giving rise to the minimum from the lower right-hand corner to the upper left-hand corner. Each position  $(i, j)$  traversed represents a pairing between element  $i$  of the upper sequence and element  $j$  in the left-hand sequence, and gives rise to the mapping shown in Fig. 7.

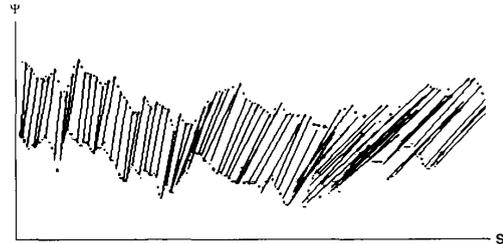


Fig. 10. Dynamic programming match for two  $\psi$ - $s$  curves derived from the boundaries of the same ice floe appearing in two SAR images. In practice, shorter segments are matched.

Each entry  $(i, 1)$  in the top row of the table is filled in with  $|x_i - y_1|$ . This fills in the entries  $(i, 1)$  in the top row with the distance between the first element of  $y$  and any subinterval of  $x$  ending at position  $i$  in  $x$ .

Filling in the rest of the table using the inductive rule given in (11) causes each entry  $(i, j)$  to contain the distance between the first  $j$  elements of  $y$  and any subinterval of  $x$  ending at position  $i$ .

Once the table is filled in, the bottom row of the array contains the distance between all of  $y$  and any subinterval of  $x$  ending at position  $i$  in  $x$ . The minimum value in this row gives the distance between  $y$  and any subinterval of  $x$ . The column where this minimum occurs gives the location where the subinterval ends in  $x$ . Starting at this entry and tracing back through the minimum upper, upper-left, or left entries leads to the top row at the column that gives the position of the beginning of the match in  $x$ . An example is shown for two sequences in Fig. 11.

One problem with the distance measure given by (11) is that when it is applied to  $\psi$ - $s$  curves, it does not penalize mappings consisting of severe warpings, which are

		1	1	3	1	6	1	3	3	2	5	3	7
1	0	0	2	0	5	0	2	2	1	4	2	6	
3	2	2	0	2	3	2	0	0	1	3	2	6	
2	3	3	1	1	5	3	1	1	0	3	3	7	
4	6	6	2	4	3	6	2	2	2	1	2	5	

Fig. 11. Using dynamic programming to find the subinterval of one sequence that best matches another sequence. The procedure is carried out in the same way as in Fig. 9, except that the top row of the table is filled in a way that does not penalize a match for starting in the middle of the upper sequence. The column where the minimum occurs in the lower row indicates the location of the end of the best match in the top sequence. Following the recursion giving rise to the minimum from that location to the upper row finds the location of the beginning of the match in the top sequence.

unlikely to represent a correct match in the context of  $\psi$ - $s$  curves. To remedy this situation, we use the following recurrence relation, which favors matches that do not involve too much warping:

$$d_{x,y}(i, j) = \min \begin{cases} d_{x,y}(i-1, j) + r \times |x_i - y_j| \\ d_{x,y}(i-1, j-1) + |x_i - y_j| \\ d_{x,y}(i, j-1) + r \times |x_i - y_j| \end{cases}$$

where  $r > 1$  is a factor that penalizes matches with excessive warping.

Although dynamic programming excels over  $\psi$ - $s$  correlation in the presence of deformation, the distance measure is sensitive to rotation, which adds a constant to one of the  $\psi$ - $s$  curves; it will not work as well as  $\psi$ - $s$  correlation in this case. It shares some of  $\psi$ - $s$  correlation's computational advantages over area correlation. Both  $\psi$ - $s$  correlation and the dynamic programming match a small segment only at each point along a boundary, rather than at each point in an area. The cost of both methods is proportional to the length of the small segment times the length of the boundary.

## VI. ELIMINATING FALSE MATCHES

Usually, some of the matches produced by  $\psi$ - $s$  correlation and dynamic programming are incorrect. Correct matches can be efficiently identified by noting that there is usually more than one correct match on each ice floe. Two or more matches on the same rigid body will exhibit rigid body motion. On the other hand, an incorrect match is unlikely to exhibit rigid motion with another match.

The following principles, illustrated in Fig. 12, are used as criteria for identifying rigid body motion between two matches.

1. The axis joining the features in one image must have the same length as the axis joining the features in the other image.
2. The relative rotation of these axes must be the same as the relative rotations of the features identified in each of the matches.

An estimate of the relative rotations of the features identified in each of the matches can be obtained from the

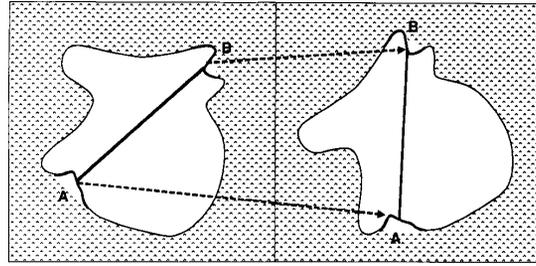


Fig. 12. Method of establishing selecting correct matches. All pairs of matches are examined to see if they exhibit rigid body motion. Features A and A' represent one of the matches, while features B and B' represent the other. The matches do not exhibit rigid body motion if the distance AB does not agree with the distance A'B', or if the rotation of A'B' with respect to AB does not agree with the rotation of A' with respect to A and with that of B' with respect to B. Any matches that do not show rigid body motion with at least one neighboring match are discarded.

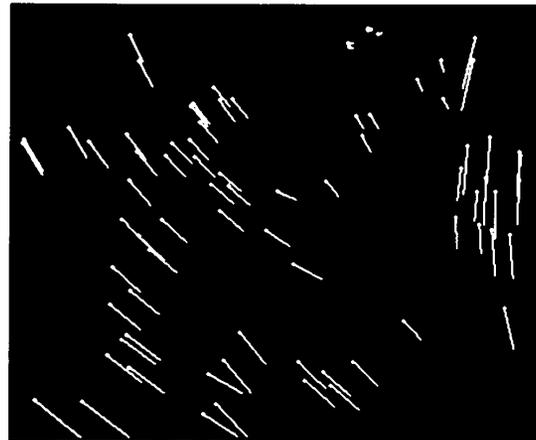


Fig. 13. Correct motion vectors derived by selecting a set of features from one of the images in Fig. 1 and finding their matches in Fig. 2 using  $\psi$ - $s$  correlation. Incorrect matches were discarded by hand.

mean difference between matched  $\psi$  values. In practice, the lengths of the axes and the rotations never coincide exactly, so these are deemed to coincide if they match to within tolerances given as parameters to the algorithm.

## VII. EXPERIMENTAL RESULTS

We compared  $\psi$ - $s$  correlation and dynamic programming on the image pair of Fig. 1. Each of the extracted boundaries was divided up into segments 75 pixel units long. Remaining fractions of segments were discarded. The  $\psi$ - $s$  curve for each of the segments was then correlated against all of the  $\psi$ - $s$  curves for the other image. The matches were verified by hand, and the incorrect matches were thrown out, leaving the set of motion vectors shown in Fig. 13.

Fig. 14 depicts the results of the same test with dynamic programming substituted for  $\psi$ - $s$  correlation. A value of 4.0 was used for the parameter  $r$  in (11). The number of correct matches is 117, as opposed to 78 for  $\psi$ - $s$  correlation.

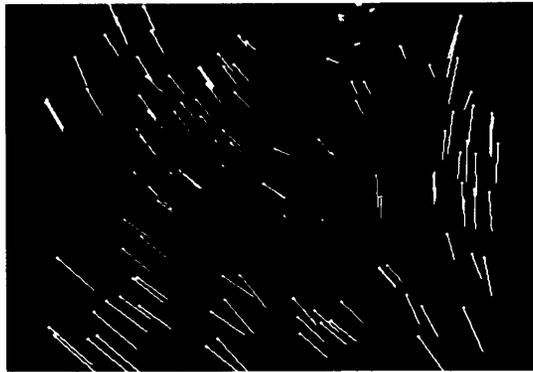


Fig. 14. Results of performing dynamic programming on the same set of features from Fig. 1 and discarding incorrect matches by hand.

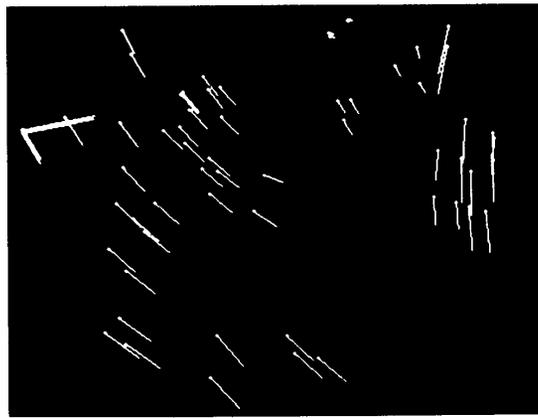


Fig. 15. Results of using the algorithm for automatically discarding incorrect matches on the  $\psi$ - $s$  correlation matches in Fig. 13.

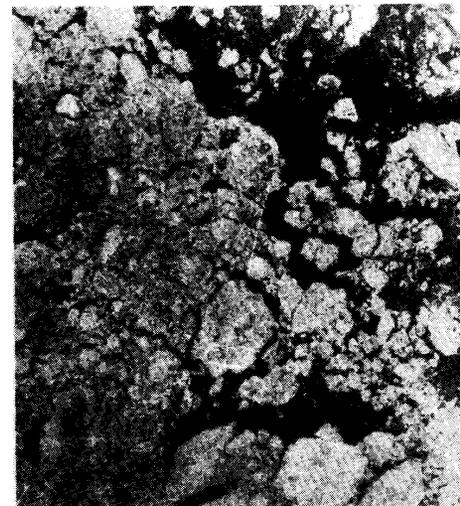
TABLE I  
COMPARISON OF  $\Psi$ - $S$  CORRELATION AND DYNAMIC PROGRAMMING

Image pair from Figure 1			
Match Method	Segment Length	# Attempted	# Successful
Correlation	20	567	36
D.P.	20	567	88
Correlation	75	278	78
D.P.	75	278	117
Correlation	150	111	28
D.P.	150	111	49
Image pair from Figure 16			
Match Method	Segment Length	# Attempted	# Successful
Correlation	20	364	12
D.P.	20	364	34
Correlation	75	351	45
D.P.	75	351	74
Correlation	150	141	20
D.P.	150	141	25

"Segment Length" refers to the length of the  $\psi$ - $s$  segments being matched, in pixel units.



(a)



(b)

Fig. 16. Image pair used in test whose results are given in Table I.

In Fig. 15, the manual elimination of bad matches is replaced with the automated bad match filter. The total number of matches is 278. The automated bad match filter correctly identifies 55 of the 74 correct matches, and falsely identifies two incorrect matches as being correct.

Table I gives the results of further tests of the  $\psi$ - $s$  correlation and dynamic programming algorithms on the images from Figs. 1 and 16. In all these cases, in which features did not rotate more than a few degrees, the dynamic programming produced a greater number of correct matches.

### VIII. CONCLUSIONS

Matching of ice floes is complicated by rotation of features and by deformation of the ice. Because of this, the search space for area correlation matching can become prohibitively large. If, however, an approximation of these can be found in advance of performing area corre-

lation, the search space can be reduced to manageable size. Both of the ice floe matching methods presented in this paper are computationally simpler than is area correlation, and more robust under circumstances commonly arising in ice floe image pairs. The  $\psi$ - $s$  correlation approach works well in situations where ice floes rotate greatly. The dynamic programming approach appears to have superior performance to the  $\psi$ - $s$  correlation approach when deformation of the ice floes is a problem, as long as the floes do not rotate too much. This might be remedied by using  $\psi$ - $s$  correlation to get rotation estimate that can be added to one of the  $\psi$ - $s$  curves before dynamic programming has been performed.

The output of the procedures is a set of motion and rotation vectors. This can be used directly as an ice floe motion map, or it can be used to reduce the search space for an area correlation approach if motion vectors on a regular grid are desired.

#### REFERENCES

- [1] R. McConnell, W. Kober, F. Leberl, R. Kwok, and J. Curlander, "Automated tracking of Arctic ice floes in multitemporal SAR imagery," *Int. Geosci. Remote Sensing Symp. (IGARSS)*, July 1989, Vancouver, Canada.
- [2] M. Fily and D. A. Rothrock, "Extracting sea ice data from satellite SAR imagery," *IEEE Trans. Geosci. Remote Sensing*, vol. 24, pp. 849-854, 1986.
- [3] J. F. Vesecky *et al.*, "Observation of sea-ice dynamics using synthetic aperture radar images: automated analysis," *IEEE Trans. Geosci. Remote Sensing*, vol. 26, pp. 38-47, 1987.
- [4] J. Daida and J. Vesecky, "Object-based feature-tracking algorithms for SAR images of the marginal ice zone," *Trans. Int. Geosci. Remote Sensing Symp.*, (Vancouver, Canada), 1989.
- [5] R. McConnell, F. Leberl, and W. Kober, "Tracking of Arctic ice floes in SAR images," in *Radargrammetric Image Processing*, Franz Leberl *et al.*, Eds., Norwood, MA: Artech House, 1989.
- [6] R. Kwok, J. Curlander, R. McConnell, and S. Pang, "An ice motion tracking system at the Alaska SAR Facility," *IEEE J. Oceanic Eng.*, vol. 15, pp. 44-54, 1990.
- [7] D. Ballard and C. Brown, *Computer Vision*. Englewood Cliffs, NJ: Prentice Hall, 1982.
- [8] Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic programming," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 7, pp. 139-154, 1985.
- [9] B. W. Erickson and Peter H. Sellers, "Recognition of patterns in genetic sequences," in *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, D. Sankoff and J. B. Kruskal, Eds., Reading, MA: Addison Wesley, 1983.
- [10] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. 26, pp. 43-49, 1978.
- [11] D. J. Burr, "Designing a handwriting reader," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 5, pp. 554-559, 1983.
- [12] D. Sankoff and J. B. Kruskal, Eds., *Time Warps, and Macromolecules: the Theory and Practice of Sequence Comparison*. Reading, MA: Addison-Wesley, 1983.



**Ross M. McConnell** was born in Denver in 1958. He received the B.A. degree from Williams College in 1981, and the M.S. degree in computer science from the University of Denver in 1985, where he worked on algorithms for finding recurring patterns in DNA sequences and other strings.

From 1985 until 1990, he was at Vexcel Corporation, where he did research on opposite-side SAR image matching and ice floe tracking. He is currently working toward the Ph.D. degree in computer science at the University of Colorado,

where he is doing research on algorithms for finding structures in graphs. His research interests include image processing, sequence analysis, and graph algorithms.



**Ronald Kwok** (M'85) received the B.Sc. (summa cum laude) degree from Texas A&M University, College Station, TX in 1976 and the Ph.D. degree from Duke University, Durham, NC in 1980. He was a postdoctoral fellow at the University of British Columbia, Vancouver, BC in 1981.

He then was with MDA in Richmond, BC, where he worked on algorithms for Landsat and Radarsat ground data systems. In 1985, he joined the Radar Science and Engineering Division at the Jet Propulsion Laboratory in Pasadena, CA, where he developed techniques for analysis of SAR imagery and served in a radar system engineering capacity on the Magellan and Alaska SAR Facility projects. He is currently Group Supervisor of the SAR Systems Development and Processing Group responsible for research and development of analysis and processing techniques for SAR data.

Dr. Kwok is a member of the American Geophysical Union, Electromagnetics Academy, Tau Beta Pi, Phi Kappa Phi, and Eta Kappa Nu.



**John C. Curlander** received the BSEE and MSEE degrees from the University of Colorado in 1976 and 1977, respectively. He received the Ph.D. degree from the University of Southern California in 1985.

He served as the supervisor of the SAR Systems Engineering Group and was the lead system engineer of the SIR-B project. He is currently the Assistant Manager Of Radar Science and Engineering Section at the California Institute of Technology, Jet Propulsion Laboratory, Pasadena, CA,

where he is primarily responsible for SAR signal processing research and development activities and the implementation of the Spaceborne Imaging Radar (SIR-C) ground data system. His research has centered on preprocessing and post-processing techniques to improve the performance of the SAR correlator.

Dr. Curlander has received several NASA achievement awards for his contributions to the Seasat and SIR programs as well as the NASA Exceptional Service Medal for his research in SAR signal processing techniques. He has authored over 50 technical articles on SAR including a recently published text on SAR signal processing and has received more than 15 new technology awards from NASA.



**Wolfgang Kober** received the B.S., M.S., and Ph.D. degrees in mathematics from the University of Colorado at Boulder in 1969, 1971, and 1976, respectively.

He has worked at Control Data Corp., Martin-Marietta Corp., Computer Technology Associates, and Advanced System Technologies. From 1987 to the present, he has been a Vice President of VEXCEL Corporation. He was a session chairman at Sixth Meeting of the Coordinating Group on Modern Control Theory, Aberdeen Proving Grounds, VA, 1984. He has occasionally taught graduate-level courses in analytic queueing and reliability theory, digital control theory, and signal processing at the University of Colorado at Denver. He was a contributor to numerous chapters of the book: *Radargrammetric Image Processing*, F. Leberl, Artech Press, 1990. His present research interests include image processing and computer vision, multidimensional signal processing, and SAR imaging.



**Shirley S. Pang** received the B.A. degree in mathematics from Douglass College, and the M.S. degree in computer science from the University of Hawaii. She joined the Jet Propulsion Laboratory in Pasadena, CA in 1973, and has participated in the software design and development for various interplanetary flight projects and ground processing systems. She is currently involved in the software development for the Alaska SAR Facility Geophysical Processor System (GPS) and Archive and Operations Systems (AOS), as well as the Shuttle Imaging Radar Project.

She is the recipient of five NASA Group Achievement awards for her contributions to the Viking, Voyager, and Infrared Astronomical Satellite (IRAS) projects, and the Interim Digital Processor (IDP) and the Shuttle Imaging Radar B (SIR-B) processor.