Scale-based Description and Recognition of Planar Curves and Two-Dimensional Shapes

Farzin Mokhtarian[1] and Alan Mackworth[2]

Technical Report 84-15
October 1984

Laboratory for Computational Vision
Department of Computer Science
University of British Columbia
Vancouver, B.C.
Canada V6T 1W5

---

[1] Current address: Schlumberger-Doll Research, Old Quarry Road, Ridgefield, CT 06877.

[2] Fellow of the Canadian Institute for Advanced Research.

Abstract

The problem of finding a description for planar curves and two-dimensional shapes at varying levels of detail and matching two such descriptions is posed and solved in this paper. A number of necessary criteria are imposed on any candidate solution method. Path-based Gaussian smoothing techniques are applied to the curve to find zeroes of curvature at varying levels of detail. The result is the 'generalized scale space' image of a planar curve which is invariant under rotation, uniform scaling and translation of the curve. These properties make the scale space image suitable for matching. The matching algorithm is a modification of the uniform cost algorithm and finds the lowest cost match of contours in the scale space images. It is argued that this is preferable to matching in a stable scale of the curve because no such scale may exist for a given curve. This technique is applied to register a Landsat aerial image of the Strait of Georgia, B.C. (manually corrected for skew) to a map containing the shorelines of an overlapping area.

1. Introduction

Images interpreted by computational vision systems can be those of two- or three-dimensional objects. Two-dimensional shapes (such as letters of the alphabet, chromosomes and shorelines in aerial photographs) are bounded by, or composed of, planar curves. Various curve representations have been proposed in the computational vision literature. A reliable representation, suitable for matching, should be essentially invariant with the rotation, scaling and translation of the curve to make recognition of the curve possible after arbitrary instances of those transformations. Moreover, it should represent the curve at varying levels of detail rather than at only one level.

The goals of this paper are:

(a) To find a method of obtaining a representation for a two dimensional curve which is invariant under rotation, scaling and translation of the curve.

(b) To develop a matching algorithm to find the best match of two such representations. Such an algorithm should also be able to match one representation to part of another, to account for incomplete data.

(c) To apply the techniques developed in (a) and (b) to register a Landsat satellite image of an area to a map which contains the shorelines of an overlapping area.

2. Some criteria for a reliable representation

A number of necessary criteria that any reliable method for curve description and recognition must satisfy are presented here:

(a) The representation must be computable.

(b) The representation should be essentially invariant under rotation, uniform scaling and translation of the curve, otherwise, reliable recognition will not be

possible.

(c) The representation should contain information about the curve at varying levels of detail. Moreover, it should be clear from the representation which features of the curve belong to coarser levels of detail and which features to finer levels.

(d) The amount of change in the representation should correspond to the amount of change made to the curve. In other words, a small change to part of the curve should create a small change in the representation.

(e) Arbitrary choices should not affect the representation.

(f) In case of open curves intersecting the frame, the representation should only change locally with the location of the cutoff points.

(g) The representation should uniquely specify a single curve, otherwise it would be possible to match a curve against a class of curves all of which have the same representation. This criterion only requires uniqueness up to the curve equivalence classes induced by requirement (b) above.

Several shape representation techniques may be judged by these criteria. For example, the Hough transform has been used to detect straight lines [1], circles [2] and arbitrary shapes [3] in images. O'Gorman and Clowes [4] used the direction of the gradient to improve the efficiency and accuracy of the transform. In a typical Hough transform technique, edge elements in the image vote for the parameters of the objects they can be located on. All the votes are gathered in a *parameter space*. The highest peaks indicate the location of the objects in the image. The Hough transform can suffer from false peaks in the accumulator array due to accidental matches with the data. Poor results will be obtained for incomplete data even if the existing data

matches well with the model. In order to account for rotation, scaling and translation of the shape, more parameters must be added to the parameter space which makes the implementation impractical. The generalized Hough transform [3] also suffers from the arbitrary choice of the required reference point for the shape.

Another class of methods are those which find hierarchical straight line or angle approximations to the curve [5,6,7,8]. A hierarchy of straight line approximations to the curve is formed by initially joining the endpoints and recursively refining each approximation by breaking it at the point on the curve furthest away from it. The algorithm stops when every point on the curve is within some threshold distance from the straight line segment which approximates the portion of the curve containing the point. This method does combine information about the curve at various levels of detail but the representation can change greatly due to a small change in the curve. Moreover, if the curve is closed, an arbitrary choice of endpoints must be made which will certainly affect the representation. Strip trees also use a similar idea in order to represent a curve. The only difference is that a rectangle which contains all of the points on a segment is used to represent that segment. Smaller and smaller rectangles are used to give a finer representation of the curve. The deficiencies of the previous method also make this one unattractive.

The "codon" representation proposed by Hoffman and Richards [9] satisfies many of our criteria in that it is based on segmenting at minima of curvature. However, it does not reflect considerations of detail (c) or sensitivity (d).

A final candidate method breaks the curve into several segments (if needed) such that each segment is a single-valued function $y = y(x)$. Then the Stansfield-Witkin method [10,11] can be used to construct a scale space image of the curve. The effectiveness of this method depends on the number of segments the curve has to be divided into. In the special case where the curve already is a single-valued function, no

divisions are needed and the method would work but in general (when divisions are necessary), there will be problems with handling the boundary conditions at each break in the curve. This method violates criterion (b) in that the representation is not invariant under rotation.

3. Scale-Based description of planar curves

This section describes a method for computing a representation for a curve invariant under rotation, uniform scaling and translation of the curve as explained in [12]. This method is based on finding points of inflection on the curve at varying levels of detail using a path length parameter and combining them to obtain the scale space image of the curve.

3.1. Finding points of inflection on a planar curve

It is desired to find zero-crossings in curvature of the curve at varying levels of detail, that is, for varying degrees of smoothing of the curve. Since a planar curve does not behave like a single-valued function in general, a parametrization of the curve should be found which makes it possible to compute the curvature of the curve at varying levels of detail. Such a parametrization is made possible by considering a path length variable along the curve and expressing the curve in terms of two functions $x(t)$ and $y(t)$:

$$C = \{ x(t), y(t) \}$$

where t is a linear function of the path length ranging over the closed interval [0,1]. If the curve is closed, $x(t)$ and $y(t)$ are periodic functions. The curvature κ of a planar curve at a point P on the curve is defined as the instantaneous rate of change of the slope angle ψ of the tangent at point P with respect to arc length s, and is equal to the inverse of the radius ρ of the *circle of curvature* at point P.

$$\kappa = \frac{d\psi}{ds} = \frac{1}{\rho}$$

The circle of curvature at point P is a circle tangent to the curve at point P whose center lies on the concave side of the curve and whose curvature is the same as that of the curve at point P. The circle of curvature is also called the *osculating circle* because it has a higher degree of contact with the curve at point P than any other circle.

κ can be computed if it is expressed in terms of the derivatives of functions $x(t)$ and $y(t)$.

Define

$$y' = \frac{dy}{dx} \qquad\qquad y'' = \frac{d^2y}{dx^2}$$

It is known that:

$$\kappa = \frac{y''}{(1+(y')^2)^{3/2}}$$

therefore y' and y'' should be expressed in terms of the first and second derivatives of $x(t)$ and $y(t)$.

Denote

$$\dot{x} = \frac{dx}{dt} \qquad \ddot{x} = \frac{d^2x}{dt^2} \qquad \dot{y} = \frac{dy}{dt} \qquad \ddot{y} = \frac{d^2y}{dt^2}$$

Then

$$y' = \frac{\dot{y}}{\dot{x}}$$

and

$$y'' = \frac{dy'}{dx} = \frac{\frac{d}{dt}\left(\frac{\dot{y}}{\dot{x}}\right)}{\dot{x}} = \frac{\dot{x}\,\ddot{y} - \dot{y}\,\ddot{x}}{\dot{x}^3}$$

using the derivative rule:

$$\left(\frac{u}{v}\right)' = \frac{u'v - v'u}{v^2}$$

So we obtain

$$\kappa = \frac{\dot{x}\,\ddot{y} - \dot{y}\,\ddot{x}}{(\dot{x}^2+\dot{y}^2)^{3/2}}$$

In order to compute the curvature of the curve at varying levels of detail, functions $x(t)$ and $y(t)$ are convolved with a one-dimensional Gaussian kernel $g(t,\sigma)$ of width σ [13]:

$$g(t,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-t^2}{2\sigma^2}}$$

$X(t,\sigma)$, the convolution of $x(t)$ and the Gaussian kernel, is defined as:

$$X(t,\sigma) = x(t) \circledast g(t,\sigma) = \int_{-\infty}^{\infty} x(u) \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(t-u)^2}{2\sigma^2}} du$$

$Y(t,\sigma)$ is defined similarly.

In the smooth curve one needs $\dot{X}(t,\sigma)$, $\dot{Y}(t,\sigma)$, $\ddot{X}(t,\sigma)$ and $\ddot{Y}(t,\sigma)$ to compute curvature. These can be computed from $x(t)$ and $y(t)$ using

$$\dot{X}(t,\sigma) = \frac{\partial X(t,\sigma)}{\partial t} = \frac{\partial[x(t) \circledast g(t,\sigma)]}{\partial t} = x(t) \circledast \left[\frac{\partial g(t,\sigma)}{\partial t}\right]$$

and

$$\ddot{X}(t,\sigma) = \frac{\partial^2 X}{\partial t^2} = x(t) \circledast \left[\frac{\partial^2 g(t,\sigma)}{\partial t^2}\right]$$

and similarly for $Y(t,\sigma)$. Figure 1 shows the coastline of Africa. Figure 2 shows an application of this method to that coastline, with the zeroes of curvature marked on each smoothed curve.

3.2. How to handle the endpoints of the curve

As mentioned earlier, if the curve is closed, functions $x(t)$ and $y(t)$ can be treated as periodic functions which eliminates all edge effects. But if the curve is open, these functions are not periodic. A way must be found to convolve the mask with the curve when part of the mask is beyond the endpoint since the convolution value for all points on the curve should be computed.

There is a basic difference in the nature of two open curves, one of which is contained completely inside the image and the other which ends at the frame boundary. In the first case there is in fact no missing information and the endpoint problem can be solved by creating an extension to the curve which is an extrapolation of points close to the endpoint. As long as one is consistent, the results for similar curves will be similar. In the second case there is indeed a loss of information around the endpoint. There are several ways to handle this problem. Each one will be discussed:

(a) One could still extrapolate the curve beyond the endpoints. Since this does not in general result in a reconstruction of the original shape, the results can be different for two similar curves which are cut at different locations by the frame boundary.

(b) The endpoints of the curve can be joined with a straight line to change it into a closed curve. The arguments of section (a) also apply here.

(c) When the endpoint of the curve is reached, turn back and go in the opposite direction, towards the other endpoint. This method will also not guarantee similar results and it also suffers from the fact that artificial inflection points will be introduced at the endpoints of the curve.

(d) Repeat the endpoint of the curve as many times as necessary: treat *x(t)* and *y(t)*

as constant functions beyond the endpoint. This method does not introduce extra inflection points and seems like the appropriate thing to do in absence of other information. It is also the easiest method to implement and is the one used in this paper to handle the endpoints.

3.3. The scale space image of a planar curve

Section 3.1 described how to find points of inflection on the curve at varying levels of detail. This section will describe how to combine that information in the form of a generalized scale space image in order to obtain a representation for the curve.

Compute a Gaussian mask using a value of σ corresponding to the finest level of detail desired. The limit is σ=0 which is equivalent to convolving an impulse function with the functions $x(t)$ and $y(t)$. In practice a higher value can be used to avoid the excessive detail usually obtained at very fine levels of detail. Find points of inflection on the curve corresponding to this value of σ using the techniques explained in section 3.1. Mark these points in a coordinate system where the x-axis shows the value of the path-length parameter, t, along the curve and the y-axis represents σ, the width of the Gaussian kernel.

Increase the value of σ by a small amount, find the new inflection points and mark them also in the scale space image. Repeat this process until no inflection points are found for some value of σ. The whole scale space image has been derived (Figure 3).

The scale space image can be thought of as a binary image which is 1 wherever there is an inflection point on the curve corresponding to some values of the path-length parameter and sigma and 0 otherwise.

Some remarks about the scale space image are appropriate:

(a) The scale space image contains contours which are closed at the top and open at the bottom. There may be contours open at the top but these contours are incomplete ones which end at the edge of the scale space image and are due to incomplete information.

(b) Contours in the scale space image never intersect each other but may "touch" at fine levels of detail.

(c) Yuille and Poggio [14] have shown that under certain assumptions, no new inflection points are created at higher levels of detail (more smoothing). This property can be used to speed up the computation of the scale space image by a considerable amount. Indeed once one is beyond fine levels of detail, one can track the inflection points by only computing the curvature and looking for zero-crossings in the neighborhood of the previous inflection points.

(d) Yuille and Poggio [15] have also shown that almost all signals can be reconstructed up to an equivalence class from their scale space images. This implies that the scale space image is a unique representation for those signals.

(e) For some open curves, there may be one or more incomplete contours in their scale space images. Since the "movement" of such contours towards the edge can be quite slow, a lot of unnecessary computation time will be spent if such a contour is allowed to take its normal course. A shortcut can be made by anticipating cases such as this. As soon as only one inflection point is obtained on the whole curve for some value of sigma, find the slope of the corresponding contour to determine whether it is moving towards the right or the left edge and connect the contour to the correct edge.

4. An algorithm to match scale space images

The matching is carried out in the scale space image because of the invariance properties it displays. Furthermore, matching in the scale space image is preferable to matching at a specific scale. A specific scale is usually chosen by imposing some stability criterion on the features of the curve. It is not clear that there always exists a "most stable" scale for a given curve. That approach would be unreliable for such a curve.

A scale space image can be considered to be a hierarchical representation for a curve. An imaginary super-contour exists at the root of this tree. This contour contains all the real contours in the scale space image. Each real contour has zero or more children, contours which exist inside it, and zero or more siblings, contours which share the same parent. Every contour has a peak, a right branch and a left branch unless it is incomplete in which case one branch is totally missing and the other one incomplete or both branches are incomplete.

The algorithm described in this section is an adaptation of the Uniform Cost Algorithm [16] which is a special case of the A^* algorithm [17]. It finds the lowest cost match between contours in the two scale space images. The image space transformation parameters which take one curve to the other are then computed from that match. There are 4 parameters which correspond to uniform scaling, rotation and translation of the curve. The two curves can then be registered to show the goodness of match between them.

4.1. Creating the initial matching nodes

The algorithm starts out by creating a queue of nodes corresponding to the possible match of every pair of contours in the two scale space images. Since the two original curves could be at different scales, it is possible for contours at different levels to

match with each other. There are two scale space transformation parameters mapping one contour to another which need to be computed for each node. These two are the scale k and shift d parameters. The relationship between the old coordinates, t and σ, and the new coordinates, t' and σ', is as follows:

$$t' = k\,t + d$$

$$\sigma' = k\,\sigma$$

Only one pair of scale space points is needed to compute k and d. These can be computed using the coordinates of the peaks of the two contours since peaks provide a pair of points on the contours which correspond to each other.

These parameters are always used to transform the smaller contour so that it can be matched against the larger one. If the smaller contour is in the first scale space image, then the transformation is from the first to the second scale space image. If it is in the second scale space image, there is a reverse transformation. As a result, contours do not shrink and matching errors are better accounted for. The same set of parameters is used to match the next pairs of contours when a node is expanded.

The cost of match between two contours is defined as the average distance between them after one of them has been transformed. The average distance between two contours is the average of the distances between the peaks, the right branches and the left branches. Incomplete contours are not matched since their true shape is not known. Since it is desirable to find a match corresponding to the coarse features of the curves, there is a penalty associated with starting a match with a small contour. This penalty is a linear function of the difference in height of that contour and the tallest contour of the same scale space image.

4.2. Expanding a node

The algorithm proceeds by finding the lowest cost node from the queue of initial

nodes and expanding it. The cost of expanding that node is computed and added to the previous cost. Then the new node is added to the queue and this process is repeated until the lowest cost node can not be expanded any more. The correct match is assumed to be the one indicated by this node.

In order to expand a node, the next pair of contours in the scale space images to be matched must be found. Instead of randomly choosing any other contour in the scale space image and finding the next match value, this algorithm makes use of the constraints of the scale space image as much as possible.

To find the next two contours to match, a contour is searched for in the first scale space image using an order-by-height procedure. If it is found, then a contour is searched for (but not necessarily found) in the second scale space image which would correspond to the contour from the first image. The following is a detailed description of this procedure. Note that if at any step the program fails to find a contour from the first image, it will search for a contour in the second image using the same guidelines for finding contours in the first image before going to the next step.

(a) If the last contour matched from the first scale space image has any children then its tallest child will be the next contour to be matched. In order to find the next contour in the second image to match, use the transformation parameters to estimate its location. Since its parent is known, one can search for any children of that parent whose peaks fall inside that range. If more than one such contour is found, choose the one whose height is closest to that estimated by the transformation parameters. If no corresponding contour is found in the second image, the first contour will be matched against the null contour, which has a height of zero.

(b) If the last contour matched from the first image does not have any children,

search for its next smaller sibling. If such a sibling exists, it will be the next contour to be matched. To find the next contour from the second image, use the procedure outlined in section (a).

(c) If the last contour matched from the first image does not have any smaller siblings either, return one level and search for a smaller sibling of its parent. Repeat this process until either the first ancestor which has a smaller sibling is reached or there are no more ancestors left. In the latter case, no more expansion is possible and the algorithm returns with the lowest cost node.

If the curve for which the scale space image is computed is closed then the scale space image will be periodic: moving right when one is at the right edge of the scale space image will place one at the left edge and *vice versa*. This must be taken into account when applying transformations to the contours since contours may wrap-around in the scale space image.

4.3. Dealing with incomplete information

If one of the curves being matched only matches with part of the other curve, the contours in the scale space image for that curve will correspond to only some of the contours in the other scale space image. This will not result in poor matches since it is easy to find a region in the larger scale space image which corresponds to the smaller scale space image (again wrap-around is possible for closed curves) and extract the relevant contours which should be used in the matching process.

5. Application

The Landsat registration problem was chosen to illustrate the curve description and recognition methods explained in sections 3 and 4. The Landsat registration problem is that of registering a Landsat aerial image of an area to a map which contains

the shorelines of the same area. Shorelines must be extracted from the Landsat image before they can be matched against the shorelines in the map. It should be noted that exact registration of the shorelines is not a goal of this paper. The purpose is to demonstrate that the method is successful in finding shapes which have basically the same features.

Section 5.1 explains a simple method for extracting shorelines from the Landsat image. Section 5.2 describes a way of putting together the curve description and recognition methods in order to create a working system to solve the Landsat registration problem.

5.1. Obtaining shorelines from the Landsat image

The Band 7 (near infrared) Landsat image of part of the Strait of Georgia, B.C. is a gray-level image showing a clear distinction between water and land in most areas (Figure 4). A histogram of gray-level distributions of all the pixels in the Landsat image was obtained (Figure 5a). There are two main peaks in this histogram corresponding to water and land respectively but there are also many smaller peaks due to noise and one at the very end of the histogram composed of saturated pixels due to snow patches. If one could locate the land and water peaks in the histogram, the value of the trough between those two peaks could be used to threshold the image. The contours of dark regions in the thresholded image could then be followed to obtain approximations to the shorelines. These peaks can in fact be found by smoothing the histogram, which can be treated as a function of one variable, with a mask based on a one-dimensional Gaussian function [18]. The amount of smoothing is controlled by σ, the standard deviation of the Gaussian.

The minimum value of σ which results in two peaks can be obtained using binary search. The resulting smoothed histogram is shown in Figure 5b. To find the

location of the peaks one can simply convolve the first derivative of the Gaussian, using the correct value of σ, with the histogram and look for transitions from positive to negative (since the first derivative measures the slope of the smooth curve and a peak is where the slope goes from a positive value to a negative value). Similarly the trough between the two peaks can be found by looking for a transition from negative to positive.

Once the location of the trough is known, the Landsat image can be thresholded using the intensity value of the trough as the thresholding value. Every pixel in the image will be classified as water if its gray-level is less than the thresholding value and as land if its gray-level is greater than or equal to the thresholding value. The result is a binary image where 1 pixels represent land and 0 pixels represent water.

If the land and water pixels are assumed to be normally distributed with the mean of the distribution at the location of the peak then the intersection point of the land and water distributions, when chosen as a thresholding value to threshold the image, results in a minimum total number of land and water pixels classified incorrectly. A thresholding value larger than this value would result in fewer water pixels classified incorrectly but even more land pixels classified incorrectly. Similarly, a thresholding value less than this value would result in fewer land pixels classified incorrectly but even more water pixels classified incorrectly. It was decided to use the intensity value of the trough to threshold the image since the algorithm to find the trough was easier to implement: the difference in results is not significant for the purpose of this paper.

To obtain the land-water boundaries from the thresholded image, one need only follow the contours of the land masses. One simple algorithm for following the contours of light regions in binary images [19] starts from any point on the contour and turns left if it is on a light pixel and right if it is on a dark pixel. This process is

repeated until one of the neighbors of the starting point is reached. As a result the contour of the region will be traversed in a clockwise fashion. If it is desired to traverse the contour of the region in a counter-clockwise fashion, one should turn right when on a light pixel and left when on a dark pixel. This may be necessary in order to traverse the contour of a region only partly contained in the image.

Duda and Hart indicate that the algorithm may show 4-connected or 8-connected behavior depending on the starting point and the direction in which it is entered. One can easily force consistency on this algorithm by finding all the light pixels in the image which are only connected diagonally and disconnecting them. All the resulting regions will be 4-connected therefore the algorithm will always show 4-connected behavior also. Disconnecting pixels which are only connected diagonally is not an undesirable effect since two such pixels do not have any common boundary and can be considered as disconnected. Figure 6 shows the approximations to the shorelines extracted from the Landsat image after manual correction for skew.

5.2. A working system

The complete working system to register the Landsat image and the map functions as follows:

(a) Threshold the Landsat image using the value of the trough between the land and water peaks in the smoothed histogram of the gray-level distribution of the pixels in the image. Follow the contours of dark regions to find land-water boundaries.

(b) Eliminate as much of the skew in the shorelines from the Landsat image as possible. This is done manually by finding corresponding pairs of points in the Landsat image and a map (different from the one used in part (c)) and using them to estimate the parameters of an affine transformation which takes the

Landsat image to the map. A transformation of uniform scaling, rotation and translation still exists between the Landsat image and the map used in part (c).

(c) Choose only a few contours from the Landsat image and the map which are longer than all the other contours. The purpose of this is to improve the efficiency of the program but also to eliminate very low cost matches corresponding to very small contours which may not have any significance. From these contours keep only the ones which are closed.

(d) Compute the scale space images for the contours which were selected by the previous step and create a hierarchical representation based on that image.

(e) Match every model curve against every object curve and remember the cost of match for all those pairings. Since every curve considered in the matching process has a significant length, the lowest cost match should also be a correct one. Find such a match, compute the transformation parameters predicted by that match and choose a subset of all the matches which are consistent with that match, in that they predict roughly the same transformation parameters. Use a least squares method to estimate the parameters of an affine transformation from the Landsat image to the map using the data gathered from all the correct matches.

(f) Generate the transformed image of the Landsat shorelines and transform the original Landsat image if so desired.

6. Results and discussion

Figure 7 shows the map to which the Landsat image was to be registered. It should be noted that the transformation between the Landsat image and the map is a general affine transform. Since currently only rotation, uniform scaling and

translation can be handled by the matching algorithm, the skew in the Landsat image had to be corrected before the scale space images for its shorelines could be computed. Since a relatively small amount of skew still remains in the Landsat shorelines even after manual correction, it is necessary to compute the parameters of an affine transformation which takes the Landsat image to the map once a consistent subset of matches have been found. Figures 8 and 9 show two correct matches found by the matching algorithm and figure 10 shows the Landsat image registered to the map using the same matching algorithm. The goodness of match between the Landsat image and the map indicates that a small number of shorelines matched correctly can give a good estimate for the transformation parameters.

It was stated in section 5 that only closed curves were chosen for matching purposes. One reason for this is that the scale space images corresponding to closed curves are complete and it is preferable to use complete information when available. The other reason is a potential problem with the matching algorithm as described in section 4 when matching open or incomplete curves. It was stated that an initial cost is associated with every node when it is created which is a linear function of the difference in heights of the contour for which the node is created and the highest contour of the scale space image. This measure is to encourage matches corresponding to larger contours of the image and is reasonable when both images are complete since the correct match does correspond to the large contours of the image (usually the largest ones) but it can run into problems when there is incomplete information since a correct match might not correspond to one of the largest contours. The matching algorithm in its present form is good at finding shapes which have the same basic features but to overcome the problem associated with open curves, one can eliminate the necessity of attaching initial costs to nodes by matching all contours in the scale space image (including parents and ancestors and siblings of those) which should match as a result

of matching any two contours in the two images (not just the smaller siblings and sub-contours of them) and making sure that duplicate nodes are not created.

Since computing time is always an important constraint in every working computational vision system, the following is an evaluation of the computation time requirements of the various components of the system described in this paper:

The system was implemented in UBC's Laboratory for Computational Vision on a Vax[3] 11/780 with 4 megabytes of main memory running BSD 4.2 Unix[4]. Those parts of the system concerned with extracting shorelines from the Landsat image and computing the scale space images of the curves to be matched were implemented in C. The matching algorithm was implemented in Franz Lisp.

Tracking the zero-crossings in the scale space image can reduce the required computation time from hours to minutes for a typical curve. It took approximately one hour of CPU time to compute the scale space images of all the curves in the map and slightly less than that to compute those of the shorelines in the Landsat image. The matching algorithm took roughly 30 minutes of CPU time to estimate the transformation parameters. As mentioned earlier, only a subset of the scale space images were used in the matching process.

If real time response is expected from the system described in this chapter, its CPU requirements must be decreased further. Parallelism would significantly reduce the CPU requirements of the system described in this paper.

The whole scale space image of a curve is not more compact than the curve itself in general (this is obviously not true for convex or nearly convex curves). Therefore the program which computed the scale space image of various curves usually produced fairly large binary images but this should not be viewed as a shortcoming of the

[3] Vax is a trademark of Digital Equipment Corp.

scale space image since it has value as a representation for planar curves. Furthermore, the scale space image could be efficiently encoded.

Small changes in the shape of the curve usually result in small changes in its scale space image; however this may not seem to be true if a relatively small but highly convex feature is added to the curve. Such a feature may last through many levels of scale despite its relatively small size and therefore create a relatively large contour in the scale space image; but it can be argued that such a feature is easily distinguishable and hence a major feature of the curve. Figure 11 illustrates this.

Shorelines have quite arbitrary shapes and rich scale space images as a result, but the scale space image may not be a suitable representation for curves which consist of long straight segments or curves which are convex or nearly convex to begin with.

7. Extensions

The shapes of the contours in the scale space image indicate certain facts about the underlying features on the curve and can be studied further. This is useful if no explicit model curve is available but certain shapes (cues) are searched for on the curve. For example, a contour which is relatively large in the scale space image but has a small base (distance between the points where its left and right branches intersect the path-length axis of the scale space image), possibly indicates the presence of a highly convex feature which is small in size.

Scale space images can be obtained for planar curves which consist of more than one segment by finding a suitable parametrization. They can also be computed for curves which intersect themselves, but it should be noted that the scale space images for this class of curves will be fundamentally different. It is, for example, possible to obtain new inflection points at coarser levels of detail.

---

[4] Unix is a trademark of AT. & T. Bell Laboratories.

If it can be shown that the skewing of a planar curve translates to skewing of its scale space image then the matching algorithm can be modified to detect skew in a planar curve also. What is needed is to find at least one more pair of corresponding points on the contours in the scale space image (in addition to the peaks) in order to estimate the skew parameters.

Once the Landsat image is registered to the map, the results can be used to improve the initial segmentation of the Landsat image. An edge detector can be used to find the location of land-water boundaries at various levels of scale.

We have demonstrated that the method satisfies the necessary criteria (a)-(f). However, it does not satisfy criterion (g). The generalized scale space records only the sign of the curvature. Thus, for example, any two wholly convex shapes are indistinguishable to the method. Various possible extensions could be considered to cope with this problem. Using the locations of curvature maxima and minima at varying scales would reduce the size of the equivalence classes but would not eliminate the difficulty. A second extension would follow or combine scale space matching with image space matching based on a Euclidean distance. The third (and the most satisfactory) possible extension requires generalizing the scale space image again to record not just the sign of curvature but also its magnitude. The matching algorithm and its associated cost metric would have to be extended to cope with this.

Last but not least, it is possible to extend these results to three dimensions by obtaining the scale space image of a surface (which is three-dimensional) and using a generalized contour matcher.

8. Conclusions

We have posed the problem of scale-based description of planar curves, proposed a number of criteria for any solution method and described a method which

more nearly satisfies those criteria than do the other candidate methods. We have also described a matching algorithm which can optimally match two such shape descriptions.

9. Acknowledgements

The work described here was supported by the National Sciences and Engineering Research Council and the Canadian Institute for Advanced Research. We are grateful to Bob Woodham for his comments.

10. References

[1] P. V. C. Hough, Method and Means for Recognizing Complex Patterns, U.S. patent 3,069,654, 1962.

[2] R. O. Duda and P. E. Hart, Use of the Hough Transformation to Detect Lines and Curves in Pictures, *CACM, 15,* 1, 11-15, 1972.

[3] D. H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, *Pattern Recognition, 13,* 2, 111-122, 1981.

[4] F. O'Gorman and M. B. Clowes, Finding Picture Edges Through Collinearity of Feature Points, *Proc. 3rd Int. Joint Conf. Artificial Intell.,* Stanford, CA, 1973, pp. 543-555.

[5] A. K. Mackworth, On Reading Sketch Maps, *Proc. 5th Int. Joint Conf. Artificial Intell.,* Cambridge, MA, 1977, pp. 598-606.

[6] T. Pavlidis, *Structural Pattern Recognition,* Springer-Verlag, New York, 1977.

[7] L. S. Davis, Understanding Shape: Angles and Sides, *IEEE Transactions on Computers, C-26,* 3, 236-242, 1977.

[8] D. H. Ballard and C. M. Brown, *Computer Vision,* Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[9] D. D. Hoffman and W. A. Richards, Representing Smooth Plane Curves for Recognition: Implications for Figure-Ground Reversal, *Proc. National Conf. Artificial Intell.,* Pittsburgh, PA, 1982, pp. 5-8.

[10] J. L. Stansfield, Conclusions from the Commodity Expert Project, *MIT AI Memo 601,* 1980.

[11] A. P. Witkin, Scale space filtering, *Proc. 8th Int. Joint Conf. Artificial Intell.,* Karlsruhe, West Germany, 1983, pp. 1019-1022.

[12] A. K. Mackworth and F. Mokhtarian, Scale-based Description of Planar Curves, *Proc. 5th Canadian Soc. for Computational Studies of Intell.,* London, Ontario, May 1984, pp. 114-119.

[13] D. Marr and E. Hildreth, Theory of Edge Detection, *Proc. Royal Soc. London B* (207) 187-217, 1980.

[14] A. L. Yuille and T. Poggio, Scaling Theorems for Zero-Crossings, *MIT AI Memo 722,* 1983.

[15] A. L. Yuille and T. Poggio, Fingerprint Theorems for Zero-Crossings, *MIT AI Memo 730,* 1983.

[16] N. J. Nilsson, *Problem-Solving Methods in Artificial Intelligence,* McGraw-Hill, New York, 1971.

[17] N. J. Nilsson, *Principles of Artificial Intelligence,* Tioga Publishing Company, Palo Alto, CA, 1980.

[18] J. Glicksman, A Cooperative Scheme for Image Understanding Using Multiple Sources of Information, Ph.D. Thesis, Dept. of Computer Science, Univ. of British Columbia, 1982.

[19] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis,* John Wiley & Sons, New York, 1973.

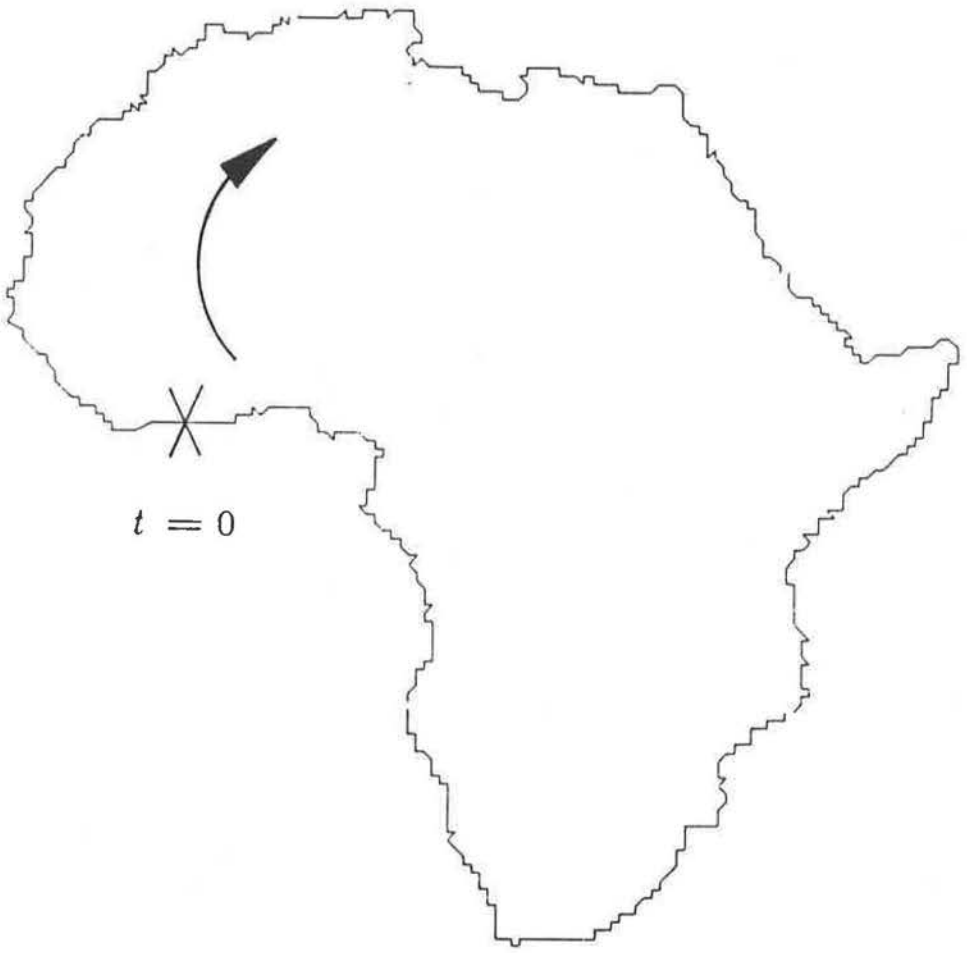
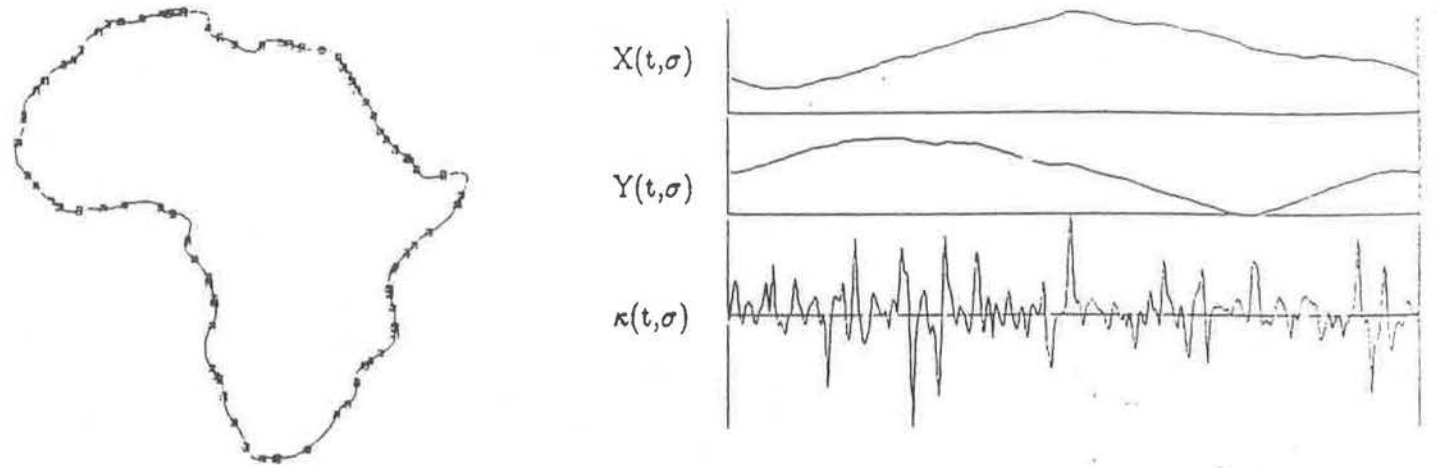
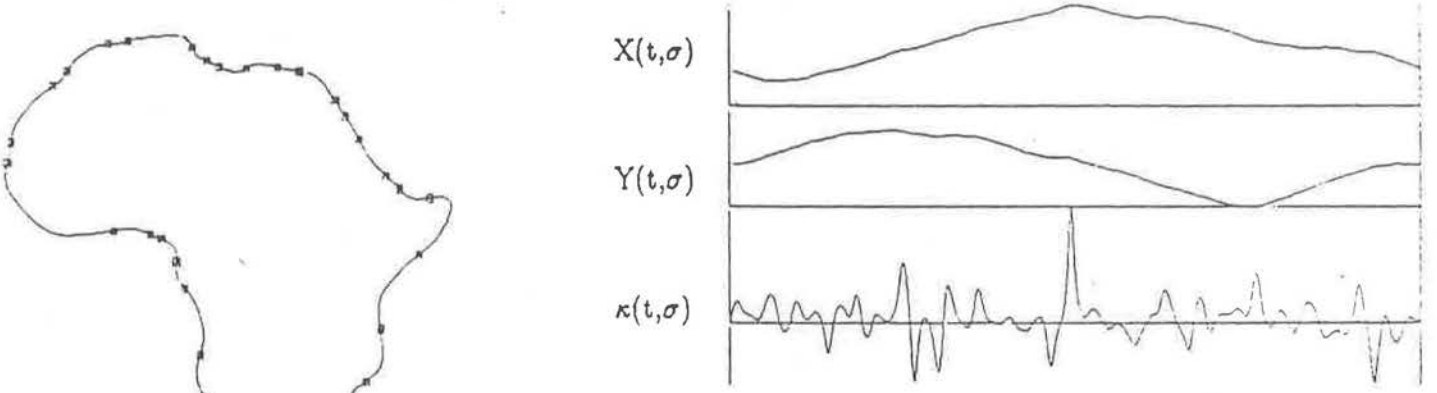


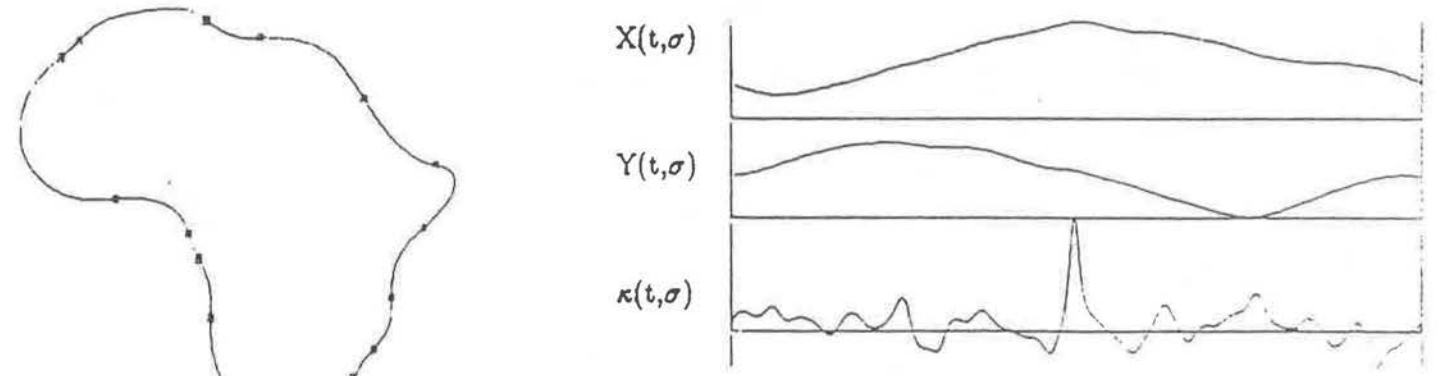
Figure 1. Shoreline of Africa



(a) $\sigma = 4$

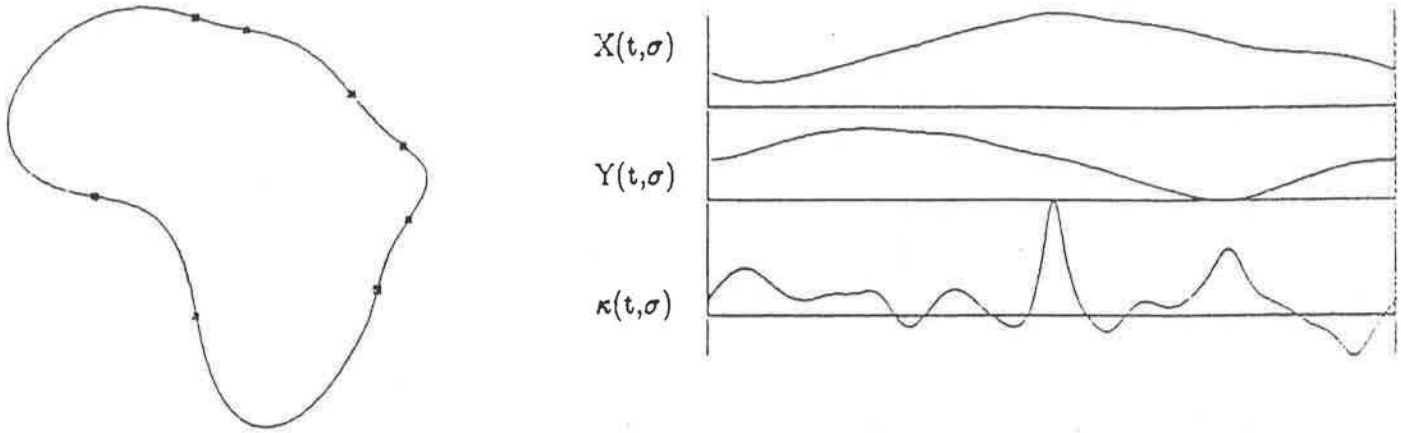


(b) $\sigma = 8$

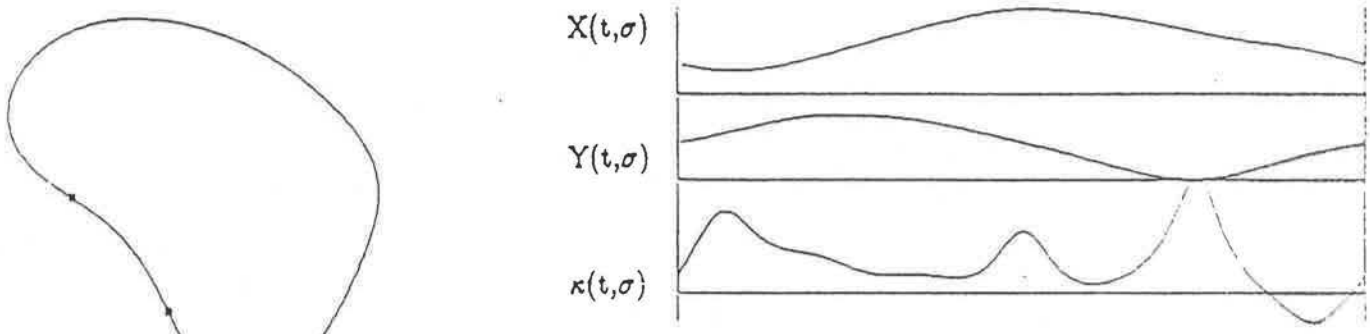


(c) $\sigma = 16$

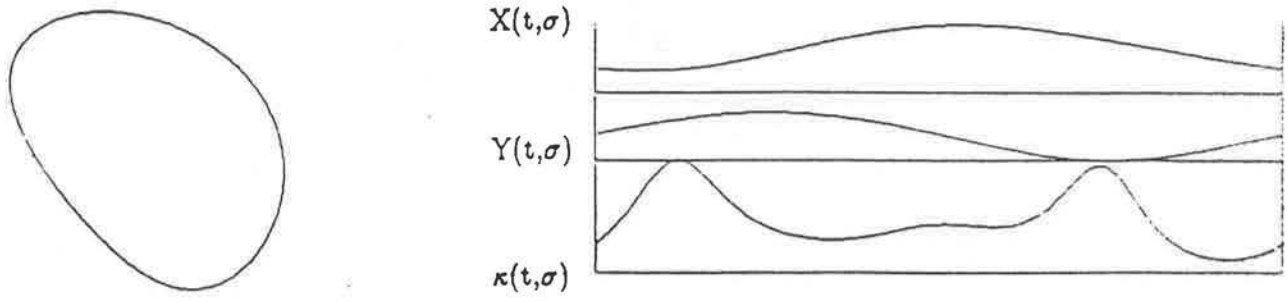
Figure 2. Smoothing a Curve: Scale-based effects.



(d) $\sigma = 32$



(e) $\sigma = 64$



(f) $\sigma = 128$

Figure 2. (Continued) Smoothing a Curve: Scale-based effects.

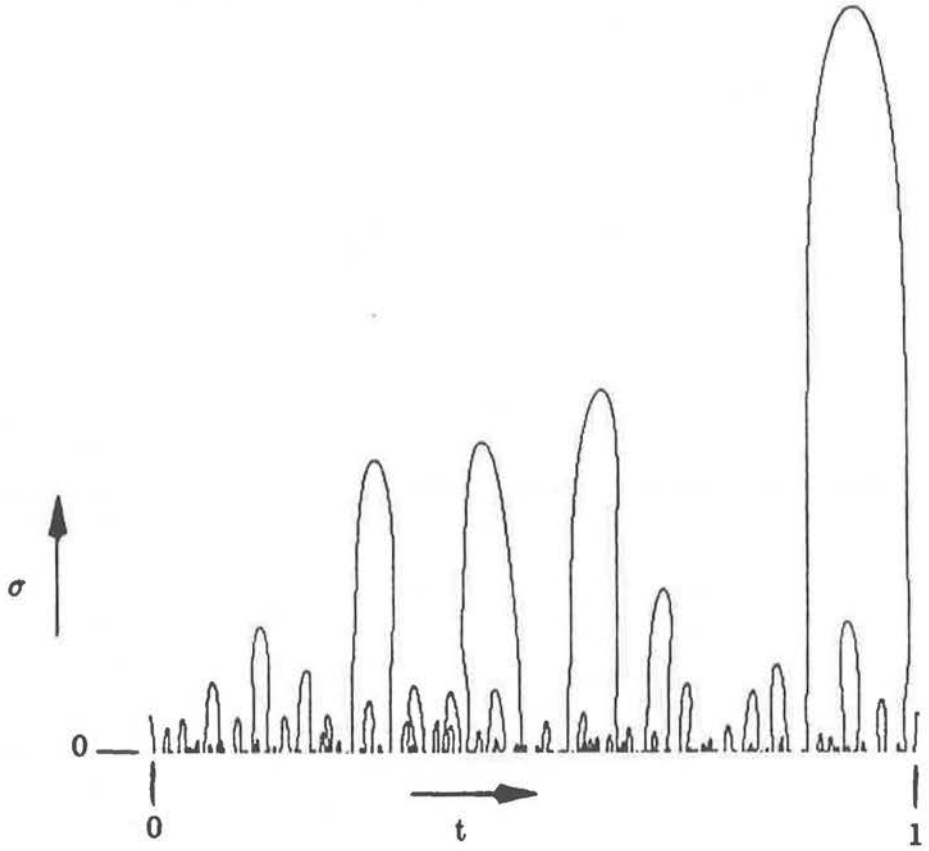


Figure 3. Generalized Scale Space image of Africa

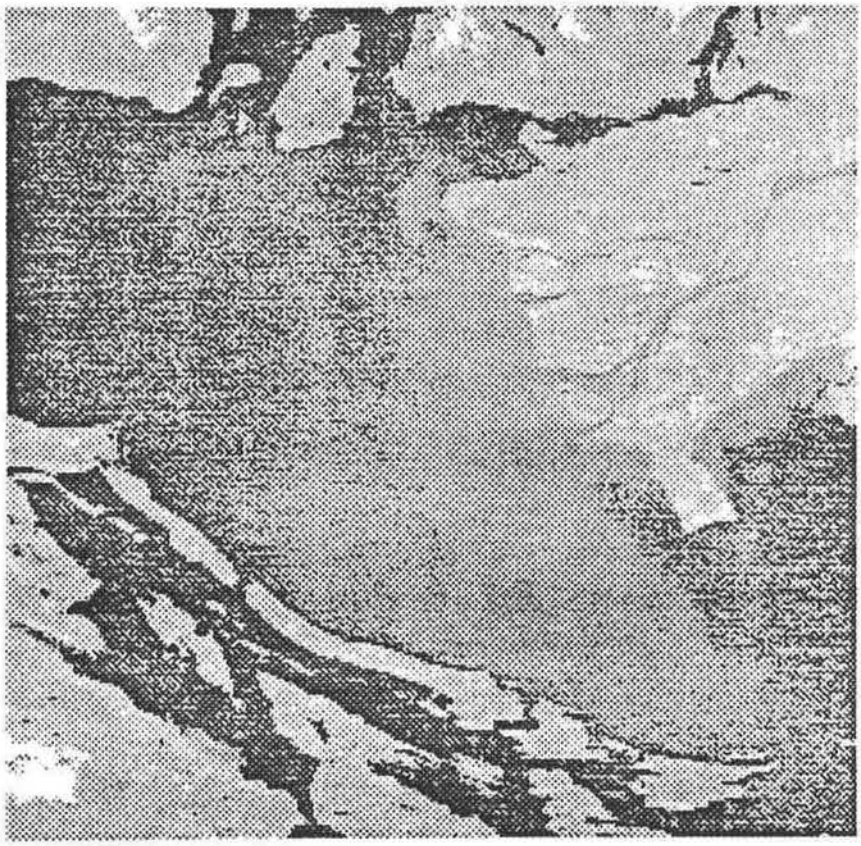
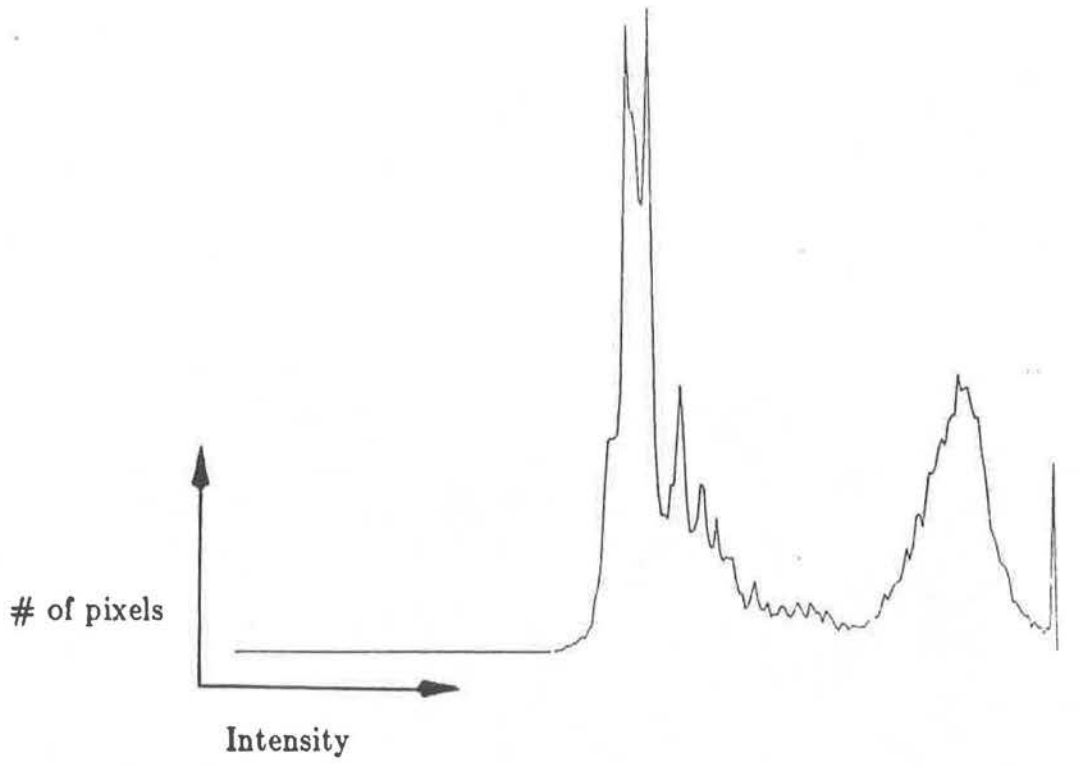
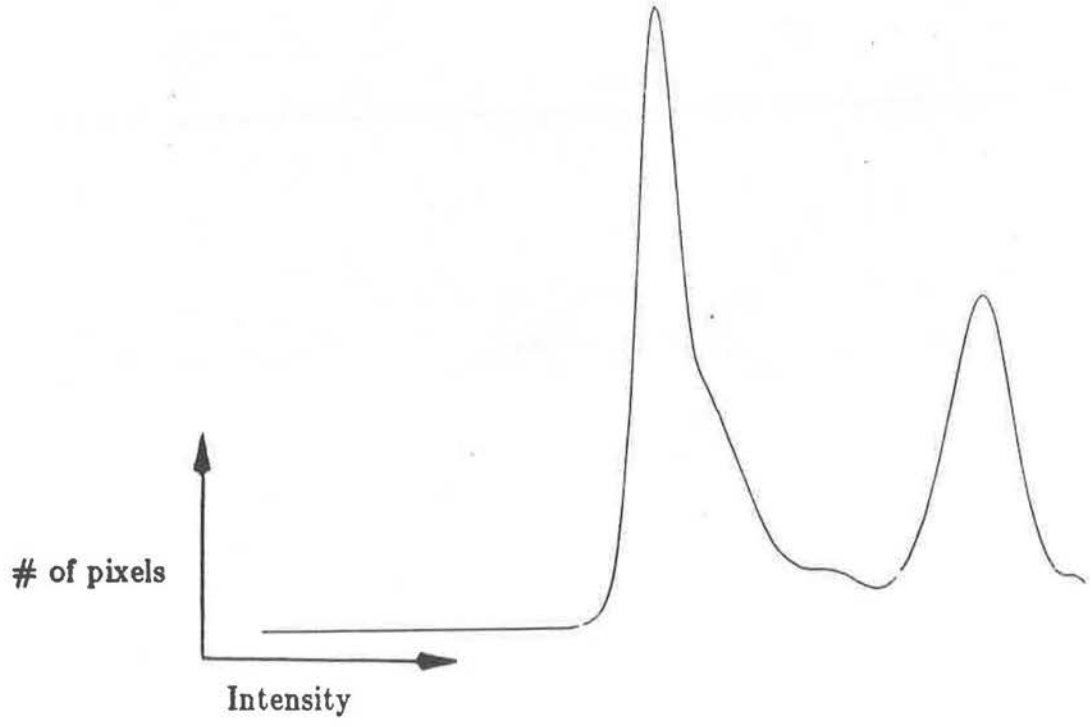


Figure 4. Band 7 Landsat image of part of the Strait of Georgia, B.C.



(a) Original histogram

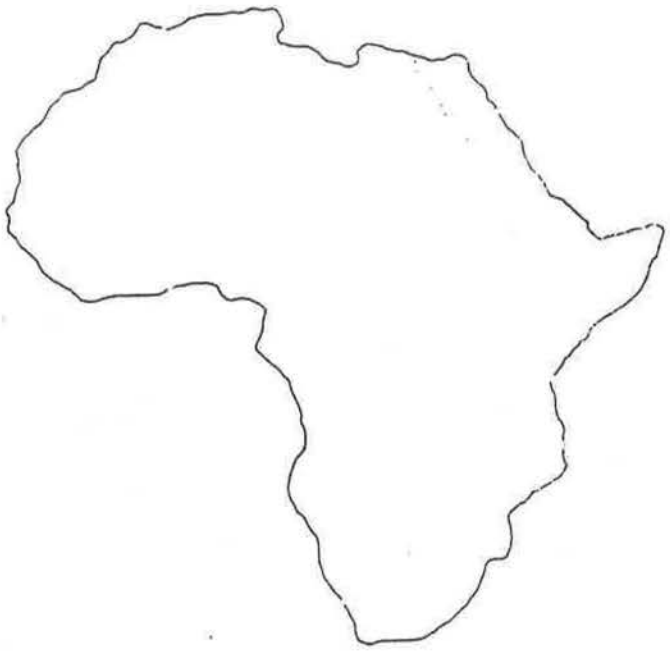


(b) Smoothed histogram with two peaks

Figure 5. Gray-level distribution of pixels in Landsat image.

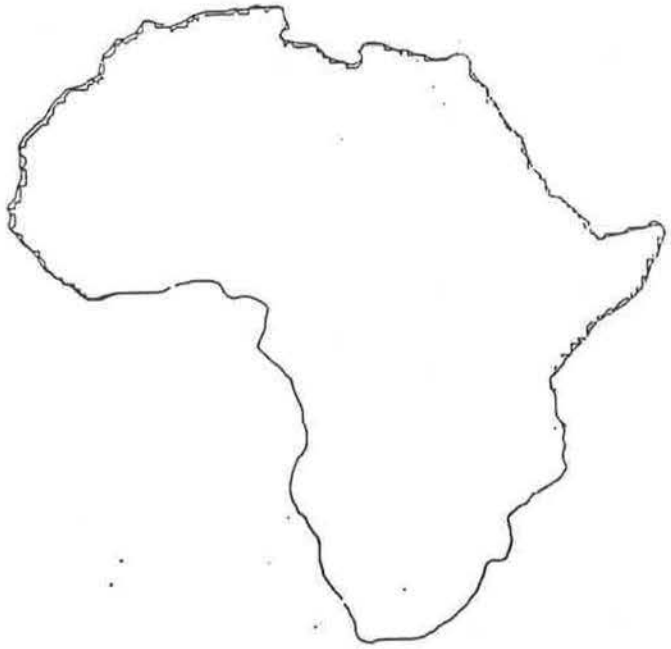
Figure 6. Shorelines from the Landsat image after correction for skew

Figure 7. Map of part of the Strait of Georgia, B.C.



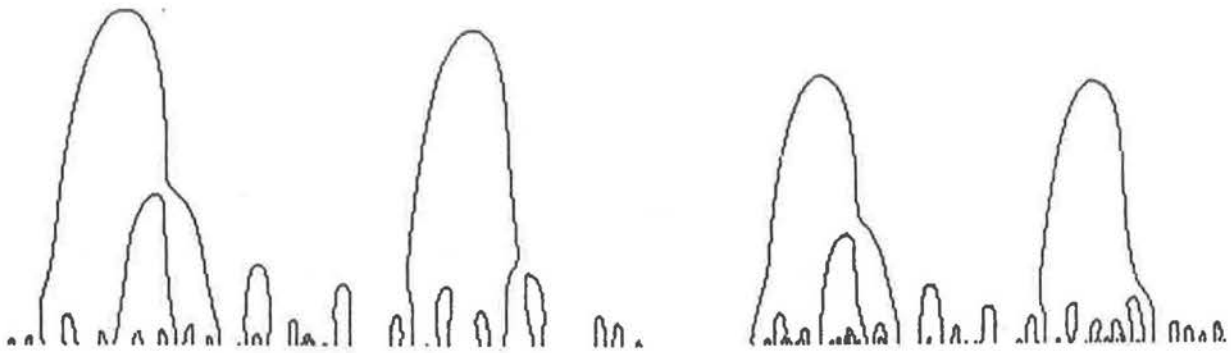
(a) Complete shoreline of Africa

(b) Part of the same shoreline

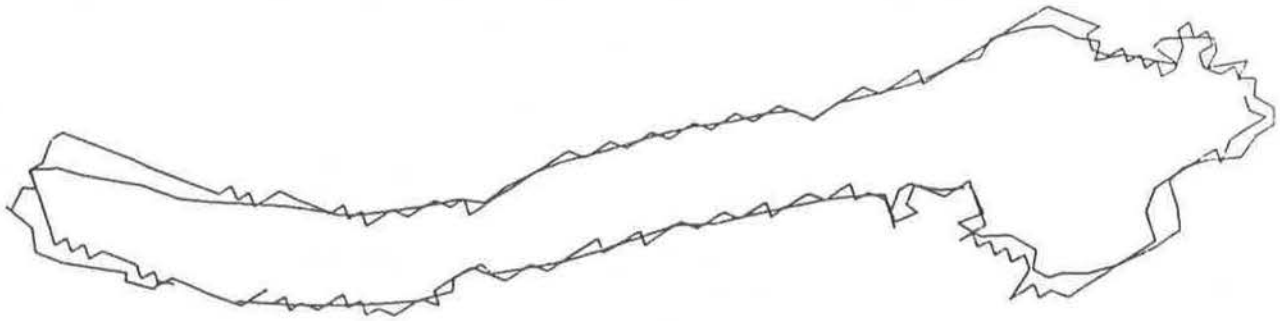


(c) Result of the match

Figure 8. Matching shorelines of Africa.



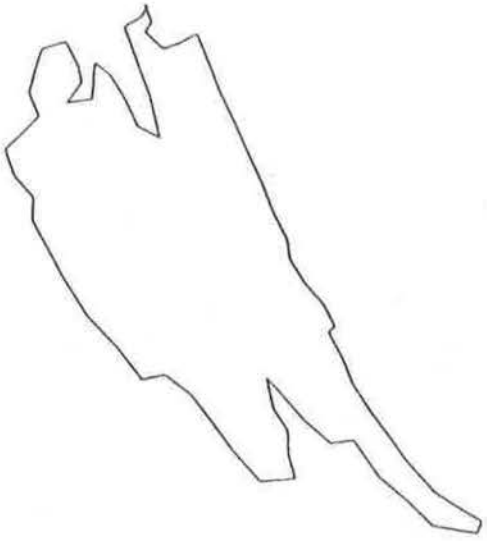
(a) Scale space image of map island (b) Scale space image of Landsat island



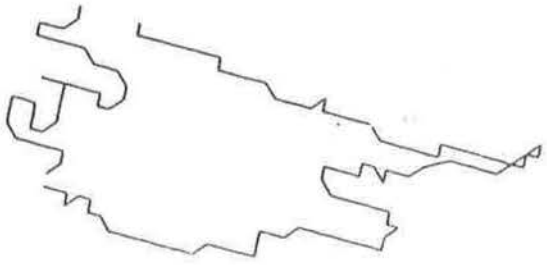
(c) Result of the match

Figure 9. Matching islands from the map and the Landsat image.

Figure 10. Registration of the Landsat image to the map.



(a) Island from the map



(b) Island from the Landsat image

(c) Scale space image of (a)

(d) Scale space image of (b)

Figure 11. Effects of features on the scale space image.