Extraction of Shoreline Features by Neural Nets and Image Processing

T. W. Ryan, P.J. Sementilli, P. Yuen, and *B. R. Hunt*
Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721

ABSTRACT: This paper demonstrates the capability of using neural networks as a tool for delineation of shorelines. The neural nets that we use are multi-layer perceptrons, i.e., feed-forward nets with one or more layers of nodes between the input and output nodes. The back-propagation learning algorithm is used as the adaptation rule. All the images are preprocessed before training set selection. The preprocessing operation is done by normalizing each picture in order to obtain grey level statistics that do not vary significantly from scene to scene. A neural net is then trained to categorize small blocks of image data as land or water. After a category map has been generated by the neural net, image processing techniques are used in order to delineate the shoreline down to the pixel level. A majority of the misclassified regions of the binary land/water category map are corrected by applying connected components labeling followed by the merging of small isolated regions into the surrounding area. Finally, a sequence of image processing operations are used to detect edges, reject isolated edge points, and join small edge gaps to produce a continuous shoreline delineation.

INTRODUCTION

A PRINCIPAL ACTIVITY OF CARTOGRAPHY for many years has been the application of human interpretive skills to compile a map manuscript from an aerial photograph. This process of compilation can be characterized by the activities of feature delineation, feature identification, and feature attribution. In delineation the cartographer uses a suitable graphic mechanism (usually a pencil tracing on vellum overlaying the aerial photograph) to create a representation of the feature. In feature identification and attribution the cartographer must code the feature characteristics according to a defined glossary of feature properties.

Of the activities of delineation, identification, and attribution, it is delineation which represents the greatest source of frustration and fatigue. Detailed tracing of the border of a feature is not usually considered a challenging activity to an image analyst. On the other hand, feature identification and attribution usually require substantial skill to properly interpret the visible aspects of the feature in an aerial photograph. Most cartographers would appreciate the opportunity to reduce the delineation burden so that they could devote more time to the subtle interpretive tasks that require the maximum level of their professional skills. Thus, it is not surprising to find the infusion of new computer technology into the process of feature delineation.

New cartographic workstations that have come into the marketplace have emphasized the use of digital graphics for feature delineation. Instead of tracing on vellum paper, the typical workstation provides a variety of graphic tools to simplify the task of feature delineation. Although these tools enhance productivity, they do not reduce the tedium of tracing features. Paper may be replaced by electronics in these systems, but the primary burden for the cartographer remains the same: convert image data into feature delineations. The next step in improving the environment for the cartographer will be to employ technology that can decrease the burden of feature delineation.

BACKGROUND

Previous attempts to automate cartographic feature extraction have led to a "toolbox" of algorithms (Nagao and Matsuyama, 1980; Ballard and Brown, 1982). Many of these algorithms depend, at some level, on the classification of regions of image data into categories that provide a means to make decisions regarding the presence or absence of edges, boundaries, textures, shapes, or other significant patterns.[1] There has been little effort, however, to provide a unified algorithmic foundation for feature extraction and delineation. One notable exception is the attempt by McKeown (1984) to develop feature extraction methodologies based on the creation of a *scene model* which is gradually built up using a combination of image processing and expert systems. McKeown has focused on the delineation of features in airport scenes. These features (e.g., hangars, runways, terminal buildings) exhibit strong associations, growing out of functional constraints, which can be exploited to enhance extraction accuracy. Unfortunately, most cartographic features do not exhibit strong associations with neighboring features. Furthermore, both terrain and man-made features exhibit extreme variability in their appearance due to variations in acquisition geometry, illumination conditions, masking, natural variations in feature composition, as well as construction techniques and designs. Although scene context is important, it is our view that feature extraction techniques must take maximal advantage of available pattern recognition technology at the lowest levels of processing and must eventually provide a capability that is largely invariant with respect to many of the sources of feature variation.

The resurgence of interest in connectionist approaches to pattern recognition has resulted in the development of several "neural network" algorithms which have significantly advanced the state-of-the-art in pattern analysis systems. Hepner *et al.* (1990) applied neural nets to the problem of land-cover classification from multispectral data. Their results suggest that the neural network approach is better than conventional classification techniques, especially in situations involving small training sets. Other studies (e.g., Huang and Lippman, 1987) have shown that neural network performance exceeds that of conventional classifiers (k-nearest neighbor, quadratic Gaussian) when the underlying data distributions do not conform to prior assumptions (usually Gaussian models). Even in cases where the performance is comparable, the neural network approach provides the distinct advantage of simplifying the training and testing processes. Our approach is to provide a feature extraction environment in which neural net architectures can be easily and rapidly configured and trained to perform desired categorizations.

[1]These patterns are often referred to as *features* in the pattern recognition literature. We will use the term *patterns* or *measures* to eliminate confusion with cartographic *features*.

PHOTOGRAMMETRIC ENGINEERING & REMOTE SENSING,
Vol. 57, No. 7, July 1991, pp. 947–955.

0099-1112/91/5707–947$03.00/0

OBJECTIVE

The specific objective addressed in this study is the automated delineation of shorelines. This feature category was selected as representative of the class of "line" features. The manual (or interactive) delineation of these features by an extraction specialist is a tedious, time consuming process. Automation of shoreline delineation provides a useful example of how pattern recognition technology can be brought to bear on the more general feature delineation problem. Because a shoreline is, by definition, a boundary between a land area and a water area, it presents a relatively straightforward application of pattern classification. The objective of this study, therefore, is to demonstrate that neural networks, in conjunction with image processing techniques, provide an effective, efficient methodology for creating accurate shoreline delineations.

GENERAL APPROACH

For this study, our approach is based on the use of trainable neural networks for low-level categorization of data into land and water categories. Although this categorization is specific to the shoreline delineation problem, the neural network methodology is capable of treating many different pattern analysis problems. For the shoreline delineation problem, the land/water categorization is done at a coarse resolution relative to the desired accuracy of the final delineation. This categorization creates a binary category map which is refined down to the pixel level to provide an estimate of shoreline location. The actual delineation, i.e., the drawing of lines and curves to graphically represent the shoreline feature, draws on the technology of image and graphics processing.

Description of Neural Networks

Artificial neural networks[2] are a class of computational models inspired by our present understanding of biological nervous systems. They have found a variety of diverse applications such as industrial inspection (Glover, 1988), credit rating (Dutta and Shekhar, 1988), character recognition (Guyon *et al.*, 1989), speech recognition (Lippman and Gold, 1987), image segmentation (Cortes and Hertz, 1989), and robotic control (Kuperstein and Rubinstein, 1989).

Neural net models are composed of many simple processing elements (PEs) connected by links with variable weights. The simplest and most common PE sums a set of weighted inputs and passes the sum through a non-linearity (Figure 1). The weights w_1, w_2..., w_N and the internal bias parameter θ are adjusted to improve system performance. The most common non-linearity is the *sigmoid* function (Figure 2) which limits the range of output values. The non-linearity is critical to network performance: it has been shown that a two-layer network (Figure 3) is capable of realizing *any* functional mapping from input to output (Funahashi, 1989; Hecht-Nielsen, 1989). This is not possible in a linear system.

Neural network models are specified by the PE characteristics, the learning rules, the network interconnection geometry, and dimensionality (Lippman, 1987). Although these topics continue to be active areas of research, several approaches have emerged which have definite applicability to complex pattern analysis problems (Pao, 1989).

The potential benefits of neural nets include the high computation rates provided by massive parallelism and greater fault tolerance than can be achieved by serial computations. Damage to a few processing elements or interconnection links in a parallel implementation does not necessarily impair system performance. In many applications, however, neural nets are simulated on a serial computer—PEs time-share the CPU. The primary advantage of a neural net approach in this case is the ease with which a variety of network specifications can be implemented, trained, and tested in response to knowledge gained by observing network performance. Secondly, the family of neural nets provides a much richer set of classification mappings than can be obtained by traditional means. Finally, the operational mode of a network is a functional subset of the training mode. Thus, the architectures allow an extremely convenient environment for rapid development of pattern recognition systems which bypasses the cumbersome data analysis process required in traditional pattern recognition technology.

Virtually all types of neural network architectures can be viewed as adaptive filters which transform a multidimensional (input) pattern or vector $\mathbf{x}=[x_1x_2 \ldots x_N]$ into another (output) pattern $\mathbf{y}=[y_1y_2 \ldots y_M]$. The mapping is determined by a set of parameters $\mathbf{p}=[P_1P_2 \ldots P_L]$ which can be modified in response to input and output patterns. This transformation from data to data is often referred to as an *associative memory* (y is "associated" with $\mathbf{x}$). For some applications (e.g., pattern classification) the output

[2]Also referred to as neural nets, parallel distributed processing models, connectionist systems, or neuromorphic systems.

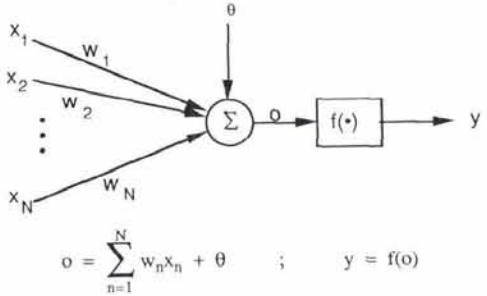


FIG. 1. Simple processing element in a neural network architecture. The inputs {x} are *filtered* by the weights {w} and passed through the non-linearity f(). The quantity θ provides an adjustable bias.

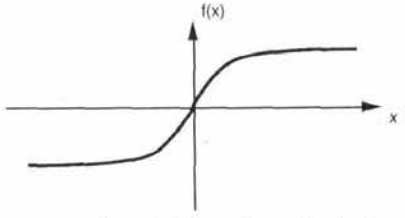


FIG. 2. Sigmoid function. Typically, f(x) = tanh(x).

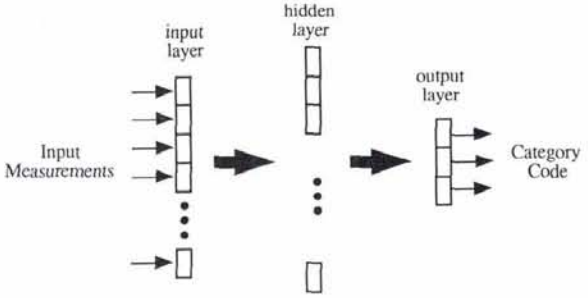


FIG. 3. Architecture of the network. Each cell represents a processing element (as depicted in Figure 1). The black arrows represent weight matrices.

pattern y is interpreted as a *category code* which assigns the input pattern x to a specific pattern class. For other applications (e.g., pattern correction or image restoration), y is interpreted as a *prototype* for a pattern class, i.e., y is a typical pattern from the pattern class.

Neural nets can be categorized in a variety of ways:

(a) dynamic versus static (Hopfield and Tank, 1985). A dynamic network solves a set of coupled nonlinear differential or difference equations, while a static net simply passes the data through a series of mappings to produce an output pattern.
(b) supervised versus unsupervised (Lippman, 1987). A supervised network is designed with a set of desired associations $\{(x_1,y_1),(x_2,y_2),\ldots,(x_K,y_K)\}$ and the design process is to build a network that can reproduce these associations in response to an input vector x. Unsupervised systems, however, perform clustering on a set of training samples, where the clusters are usually unknown to the system prior to training.
(c) autoassociative versus heteroassociative (Rumelhart and McClelland, 1986). If the desired associations (x_i,y_i) are such that $y_i=x_i$ for all i, then the memory is termed autoassociative. If $y_i \neq x_i$, the memory is heteroassociative. In either case, the objective is to devise a system which is robust with respect to errors in the input patterns.

Multi-layer perceptrons are used as the neural network architecture for this study. They are feed-forward static, supervised, heteroassociative nets with one or more layers of nodes between the input and output nodes. The adaptation rule which we used is the back-propagation learning algorithm with momentum (Werbos, 1974; Parker, 1986)[3]

Processing Overview

The specific approach taken for shoreline delineation involves four steps:

(1) The image dynamic range is standardized by histogram remapping and by performing median filtering to reduce interference from noise.
(2) The image is partitioned into small (8 by 8) non-overlapping blocks. Each block is classified into *land* or *water* (and possibly *shoreline*) categories. This categorization is based on local rotation invariant texture measures and is implemented with the neural network. The categorization map is then segmented and a region merging algorithm is used to eliminate small isolated areas.
(3) Using the adjusted categorization map as a guide, edge detection is applied to identify shoreline candidate points.
(4) Image processing algorithms are employed to reject isolated edge points and join small (single-pixel) edge gaps. Finally, gaps of medium length ($\leq$ 5 pixels) are filled by interpolation.

The output of this process, as developed under this study, is a set of delineations overlaid on image data. In an operational system, these delineations would most likely be converted to a vector format and stored in a feature database for review by an analyst at his or her convenience. Step four occasionally leaves gaps in the shoreline delineation. These gaps result from either poor visibility of the shoreline or an unfortunate spatial orientation of the shoreline relative to the coarse shoreline estimate. Recommendations will be presented for correcting this deficiency.

Two kinds of errors will occur in any automated delineation system: errors of *omission* (gaps) and errors of *addition* (extraneous edge points). Given that delineations will eventually be reviewed by a human analyst with editing tools, we believe it is preferable to minimize the *addition* errors, i.e., editing should consist primarily of filling in omissions. The current algorithm does this by selecting candidate shoreline points displaying a sufficiently strong edge at the land/water transition.

[3]Also called the *Widrow-Hoff rule*, the *LMS* (least mean square) *algorithm*, and the *delta rule*.

PROCESSING ALGORITHMS

The four processing steps described above are depicted in Figure 4.

Step 1: Image Standardization

In order to improve the robustness of subsequent algorithms, we found it was beneficial to apply normalization techniques to the input image data. The purpose of normalization is to provide the land/water classifier with data whose gray level statistics do not vary significantly from scene to scene. Not unexpectedly, the neural network classifier learns to categorize dark, relatively uniform textures as water. It is important, therefore, that water be remapped to a dark gray tone. This remapping, however, must be done automatically with no *a-priori* information regarding the locations of regions actually occupied by water. An alternative to normalization would be to train the network using an extremely large sample of images spanning the set of expected acquisition conditions. Even then, there would likely be increased confusion between land and water categories. The normalization approach avoids the difficulty of acquiring and training on unwieldy quantities of image data.

Normalization was implemented with a linear stretch with a fixed fraction of saturated pixels at each end of the gray scale. Given an image $I(x,y)$, we compute the cumulative distribution $C(i)$, $i=0,\ldots,K$ where K is the maximum allowable gray level (typically 255 for 8-bit data). Let g_L be the highest gray level for which $C(g_L)<r$ and g_H be the lowest gray level for which $C(g_H)>1-r$ where r is a fixed parameter ($r=0.02$ was used in this study). The relationship between the input image $I(x,y)$ and the output image, $O(x,y)$, is given by

$$O(x,y) = \max\left[0, \min\left(255, 255\,\frac{I(x,y) - g_L}{g_H - g_L}\right)\right]$$

Following the normalization remapping, $O(x,y)$ was passed through a 3 by 3 median filter (Ballard and Brown, 1982) to reduce noise effects. The median filter was found to be appropriate for the class of data that was available for this study. The data were digitized from 1:50,000-scale USGS panchromatic transparencies at a ground sample interval of 2.5 metres and which contained a considerable amount of film grain noise. The median filter effectively reduced the noise without seriously affecting the edge quality.

Step 2: Coarse Categorization of Land and Water

In Step 2, the image data are accessed sequentially by blocks (e.g., 8 by 8). Texture measurements are extracted from a block and fed to a neural network classifier which places each block in the *land* or *water* category. The result of this process is a *Category Map* (Figure 5) which depicts the block memberships in an image format. The transition between categories provides a coarse estimate of the shoreline location.

The texture measures used by the classifier are currently obtained from the power spectrum of the N by N block of data.

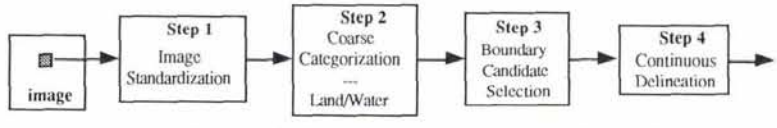


FIG. 4. Shoreline delineation overview. See text for details.

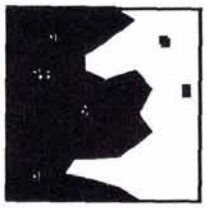


FIG. 5. Land/water category map. The map is a binary image in which each pixel represents an N by N block of data. White pixels in the black area and black pixels in the white area represent misclassified blocks.

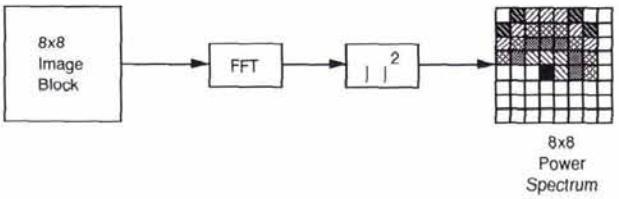


FIG. 6. Calculation of power spectral rings. Values of the spectrum are summed over the regions of similar shading. The black point in the center of the array is the DC coefficient. The coefficients in the lower half of the array are redundant. This arrangement produces six ring measurements. Any subset of these measurements can be used as features in the neural net classifier.

In order to achieve rotation invariance, the power spectrum values are integrated within bands of (roughly) constant spatial frequency (Figure 6).

We experimented with an alternative measurement set comprised of histogram bin populations. Reasonably good results were achieved by using the lowest five bins from a uniformly distributed 10-bin histogram taken over the N by N block. Visual inspection showed that the resulting category maps were not quite as good, however, as those obtained with the Fourier spectral rings. We also found that equivalent categorization results could be achieved by using only the *first three* rings (i.e., the low spatial frequency rings) instead of all six.

The neural network classifier is *trained* by creating a training set of measurement data with known (desired) category assignments. Given the vector of measurements at the input to the classifier, it produces an output category *estimate* which is compared with the desired output category to create an error signal. The error signal is then used to update the parameters of the network using the generalized delta rule learning algorithm. A description of this algorithm is found in Rumelhart and McClelland (1986). It should be noted that there are a variety of alternative learning algorithms which are purported to achieve much higher learning rates for some problem situations (Parker, 1986; Watrous, 1987; Vogle *et al.*, 1988; Fahlman and Lebiere, 1990). For this study, however, training time was not a critical issue. We found that standard backpropagation with momentum converged to a solution in a reasonable amount of time. The architecture of the network consists of an **input layer** which simply buffers the measurement vector, a **hidden layer** of nodes (simulated processing elements), and an **output layer** which generates category codes (Figure 3). Because there are only two categories for this application, the output codes are $(1,-1)$ for land and $(-1,1)$ for water. We have experimented with a three-category classifier using *shoreline* as the third category but have found that a merger of the land and shoreline categories provides simpler subsequent processing.

The neural network classifier, as with any statistical classifier, occasionally makes categorization errors. For the land/water categorization, these errors can be corrected, in most cases, with simple image processing algorithms applied to the binary land/water category map.

Three algorithms have been empirically investigated:

(1) the morphological operations of *erosion* followed by *dilation* produce the *closing* operator and eliminate small regions but maintain the boundaries of large areas (Sternberg and Serra, 1986). Erosion can be thought of as peeling away a layer of pixels from the boundary of a region whereas dilation adds pixels to the boundary. For binary data, erosion is implemented with a local *minimum* filter whereas dilation can be implemented with a local *maximum* filter.

(2) the *voting* filter replaces the center pixel of a K by K window with the binary value most often represented in the window, i.e., a *majority* filter. Our version of the algorithm is modified to give additional votes to the center pixel in the window.

(3) *connected components labeling* followed by *merging* of small regions.

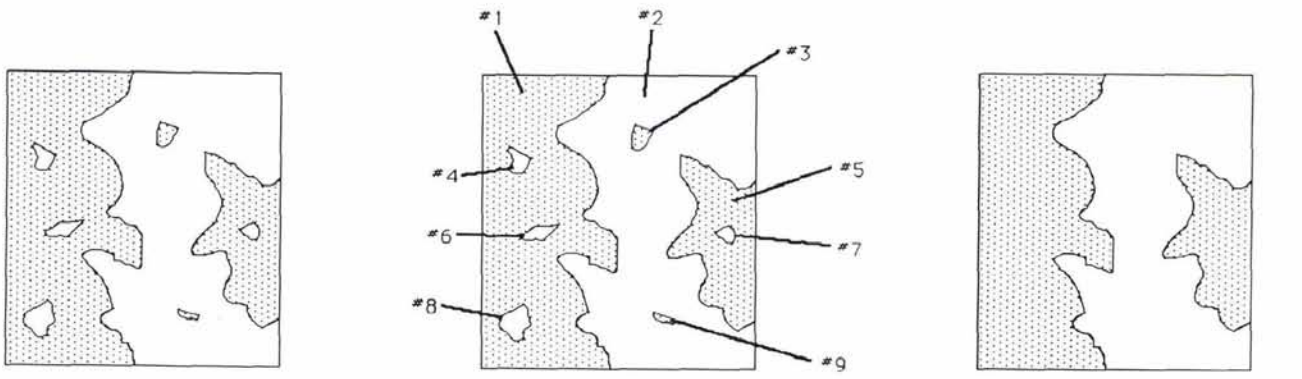
The best performance was achieved with algorithm 3: connected components labeling. Algorithms one and two both produced distortions in the shape of the regions defined by the category map.

Connected components labeling is a well known segmentation technique which is based on the search for contiguous image regions with homogeneous (but not necessarily uniform) gray tone (Ballard and Brown, 1982; Rosenfeld and Kak, 1976). The labeling algorithm scans the category map and assigns a different *label* (an integer) to each pixel within each contiguous patch and accumulates the total number of pixels in each labeled region (Figure 7). This pixel count is proportional to the area of the contiguous region. The labeling algorithm is then followed by a *merging* operation which compares each region area with a selected area threshold. If the area is smaller than the threshold, then the gray level of that region (0 or 1 for a binary category map) is simply toggled. In this way, all small misclassified regions are corrected.

STEP 3: SELECTION OF CANDIDATE SHORELINE POINTS

Following the automated correction of erroneous blocks, the land/water boundary is converted to a shoreline *strip*. This strip defines a region that completely encloses and coarsely approximates the shoreline. Conceptually, the strip is created by pushing the land/water boundary one (or more) block(s) inland and then adding this new boundary to the original boundary. This has been implemented by applying a local maximum filter to the corrected category map, and then adding the filtered map to the corrected map. The local maximum filter "dilates" the water region and contracts the land region. The size of the filter determines the width of the strip e.g., a 3 by 3 filter results in a single block strip width, a 5 by 5 window results in a two block strip width. The process used to create the strip map is illustrated in Figure 8. The result of this processing is a category map in which land blocks are coded as zero, shoreline blocks are coded as one, and water blocks are coded as two. Subsequent processing is restricted to a region in the neighborhood of the shoreline strip, thereby enhancing processing efficiency and reducing potential confusion.

Candidate shoreline pixels are extracted from the standardized image by applying a one-dimensional convolution filter horizontally or vertically over the shoreline region, depending on the orientation of the strip map. The output of this convolution is a smoothed differential of neighboring grey levels along a line or column of the image. Figure 9 illustrates an image with



(a) Original category map (b) Labelled category map (c) Category map after small region merging

FIG. 7. Correction of misclassified regions using connected components labelling

the strip map superimposed. The filter is applied from each boundary of the strip toward the opposite boundary. We selected a 16-pixel window over which to apply the filter. Therefore, if the strip is two pixels wide in the strip map, the filter is applied over the region enclosed by the strip. If a single pixel strip is used, then the filter is applied over the strip region and over eight pixels (one *block*) on each side of the strip region.

We assume that the land/water interface is characterized by a step edge, i.e., an abrupt change in grey level at the shoreline. The land and water classification contained in the strip map is used to determine the expected polarity (positive/negative) of the step edge at the land/water transition (Figure 10). After the convolution is performed over a given line (or column) of the standardized image, the location in that line (column) for which the edge magnitude is maximum with respect to the expected polarity is identified. If this maximum edge magnitude exceeds a fixed threshold, then the corresponding point in the image is selected as a candidate edge point. As the convolution is applied over the shoreline region, a new binary image is created, of the same size as the original image, in which each candidate shoreline point is coded as a one.

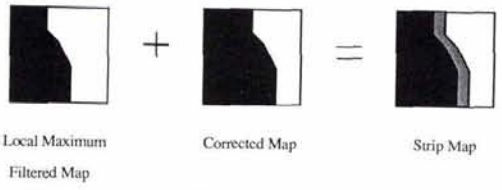


FIG. 8. Creation of the strip map.

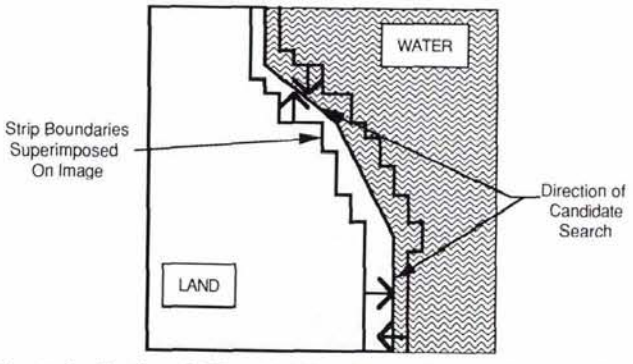


FIG. 9. Application of differential filter for candidate point extraction

The algorithm used to identify candidate edge points worked well for all images tested. The following components of the algorithm were determined empirically:

- Window Size
- Convolution Mask
- Threshold Level

Window Size. The size of the convolution filter was varied during testing of the algorithm in order to find the best trade-off between efficiency and performance. Initially, we used a simple two-point difference to detect edge points. This approach resulted in a high number of false points due to the texture contained near the shoreline. We increased the window size to four points (i.e., weights of 1, 1, −1, −1), and the performance improved, although there were still a relatively large number of spurious candidates. Lengthening the window to six points provided good results, and further increases in the window size beyond six points showed little or no performance improvement.

Convolution Mask. We experimented with two different types of weighting functions. Once we selected a window size of six, we experimented with tapered weighting of the samples, e.g., 1/3, 2/3, 1, −1, −2/3, −1/3. No noticeable performance improvement was realized with this weighting. Secondly, we weighted the filter outputs based on their proximity to the coarsely delineated land/water interface (filter outputs closer to the interface were weighted more heavily). This was done because we found that most false candidates were over land, with very few over water. As with the filter coefficients, we found no

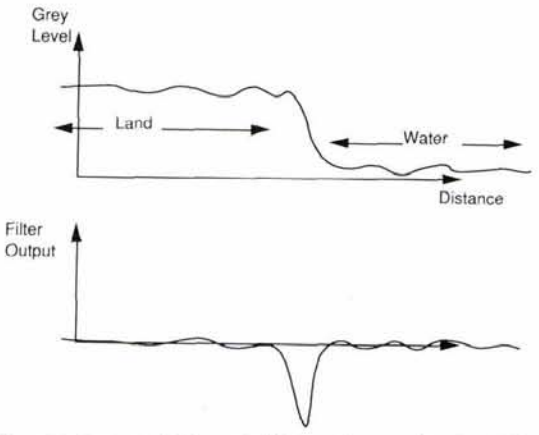


FIG. 10. Output of differential filter with scan direction left to right.

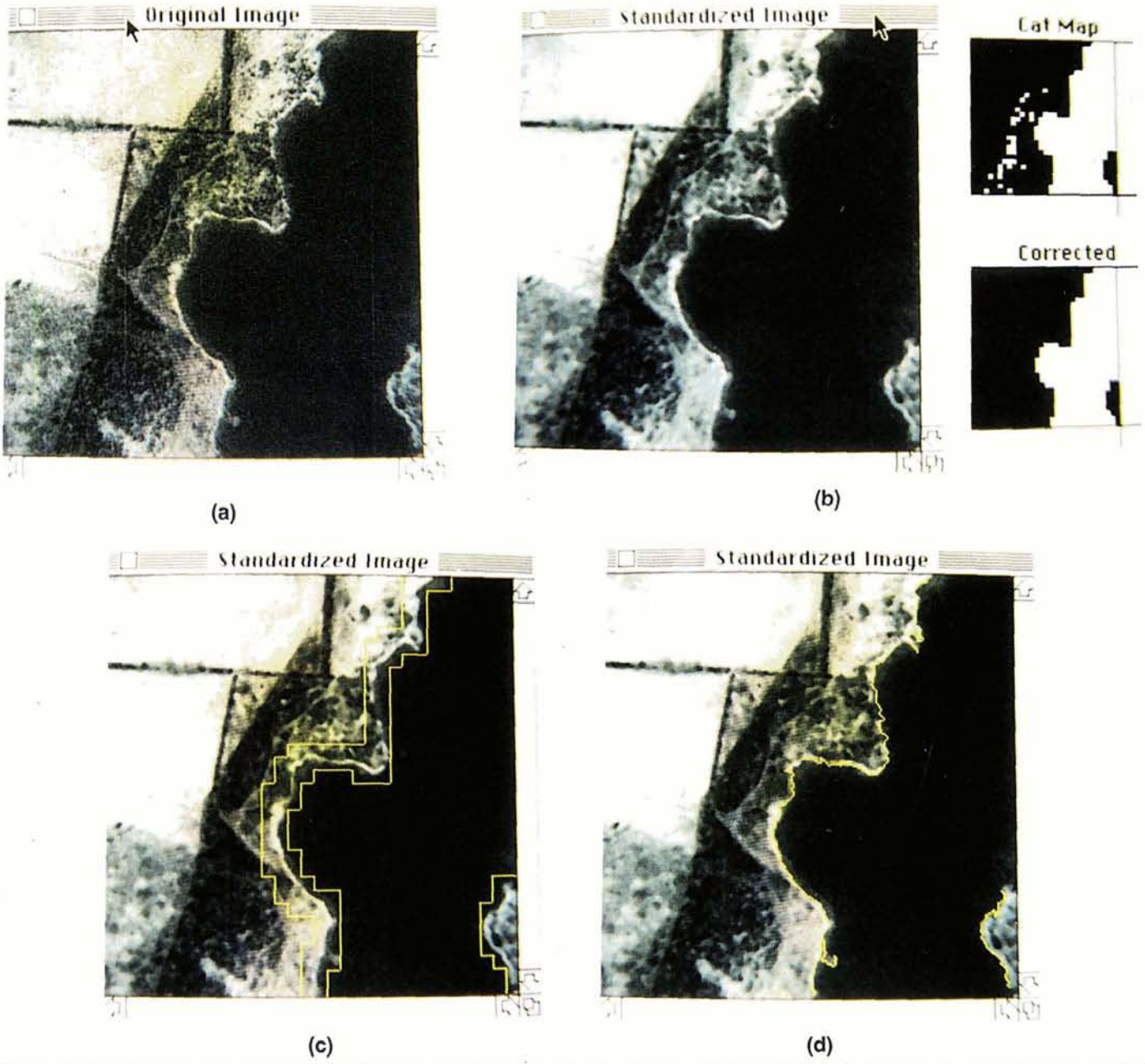


FIG. 11. Shoreline processing sequence. (a) Original image. (b) Standardized image along with the original category map produced by the neural net classifier and the corrected category map produced by connected components labeling and small region merging (Step 2). (c) The strip map transitions overlaid onto the standardized image (Step 3). (d) final shoreline delineation.

noticeable improvement when using different weighting functions. Based on these results, we elected to use the simple smoothed differential filter (1, 1, 1, −1, −1, −1) for candidate point extraction.

Threshold Level. We set the threshold by analyzing sample grey level profiles near the shoreline. We initially set the threshold at 45 and found that there was little noticeable performance difference when setting the threshold at any level between 40 and 55. We elected to keep the threshold set at 45 in the final implementation.

STEP 4: CONTINUOUS DELINEATION

The candidate shoreline points selected in Step 3 are joined into an approximately continuous shoreline delineation. The connections are made based on the proximity of points to each other, and their tendency to lie along a continuous curve. The selected connectivity algorithm requires two passes through the binary candidate point image. The initial pass attempts to determine a continuous track along the candidate points. When the track comes to a point for which there are no adjacent candidates, an attempt is made to continue by filling in a single pixel gap. If the track cannot be continued (i.e., there are no new candidates within a two-pixel distance of the end of the track), then the length of the track is compared to a segment threhold (empirically selected as nine). If the track length does not exceed the threshold, then all members of the track are eliminated from further consideration. If the threshold is met, then the track is retained as a *shoreline segment*.

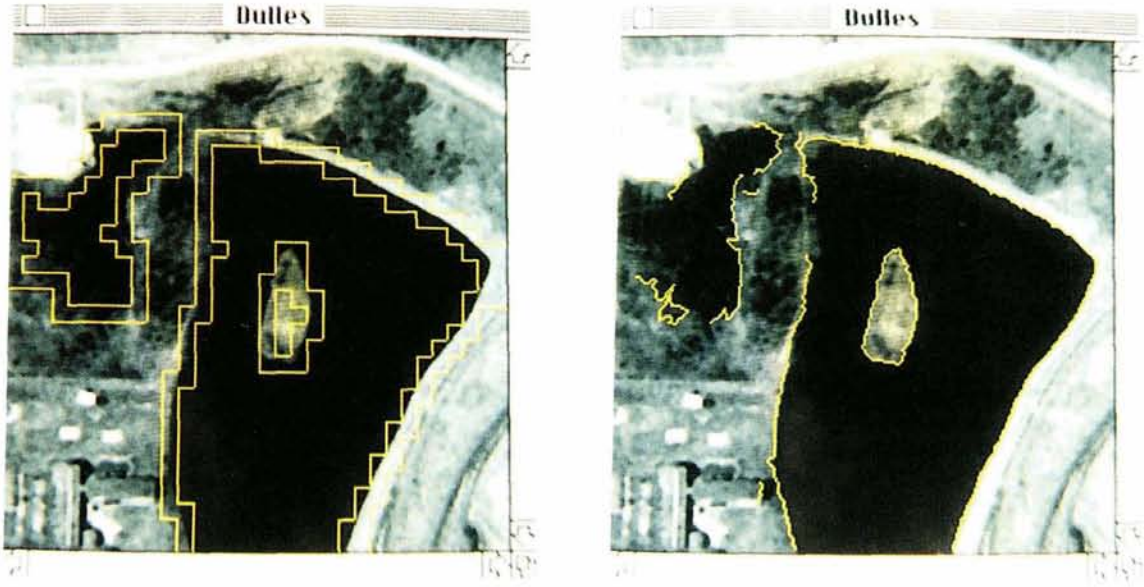


FIG. 12. Result for Scene 2. (a) Shoreline strip overlaid onto standardized image. (b) Final Shoreline Delineation. This scene shows primary limitations of current algorithm: (1) misclassified region in upper left, (2) gap in shoreline delineation in upper left corner of lake, (3) isolated segment due to building shadow in lower left part of lake.

A second pass through the candidate point image is made in order to join shoreline segments. If the endpoints of two segments are within a five-pixel distance of each other, then these points are connected by a straight line. The output of this step of the processing is the final shoreline delineation. Further details of these two passes are given below:

First Pass.

As described above, the first pass of the connectivity algorithm attempts to follow continuous strings of points through the candidate point image. In doing so, it gives priority to points that are in the direction of the previous two points along the current track (string). This decision logic is used only when there are multiple points neighboring the current endpoint of a track. The neighboring points that are not "along track" are removed from the candidate point image, thereby filtering out spurious points.

The first pass of the connectivity algorithm applies the following logic:

- Initialize Track Direction
- Scan Candidate Point Image for Next Candidate
- Look for another point within a 3 by 3 neighborhood of the candidate
  - If a single point is within the 3 by 3 window, assume that it is part of the current track, and set Track Direction to the cell number of the point.
  - If multiple points are within the window, select the one closest to the current Track Direction and set all other points within the window to zero. Set Track Direction to the cell number of the selected point.
    - If no points are within the window, expand the search window to 5 by 5:
    - If there is at least one point within the expanded window, select the point most closely aligned with the current track direction. Fill in the single pixel gap. Set track direction to the cell number of the filled pixel.
    - If there are no points within the expanded window, terminate the track.
      * Compare length of terminated track to segment length threshold:
        ** If threshold is exceeded, retain shoreline segment
        ** Else, delete all candidates within segment.
- Repeat for all remaining candidate points.

Second Pass.

The second pass of the connectivity algorithm operates on the shoreline segments resulting from the first pass. Each segment endpoint is compared to neighboring pixels in the point image. If there are any other segment endpoints within a five-pixel distance of a given endpoint, then a straight line connection is made between that endpoint and the *nearest* endpoint that meets the five-pixel distance threshold.

A more sophisticated second-pass algorithm can be used to eliminate spurious segments that are not part of the actual shoreline. This algorithm would operate on the shoreline segments as follows:

- Label each shoreline segment.
- Examine both endpoints of a shoreline segment simultaneously
  - If neither endpoint can be linearly connected to another segment, compare the length of this segment to a theshold (this threshold would be on the order of 20 pixels)
    - If the threshold is met, then retain the segment
    - Else, delete the segment
    - If either endpoint can be linearly connected to another segment, make the connection and give the connected segments the same label
- Repeat for all segements.

The result of this processing would be a few very long shoreline segments that comprise the automated delineation of the shoreline.

EXPERIMENTAL RESULTS

A variety of experiments were performed to assess the performance of the four-step processing chain. These algorithms were applied to several 256 by 256 digitized USGS aerial photos (1:50,000-scale imagery sampled at a ground sample distance of approximately 2.5 metres). Figures 11a to 11d show the sequence of operations for one of these scenes. The original image is shown in Figure 11a. Figure 11b shows the image after histogram normalization and median filtering (with a 3 by 3 kernel). Sets of about 30 8 by 8 blocks were selected from each of three standardized scenes to produce a training set of 100 three-

dimensional vectors formed from the first three spectral rings (see Figure 6). The training sites were selected to span a wide range of statistical variation as determined by visual inspection of the image data. Training on this set produced a neural net architecture having ten hidden nodes which was applied to several test images which were not part of the training data. Results on the image in Figure 11 are typical. Training sessions typically required less than 500 sweeps through the training data.

Figure 11b also shows the category map and corrected map produced by the neural net classifier in Step 2 of the processing chain. Although these maps are produced at an 8:1 reduction in resolution (due to the 8 by 8 block sizes), we have displayed them at a 1:3 magnification. Note that the connected components labeling and merging operations have eliminated small misclassified regions (the white blocks on the black background). Figure 11c shows the strip map boundaries, produced in Step 3, overlaid on the image. The shoreline strip completely encloses the shoreline and provides a guide for finding the set of initial candidate shoreline points. Many of the extraneous shoreline candidate points are removed by local operations described in Step 4. All small (<5-pixel) gaps in the shoreline are then filled in using linear segments. The final result is shown in Figure 11d.

The suite of algorithms was applied to several other scenes without retraining the neural net classifier. Most of these results were similar in performance to Figure 11. In the example shown in Figure 12, however, several interesting errors occur which reveal the limitations of the current algorithm. The area in the upper left portion of the scene is not water (determined by inspection using higher resolution image data). It is mistaken for water due to its homogeneous dark gray tone. A large gap is left in the upper left portion of the lake shoreline due to the relatively poor quality of the shoreline edges in this region. In the lower left corner of the lake, a building and shadow are mistaken for a water/land transition. In the following section, we present recommendations for enhancements to the processing chain that address these and other issues related to automated feature delineation.

SUMMARY AND RECOMMENDATIONS FOR FUTURE WORKS

We have shown that a neural network can, indeed, be trained to distinguish land from water using power spectral ring data as input. The misclassification rate is low and most misclassifications can be removed by simple image processing algorithms. Although these algorithms appear to perform reasonably well, there are a number of enhancements that could be made in future versions of a shoreline delineation system that would address current limitations. These include:

- Scanning and classifying the image data in a resolution hierachy. Early stages of processing would deal with lower resolution data in a reduced resolution data set (RRDS). Neural net classifiers may have to be trained (off-line) for *each* level in the RRDS, but this would be a straightforward process. Hierarchical processing could significantly reduce processing time by eliminating processing in regions far from shoreline areas. Processing accuracy at level k of the RRDS could be improved by using the categorization results obtained at level $k-1$.
- All empirical results shown above used a three-ring measurement set obtained from 8 by 8 image blocks. We believe that improved category maps would be obtained by using a combination of spectral ring data and histogram data. Also, all of our experiments used an 8 by 8 block size. We recommend an investigation that considers alternative block sizes.
- The block measurement algorithm currently processes only non-overlapping blocks. Within the coarse shoreline strip, however, it may be possible to further refine the shoreline location estimate by scanning with a sliding window, in a manner closer to convolution. Each pixel within the shoreline strip would then be assigned a category according to the output of the neural net classifier.
- The current edge detection and enhancement algorithms have parameters that are determined empirically by observing their performance on typical data. We believe that it may be possible to *train* a neural network-based processor to perform similar operations. Because training is an optimization process, we believe that it should be possible to obtain improved performance from this class of algorithms.
- Relatively simple post-processing algorithms could be applied to the delineation data to remove short extraneous segments and to bridge gaps in the delineation. In some cases gaps are produced as a result of the vertical and horizontal scanning limitation in Step 3 of the processing chain. A post-processing algorithm could provide scanning at arbitrary orientations.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of this research by General Electric Co., Military and Data Systems Operations, Aerospace Business Group, under Contract No. F14-J13007.

REFERENCES

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

Cortes, C., and J. Hertz, 1989. A Network System for Image Segmentation, *International Joint Conference on Neural Networks*, Vol I, Washington, D.C., pp. 121–126.

Dutta, S., and S. Shekhar, 1988. Bond Rating, A Non-Conservative Application of Neural Networks, *International Joint Conference on Neural Networks*, Vol I, San Diego, pp. 443–450.

Fahlman, S., and C. Lebiere, 1990. *The Cascade-Correlation Learning Architecture*, Carnegie Mellon University, School of Computer Science report CMU-CS-90-100.

Funahashi, K., 1989. On the Approximate Realization of Continuous Mappings by Neural Networks, *Neural Networks*, Vol. 2, No. 3, pp. 183–192.

Glover, D., 1988. An Optical Fourier/Electronic Neurocomputer Automated Inspection System, *International Joint Conference on Neural Networks*, pp. 569–576.

Guyon, I., I. Poujad, L. Personnaz, G. Dreyfus, J. Denker, and Y. LeCun, 1989. Comparing Different Neural Network Architectures for Classifying Handwritten Digits. *International Joint Conference on Neural Networks*, Vol II, Washington, D.C. pp. 127–132.

Hecht-Nielsen, R., 1989. Theory of the Backpropagation Neural Network, *International Joint Conference on Neural Networks*, Vol I, Washington, D.C., pp. 593–605.

Hepner, G. F., T. Logan, N. Ritter, and N. Bryant, 1990, Artificial Neural Network Classification Using a Minimal Training Set: Comparison to Conventional Supervised Classification, *Photogrammetric Engineering and Remote Sensing*, 56(4) 469–473.

Hopfield, J. J., and D. W. Tank, 1985. 'Neural' Computation of Decisions in Optimization Problems, *Biological Cybernetrics* 52.

Huang, W., and R. Lippman, 1987. Comparisons Between Neural Net and Conventional Classifiers, *IEEE First International Conference on Neural Networks*, Vol. IV, San Diego, pp. 485–494.

Kuperstein, M., and J. Rubinstein, 1989. Implementation of an Adaptive Neural Controller for Sensory-Motor Coordination, *International Joint Conference on Neural Networks*, Vol II, Washington, D.C., pp. 305–310.

Lippmann, R. P., 1987, An Introduction to Computing with Neural Nets, *IEEE ASSP Magazine*, April, pp. 4–22.

Lippman, R. P., and B. Gold, 1987. Neural Net Classifiers Useful for Speech Recognition, *First International Conference on Neural Networks*, San Diego, pp. 417–426.

McKeown, D. M., 1984. Knowledge-Based Aerial Photo Interpretation, *Photogrammetria* 39.

Nagao, M., and T. Matsuyama, 1980. *A Structural Analysis of Complex Aerial Photographs*. Plenum Press, New York.

Parker, D. B., 1986. A Comparison of Algorithms for Neuron-like Cells, in *Neural Networks for Computing*. (J. S. Denker, ed.) AIP Conf. Proceedings 151, Snowbird, Utah.

Pao, Y., 1989. *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, Massachussetts.

Rosenfeld, A., and A. Kak, 1976. *Digital Picture Processing*. Academic Press, New York.

Rumelhart, D. E., and J. L. McClelland, 1986. *Parallel Distributed Processing*. MIT Press, Cambridge, Massachusetts.

Sternberg, S. R., J. Serra, *et al.*, 1986. *Computer Vision, Graphics, and Image Processing*, (Special Section on Mathematical Morphology) Vol 35.

Vogl, T., J. Mangis, A. Rigler, W. Zink, and D. Alkon, 1988. Accelerating the Convergence of the Back-Propagation Method, *Biological Cybernetics* 59, pp 257–263.

Watrous, R., 1987. Learning Algorithms for Connectionist Networks: Applied Gradient Methods of Nonlinear Optimization, *First International Conference on Neural Networks*, Vol II, San Diego, pp. 619–627.

Werbos, P., 1974. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, Ph.D. Thesis, Applied Mathematics, Harvard University.

(Received 30 May 1990; revised and accepted 16 November 1990)

Forthcoming Articles

Eugenia M. Barnaba, Warren R. Philipson, Arlynn W. Ingram, and *Jim Pim,* The Use of Aerial Photographs in County Inventories of Waste-Disposal Sites.

Michel Boulianne, Louis Cloutier, and *Sanjib K. Ghosh,* Cerebral Biopsies Using a Photogrammetric Probe Simulator.

Haluk Cetin and *Donald W. Levandowski,* Interactive Classification and Mapping of Multi-Dimensional Remotely Sensed Data Using n-Dimensional Probability Density Functions (nPDF).

Yue Hong Chou, Slope-Line Detection in a Vector-Based GIS.

Raymond L. Czaplewski, Misclassification Bias in Areal Estimates.

Heinrich Ebner, Wolfgang Kornus, Gunter Strunz, Otto Hofmann, and *Franz Muller,* A Simulation Study on Point Determination Using MOMS-02/D2 Imagery.

Peter F. Fisher, First Experiments in Viewshed Uncertainty: The Accuracy of the Viewshed Area.

G. M. Foody, A fuzzy Sets Approach to the Representation of Vegetation Continua from Remotely Sensed Data: An Example from Lowland Heath.

W. Frobin and *E. Hierholzer,* Video Rasterstereography: A Method for On-Line Measurement of Body Surfaces.

Peng Gong and *J. Douglas Dunlop,* Comments on the Skidmore and Turner Supervised Nonparametric Classifier (and response).

Daniel K. Gordon, Paul W. Mueller, and *Matthew Heric,* An Analysis of TIMS Imagery for the Identification of Manmade Objects.

Peter E. Joria, Sean C. Ahearn, and *Michael Conner,* A Comparison of the SPOT and Landsat Thematic Mapper Satellite Systems for Detecting Gypsy Moth Defoliation in Michigan.

D. King, Determination and Reduction of Cover Type Brightness Variations with View Angle in Airborne Multispectral Video Imagery.

J. Lavreau, De-Hazing Landsat Thematic Mapper Images.

Zhilin Li, Effects of Check Points on the Reliability of DTM Accuracy Estimates Obtained from Experimental Tests.

Donald L. Light, The New Camera Calibration System at the U. S. Geological Survey.

Hans-Gerd Maas, Digital Photogrammetery for Determination of Tracer Particle Coordinates in Turbulent Flow Research.

Ronald T. Marple and *Eugene S. Schweig, III,* Remote Sensing of Alluvial Terrain in a Humid, Tectonically Active Setting: The New Madrid Seismic Zone.

Ram M. Narayanan, Steven E. Green, and *Dennis R. Alexander,* Soil Classification Using Mid-Infrared Off-Normal Active Differential Reflectance Character- istics.

Kurt Novak, Rectification of Digital Imagery.

Laurent Polidori, Jean Chorowicz, and *Richard Guillande,* Description of Terrain as a Fractal Surface, and Application to Digital Elevation Model Quality Assessment.

Michael Rast, Simon J. Hook, Christopher D. Elvidge, and *Ronald E. Alley,* An Evaluation of Techniques for the Extraction of Mineral Absorption Features from High Spectral Resolution Remote Sensing Data.

Randall L. Repic, Jae K. Lee, Paul W. Mausel, David E. Escobar, and *James H. Everitt,* An Analysis of Selected Water Parameters in Surface Coal Mines Using Multispectral Videography.

Thierry Toutin, Yves Carboneeau, and *Louiselle St-Laurent,* An Integrated Method to Rectify Airborne Radar Imagery Using DEM.

Ausust 1991 Image Registration and Correlation Special Issue

Emmanuel P. Baltsavias and *Dirk Stallmann,* Trinocular Vision for Automatic and Robust Three-Dimensional Determination of the Trajectories of Moving Objects.

Doug C. Brockelbank and *Ashley P. Tam,* Stereo Elevation Determination Techniques for SPOT Imagery.

Joshua S. Greenfeld, An Operator-Based Matching System.

Mengxiang Li, Hierarchical Multipoint Matching.

Eric J. M. Rignot, Ronald Kwok, John C. Curlander, and *Shirley S. Pang,* Automated Multisensor Registration: Requirements and Techniques.

Toni Schenk, Jin-Cheng Li, and *Charles Toth,* Towards an Autonomous System for Orienting Digital Stereopairs.

S. D. Wall, T. G. Farr, J -P Muller, P. Lewis, and *F. W. Leberl,* Measurement of Surface Microtopography.