Interactive digital image processing for terrain data extraction, phase 2

Thomas F. Wescott
William C. Dallam
Christopher J. Peterson

General Electric Company
Space Systems Division
4701 Forbes Blvd.
Lanham, MD 20706

SEPTEMBER 1981

Approved for public release; distribution unlimited

Copy available to DTIC does not permit fully legible reproduction

Prepared for

U.S. ARMY CORPS OF ENGINEERS
ENGINEER TOPOGRAPHIC LABORATORIES
FORT BELVOIR, VIRGINIA 22060
Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

The citation in this report of trade names of commercially available products does not constitute official endorsement or approval of the use of such products.
DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.
The Phase 2 of an experimental study has been performed to investigate the feasibility and use of man-machine interactive digital image processing techniques when applied to the extraction of terrain analysis data from aerial imagery. This phase of the study concentrated on applying techniques to extract water resources data elements and on the development and coding of software and algorithms. The developed software and algorithms, pertained to filtering of themes, and image scaling. Two digital processing
techniques to "skeletonize" features, best mapped as lines, were established and tested with limited success. Relative to water resources data elements, success was achieved in the extraction of alignment of watercourses and shore line alignment of watercourses of water bodies. Partial success was achieved with the elements of delineation of wet areas and the identification of terminal points of watercourse segments. Image processing techniques for the extraction of areas subject to flooding and for the measurement of bank heights were not practical with the existing software.

The development/implementation of software, and the processing and analysis of imagery was conducted at the General Electric Digital Image Analysis Laboratory.

<table>
<thead>
<tr>
<th>Accession For</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTIS GRAI</td>
</tr>
<tr>
<td>DTIC TAB</td>
</tr>
<tr>
<td>Unannounced</td>
</tr>
<tr>
<td>Justification</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Distribution/Availability Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dist</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>D</td>
</tr>
</tbody>
</table>

Unclassified
This document is generated under Contract DAAK70-79-C-0153 for the U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia 22060 by the General Electric Company, Space Division, Beltsville, Maryland. The Contracting Officer's Representative was Mr. Joseph H. Kitrosser.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>1</td>
</tr>
<tr>
<td>Illustrations</td>
<td>iv</td>
</tr>
</tbody>
</table>

## I INTRODUCTION

| A. Background                   | 1    |
| B. Objective                    | 2    |
| C. Scope                        | 2    |
| D. Water Resources Elements     | 3    |

## II IMAGERY USED

| A. Data Selection               | 4    |
| B. Data Sets                    |      |
| 1. Elizabeth City & North Carolina | 4   |
| 2. Ripley, West Virginia Area   | 5    |
| 3. Clarion County, PA           | 5    |
| 4. Fort Belvoir/Woodbridge, VA Area | 6   |
| 5. Tennessee Valley Authority   | 6    |

## III TECHNIQUES DEVELOPMENT

| A. Image resolution and Scale Changing | 7    |
| 1. Scale Changing via Pixel Replication | 9    |
| 2. Fast Area Averaging                | 11   |
| 3. Sub-pixel Interpolation for Image  | 14   |
|   Enlargement                        |      |
| B. Feature Skeletonizing             | 18   |
| 1. Skeletonizing Features via Gradient Zero crossing | 19 |
| 2. Skeletonizing Features via Golay Processing | 25 |
| C. Theme Smoothing Method             | 31   |
| D. Use of the Theme Filter           | 40   |
| 1. Capabilities                      | 40   |
|   Processing Example                 | 41   |
# TABLE OF CONTENTS (cont)

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IV RESULTS</td>
<td>47</td>
</tr>
<tr>
<td>A. Extraction of Watercourse and Associated Elements</td>
<td>47</td>
</tr>
<tr>
<td>1. Watercourses</td>
<td>47</td>
</tr>
<tr>
<td>2. Extracting Watercourses from Radar Imagery</td>
<td>52</td>
</tr>
<tr>
<td>3. Watercourse Extraction Using Combinations of Dissimilar Images</td>
<td>61</td>
</tr>
<tr>
<td>B. Extraction of Dry Gaps and Associated Elements</td>
<td>72</td>
</tr>
<tr>
<td>1. Dry Gaps</td>
<td>72</td>
</tr>
<tr>
<td>2. Categorizing Watercourse by Drap Gap Width</td>
<td>73</td>
</tr>
<tr>
<td>3. Terminal Points of Watercourse Segments</td>
<td>78</td>
</tr>
<tr>
<td>4. Encoding Watercourse Delineation</td>
<td>82</td>
</tr>
<tr>
<td>V CONCLUSIONS</td>
<td>85</td>
</tr>
<tr>
<td>VI RECOMMENDATIONS</td>
<td>87</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>IMAGE RESOLUTION AND SCALE CHANGING PROGRAM</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>THEME FILTERING PROGRAM</td>
</tr>
</tbody>
</table>
### LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Legend</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Image Scaling Program Flow Diagram</td>
</tr>
<tr>
<td>2</td>
<td>Scale Change by Replication</td>
</tr>
<tr>
<td>3</td>
<td>Fast Area Averaging</td>
</tr>
<tr>
<td>4</td>
<td>Four Point Context Dependent Interpolation</td>
</tr>
<tr>
<td>5</td>
<td>Typical Watercourse Theme (Purposely Left Rough to Provide a Difficult Test)</td>
</tr>
<tr>
<td>6</td>
<td>Watercourse Mask Where Connectivity Has Been Established by Slicing a Low Pass Filtered Image</td>
</tr>
<tr>
<td>7</td>
<td>Processing Sequence for Skeletonizing</td>
</tr>
<tr>
<td>8</td>
<td>Watercourse Skeletonizing by Gradient on Spatially Simple Stream</td>
</tr>
<tr>
<td>9</td>
<td>Unsuccessful Attempt at Watercourse Delineation By Zero Crossing Line of Maximum Local Gradient. Note that discontinuities are introduced only when watercourse is oriented at 10 o'clock.</td>
</tr>
<tr>
<td>10</td>
<td>Golay Skeletonizing of Watercourse Theme</td>
</tr>
<tr>
<td>11</td>
<td>Golay Skeletonizing With Updated Images</td>
</tr>
<tr>
<td>12</td>
<td>Watercourse Delineation by Skeletonizing the Mask of Figure 6</td>
</tr>
<tr>
<td>13</td>
<td>Theme Filter</td>
</tr>
<tr>
<td>14</td>
<td>Theme Filter Data Structure</td>
</tr>
<tr>
<td>15</td>
<td>Theme Filter Operator Interface</td>
</tr>
<tr>
<td>16</td>
<td>Theme Filter Replacement Logic</td>
</tr>
<tr>
<td>17</td>
<td>Comparison of Fort Belvoir/Woodbridge Vegetation Themes (Top) with Manual Analysis Results (Bottom)</td>
</tr>
<tr>
<td>18</td>
<td>18A Theme Filter Results</td>
</tr>
<tr>
<td></td>
<td>18B Simplified Themes</td>
</tr>
<tr>
<td>19</td>
<td>Outlines of Vegetation and Water Themes</td>
</tr>
<tr>
<td>Figure</td>
<td>Legend</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>20</td>
<td>Outlines of Simplified Themes</td>
</tr>
<tr>
<td>21</td>
<td>Watercourse Extraction via Level Slicing the 1-Meter Resolution Panchromatic Photographic Image</td>
</tr>
<tr>
<td>22</td>
<td>Water Identification via Joint Use of the 1-Meter Resolution Panchromatic Photographic Image and the Derived Texture Image (smoothed)</td>
</tr>
<tr>
<td>23</td>
<td>Watercourse Extraction via Level Slicing the 4-Meter Resolution Panchromatic Photographic Image</td>
</tr>
<tr>
<td>24</td>
<td>Watercourse Identification via Joint Use of the 4-Meter Resolution Panchromatic Photographic Image and the Result of the Image Minus 7 x 7 Pixel Local Average</td>
</tr>
<tr>
<td>25</td>
<td>X-Band Horizontal Polarization Subscene</td>
</tr>
<tr>
<td>26</td>
<td>X-Band Vertical Polarization Subscene</td>
</tr>
<tr>
<td>27</td>
<td>L-Band Horizontal Polarization Subscene</td>
</tr>
<tr>
<td>28</td>
<td>L-Band Vertical Polarization Subscene</td>
</tr>
<tr>
<td>29</td>
<td>SAR Data and Subscene Locations</td>
</tr>
<tr>
<td>30</td>
<td>Water Theme Sliced from X-Band SAR w/Horizontal Polarization</td>
</tr>
<tr>
<td>31</td>
<td>Water Theme Derived from X-Band SAR w/Vertical Polarization</td>
</tr>
<tr>
<td>32</td>
<td>Water Theme Derived from X-Band SAR w/Horizontal Polarization</td>
</tr>
<tr>
<td>33</td>
<td>Water Theme Derived from L-Band SAR w/Vertical Polarization</td>
</tr>
<tr>
<td>34</td>
<td>X-Band Horizontal Polarization Subscene</td>
</tr>
<tr>
<td>35</td>
<td>X-Band Horizontal Polarization Data Smoothed by a 7x7 Pixel Moving Average Operator</td>
</tr>
<tr>
<td>36</td>
<td>Associated Microtexture Image</td>
</tr>
<tr>
<td>37</td>
<td>Micotexture Image Data Smoothed by a 7x7 Pixel Moving Average Operator</td>
</tr>
<tr>
<td>Figure</td>
<td>Legend</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------</td>
</tr>
<tr>
<td>38</td>
<td>Water Theme Derived from w/Dimensional Partitioning of X-Band SAR w/Horizontal Polarization and Associated Microtexture Image Data</td>
</tr>
<tr>
<td>39</td>
<td>Water Theme Derived from 2-Dimensional Partitioning of X-Band SAR w/Horizontal Polarization Data and Textured Data Smoothed by a 7 x 7 Pixel Moving Average Operator</td>
</tr>
<tr>
<td>40</td>
<td>Theme Derived from the 7 x 7 Pixel Smoothed Version of the X-Band SAR Image w/Horizontal Polarization</td>
</tr>
<tr>
<td>41</td>
<td>Little River Water Course Theme Derived from Panchromatic Photography and SAR Data and From the Microtexture Version of the Photography</td>
</tr>
<tr>
<td>42</td>
<td>Smoothed Version of Figure 41</td>
</tr>
<tr>
<td>43</td>
<td>Edged Version of Figure 42</td>
</tr>
<tr>
<td>44</td>
<td>Chapel Creek Water Course Theme Derived from Panchromatic Photography and SAR Data</td>
</tr>
<tr>
<td>45</td>
<td>Chapel Creek Water Course Theme Derived from Panchromatic Photography and SAR Data and From the Smoothed Microtexture Version of the Photography</td>
</tr>
<tr>
<td>46</td>
<td>Smoothed Version of Figure 45</td>
</tr>
<tr>
<td>47</td>
<td>Edged Version of Figure 46</td>
</tr>
<tr>
<td>48</td>
<td>Manually Interpreted Water Course</td>
</tr>
<tr>
<td>49</td>
<td>Manually Interpreted Dry Gap</td>
</tr>
<tr>
<td>50</td>
<td>Skeletorized Water Course</td>
</tr>
<tr>
<td>51</td>
<td>Skeletorized Dry Gap</td>
</tr>
<tr>
<td>52</td>
<td>Categorizing a Dry Gap By Width</td>
</tr>
<tr>
<td>53</td>
<td>Dry Gap Categorized by Width and Displayed as Polygons</td>
</tr>
<tr>
<td>54</td>
<td>Combining Watercourse Center Line with Dry Gap Categories Performing a Logical Itersection Produces Water Course Segments Categorized by Width</td>
</tr>
</tbody>
</table>
# LIST OF ILLUSTRATIONS (CONT.)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Legend</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>Raster Operating Line Dotting Algorithm</td>
<td>83</td>
</tr>
<tr>
<td>56</td>
<td>Encoding Watercourse Delineation</td>
<td>84</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

A. Background

During the period of September 1979 through September 1980, Phase 1 of the experimental study, "Interactive Digital Image Processing for Terrain Data Extraction", was performed. The 12-month study investigated the feasibility and utility of man-machine interactive digital processing techniques when applied to the extraction of terrain analysis data from aerial imagery. The study focused primarily on the extraction of vegetation data elements from digitized panchromatic photography, with a small amount of attention given to thermal infrared and side-looking radar imagery.

Of thirteen vegetation data elements listed in the USAETL Terrain Analysis Procedural Guide for Vegetation, eight were addressed in varying degrees of depth. Interactive digital techniques were developed for vegetation/land cover boundary extraction and for the extraction of several forest-related data elements. For the forest-related elements, three digital themes were developed from digitized panchromatic photography. The three digital themes and a developed automatic operation allowed estimates of percent canopy closure, stems per hectare; crown diameter; stem diameter; stems per hectare per diameter class; and stem spacing to be produced. Best results were achieved with a spatial resolution of approximately three feet in the input digital image.

In addition, techniques were developed enabling the extraction of vegetation/land cover boundaries. These techniques were most effective when the spatial resolution of the digital image is in the range of 8 to 20 feet. With a moderate amount of interactive analysis, processing can be employed to extend the boundary extraction rapidly and efficiently to relatively large areas. Procedures were also developed for distance, elevation, and area measurement from digital imagery. Further testing and development of new techniques were recommended as a result of Phase 1 work.
B. **Objective**

The objective of the investigation was to continue the study of man-machine digital image processing techniques to the extraction of terrain analysis data from aerial imagery.

C. **Scope**

The scope of the second phase part of the study involved the extraction of data identified in the *Terrain Analysis Procedural Guide for Drainage and Water Resources*. Within the scope was the development and coding of software and algorithms for filtering themes and image scaling. Input data for the second phase included panchromatic, thermal infrared and side looking synthetic aperture radar imagery.

D. **Water Resources Elements**

In this second phase, the water resources elements considered were:

1. Alignment of watercourses
2. Shore alignment of waterbodies
3. Delineation of wet areas
4. Identification of areas subject to flooding
5. Identification of terminal points of watercourse segments
6. Measurement of bank heights
II. IMAGERY USED

A. Data Selection

Data types used for Phase 2 consisted of two forms, i.e. digital and imagery. Imagery selected as transparencies were digitized into 512 x 512 pixel images (subscenes) with 8-bit gray level quantization. Certain data sets used in phase 1 were applied in Phase 2, specifically pertaining to Fort Belvoir/Woodridge, Va. area and Daedalus scanner images of Pennsylvania. Additional panchromatic photography, synthetic aperture radar (SAR), thermal infrared (TIR) and infrared (IR) data were acquired from the Tennessee Valley Authority (purchased) and General Electric Space Division (loaned) at Valley Forge. Selection of data used for analysis considered the following factors:

- Scene content (various watercourses and water bodies)
- Resolution (watercourses of 1-15 meters width)
- Absence of clouds, cloud shadows and haze
- Uniformity of scene feature brightness levels across the image
- Near-vertical view of the scene
- General quality of the transparency (i.e. absence of scratches, etc)

B. Data Sets

A list of all data sets used for the generation of selective 512 x 512 pixels images (subscenes) is shown in Table 1.

<table>
<thead>
<tr>
<th>Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Elizabeth City, North Carolina; imagery; USAETL</td>
</tr>
<tr>
<td>a. Knobbs Creek Area</td>
</tr>
<tr>
<td>(1) 98° synthetic aperture radar #1 imagery</td>
</tr>
<tr>
<td>(2) 52° synthetic aperture radar #9 imagery</td>
</tr>
<tr>
<td>(3) Panchromatic #39 imagery</td>
</tr>
<tr>
<td>(4) High resolution thermal flightlines 2, 3 and 4</td>
</tr>
</tbody>
</table>
Table 1 (Cont.)

b. Little River Area
   (1) 98° synthetic aperture radar #5 imagery
   (2) 98° synthetic aperture radar #9 imagery
   (3) 52° synthetic aperture radar #5 imagery
   (4) 52° synthetic aperture radar #10 imagery
   (5) 278° synthetic aperture radar #5 imagery
   (6) High resolution thermal infrared #9 imagery
   (7) Low resolution thermal infrared imagery

c. New Begun Creek Area
   (1) 98° synthetic aperture radar #10 imagery
   (2) 278° synthetic aperture radar #6 imagery
   (3) Panchromatic #95 imagery
   (4) Low resolution thermal infrared imagery

d. Chapel Creek Area
   (1) 278° synthetic aperture radar #14 imagery
   (2) Panchromatic #91 imagery
   (3) Panchromatic #120 imagery

e. Symond Creek Area
   (1) Panchromatic #47 imagery
   (2) High resolution thermal infrared imagery

2. Ripley, West Virginia Area, digital, General Electric Company
   a. X-band horizontal polarization
   b. X-band vertical polarization
   c. L-band horizontal polarization
   d. L-band vertical polarization

3. Clarion County, Pa, digital, from Phase 1 study
   a. 20,000 ft. altitude; green and thermal infrared channels
   b. 10,000 ft. altitude; green and thermal infrared channels
   c. 5,000 ft. altitude; green and thermal infrared channels
Table 1 (Cont.)

4. Fort Belvoir/Woodbridge, Va. Area, imagery, USAETL, from Phase 1 study
   a. Panchromatic photographs

5. Tennessee Valley Authority
   a. Holston River set
      - Thermal infrared (TIR) low resolution (flight line)
      - Five panchromatic photographs of flight line area
   b. Night TIR (low air temperature)
      - Six panchromatic photographs
   c. Reservoir
      - TIR
      - Five panchromatic photographs
III. TECHNIQUES DEVELOPMENT

A. Image Resolution and Scale Changing

As reported in the first phase of this project, spatial operators, which include the reported microtexture and crown template operators, are very useful tools for the interpretation of monochromatic imagery. The program of successfully categorizing monochrome imagery is in general a much more difficult program than with other types, such as multispectral, because of the relatively smaller amount of information available. Attempting to categorize an image by level slicing alone encounters serious problems, the most difficult being that signatures tend not to be unique enough to be used as the sole descriptor of the classes generally desired. Even seemingly simple classes like water bodies can be impossible to extract via level slicing. Clearly more information than gray level is required to perform categorization, and clearly such information is available, in the form of spatial patterns.

In particular, textural information is of considerable value in the analysis of high resolution images. Texture is an encompassing term with a multitude of types and definitions, most of which have limited use in remote sensing. The most basic and perhaps most widely applicable type of texture is simply surface roughness, which can be relatively easily determined by computing local image variance. Textural operations are scale dependent and for effective use, must be applied at a specific scale which is defined by the size of the microfeatures which generate the rough appearance of larger features being extracted. For example, a typical image might contain three features which we wish to separate, pasture, brush land, and forest. Each of the features is basically a random mixture of vegetation and shadow, and quite likely all three exhibit approximately the same integrated reflectance. Each of the features also exhibit similar textural structure, the appearance of random roughness, but they are distinctly different in textural scale.

To realize the full potential of scale dependent spatial techniques we must be able to efficiently vary the size of the spatial operators used (or at least the effective size), allowing the extraction of a...
considerable range of spatial features. This scaling can be accomplished by either of two methods: (1) by scaling the operator itself, which is the historic method, or (2) by changing the scale and resolution of the image and applying a universal operator. The image scaling approach appears to have several significant advantages:

- Scaling of all but the simplest of spatial operators is not as straightforward as is reducing resolution and scaling an image. Any resolution up to that inherent in the original data can easily be simulated with very simple and rapidly executed algorithms. Allowing, for example, the highly effective and reasonably sized max-min microtexture operator to be used without modification, in the extraction of a wide variety of textural identifiable features.

- Matching spatial feature templates to objects of a particular size can be done much more precisely by changing image scale, as finer scaling increments can be used. Many effective spatial operators cover areas as small as $3 \times 3$ pixels, thus the finest scale adjustment would be a factor of $1/3$.

- The reduction in the computation time required to apply an equivalent operation is significant when image scaling is chosen. Reduction in resolution means reductions in the volume of data processed. Reducing image size by a factor $S$ reduces the number of pixels to be considered by $S^2$, compared with increasing the operator size, which would increase most spatial computations by $S^2$. The relative difference approaches $S^4$.

During Phase 1 of this study, image scaling was accomplished by utilizing those techniques which were readily available at General Electric DIAL without software development. While the usefulness of image scaling to expand the utility of standard and simple spatial operators was firmly established, the actual methods were cumbersome and time consuming.
The ability to vary image scale and resolution rapidly enough for use in interactive analysis was quickly identified as being a missing and necessary tool.

Given the task of using spatial information as an input for the categorization of some image feature, a fairly lengthy procedure might be followed. First the image would be reduced in scale and resolution by some factor defined by the predominant spatial frequency of the object to be extracted. For example, the mean distance between shadows in forest canopy. A spatial operator is then applied to the reduced image generating a derived image where the spatial feature has been extracted or enhanced. This secondary image is then enlarged to the scale of the original for subsequent analysis. Since the derived image contains information of a nature independent from the spatial grey level information of the original, this operation essentially creates a multidimensional feature space which significantly increases an analyst's options when attempting to categorize a monochrome image. Repeating the operation at several scales generates an even more interesting feature space of higher dimensionality.

Based on the experience gained during the Phase 1 analysis of several images, a program was written to perform image scaling by any of three operator selected interpolation methods, nearest neighbor, area averaging, and a fairly unique subpixel interpolation method based on local scene context. Figure 1 is a flow chart of the program.

1. Scale Changing via Pixel Replication

Often simple replication of pixels is a satisfactory or even desirable method of changing the scale of an image. Whether or not it should be the chosen method is dependent on the specific image processing task being performed. For example, if one desires to quickly examine an 'overview' of a large image containing too much data for complete display on a CRT, interpolative resampling with its associated computation could be a waste of time. In other cases, such as resampling an image with a very small scale change, no noticeable degradation will be suffered with this technique. Finally, there are many applications where original radiometric values must be preserved. In these cases replication or
Figure 1. Image Scaling Program Flow Diagram
nearest neighbor resampling is the only simple method. Examples are, when sensor characteristics are being extracted, or when thematic or binary images, which are not gray scale in nature must be scaled.

Pixel replication sounds like a trivial problem, but the capability is so basic and heavily used in image processing, that a serious look at implementation and execution time is worthwhile. In several recent image processing systems the implementation is via hardware and operations are performed in near time. For other systems, a software solution is very reasonable, as rapid execution can be achieved.

Since no real pixel by pixel computation is required, either in determination of output pixel value, or in determination of output position, algorithms can be used which amount only to moving data in a high speed manner from a predetermined location to a predetermined destination. This is accomplished by setting up two address tables based on the replication or sampling required along lines and between lines. These tables, one with an element for each output line and one with an element for each output pixel, contain addresses in the input image. Each of these addresses is then the source location for each desired output pixel.

Using these tables, the actual execution is performed by indirect addressing. Data movement can be done by several methods including special purpose hardware and replacement statements from a high level programming language. In this case replacement was done in an assembler routine to minimize execution time. Omitting I/O limitations, the program takes approximately three seconds to construct a 512 x 512 pixel output image. Figure 2 shows the algorithm used.

2. Fast Area Averaging
This capability is fundamental for the extraction of scaled spatial information from high resolution images in the manner described in Phase 1. Starting with an image digitized and stored at the highest desired resolution area averaging is used to generate a secondary image of "super pixels" where digital numbers are related to the integrated
Figure 2. Scale Change by Replication
reflectance from a large number of original pixels. The spatial frequency content of the derived image is cut off by the span of the averaging area, simulating an image that might result if the same sensor and objects were used in a lower resolution (higher sensor altitude) configuration.

In use, as the averaging area increases, the observable texture of finely structured areas disappears from the derived image. Textural information can then be extracted by differencing the derived image with the original in a manner similar to a technique widely used for edge extraction. Area averaging is then applied to the derived difference image as a smoothing function which generates a grey level image where the digital number is proportional to the local high frequency content (texture) cut off by the original application of the filter.

The approach chosen to implement the filter was to use an adaption of the highly efficient box filtering technique. Box filtering computes a moving average over a rectangular area by using two linear moving averages, one down image columns and one along image rows. Execution is very efficient. If the box size is small in relation to the image size, the number of operations required to compute one output pixel approaches four, and is independent of box size. While box filtering computes more output pixels than are needed for scale changing, it is unlikely that any other algorithm can approach this efficiency. The scale changing program computes all of the column averages used in box filtering, but computes row averages only for the required output lines. Required output pixels are stripped from the row average at regular intervals. All along row operations are done in assembler subroutines to minimize execution time.

As with all spatial operations, some trickery must be used to avoid undesirable image border effects without reducing the size of the output image. Since, for our application, the derived images are often to be recombined with their full resolution parents, maintaining image size is very important. This need is particularly acute if large filters are used. Data must, in effect, be manufactured so that the moving averages
do not contain pixel values of zero at image edges. The method chosen as being the least objectionable, was to increase input image size by reflecting data values about image borders. Figure 3 shows the algorithm for fast area averaging.

3. Sub-pixel Interpolation for Image Enlargement
Selection of an interpolation algorithm for use in image enlargement is not quite as straightforward as many recent papers on the subject would lead us to believe. Truncated or otherwise modified versions of the "theoretically perfect" \((\sin x)/x\) interpolation have been used to achieve good interpolation with a minimum of computation. However, some interesting artifact producing properties are made evident when we use these interpolators as an enlarging tool. In fact, making an enlargement is an excellent means of graphically showing many of the problems associated with various algorithms.

The often used and much discussed "cubic convolution," for example, which computed an interpolated value based on 16 surrounding pixels produces a curious effect on edges with certain orientations. When the point to be interpolated falls directly on one of the lines defined by the pixel grid of the original image, this algorithm performs admirably. In these cases, the algorithm degenerates to a one-dimensional operation. When the interpolation falls between the lines, two-dimensional interpolation is performed and considerable aliasing can occur. When the original image contains an edge feature oriented at 45° to the pixel grid, the aliasing is most apparent. The optimum \((\sin x)/x\) interpolation would produce a blurred (from enlargement) but straight edge in the output image. The cubic convolution reproduces the 45° edge with a wobbly sine wave appearance, showing that individual interpolated values are as much a function of resampling position as they are of image content.

The other commonly discussed interpolator, bilinear interpolation, has been shown in numerous studies to perform just a little worse than cubic
INPUT AVERAGING PARAMETERS

COMPUTE SOURCE ADDRESS TABLE FOR OUTPUT LINES

COMPUTE SOURCE ADDRESS TABLE FOR OUTPUT PIXELS

PRELOAD A SET OF MOVING COLUMN SUMS BY REFLECTING IMAGE TOP

WHILE OUTPUT LINES REMAIN

UPDATE COLUMN SUMS BY ADDING NEXT ROW AND SUBTRACTING OLDEST ROW

IF NEXT OUTPUT LINE HAS BEEN REACHED

ELSE

THEN

COMPUTE ALONG ROW MOVING AVERAGE FROM COLUMN SUMS

EXTRACT OUTPUT LINE FROM AVERAGE AT ADDRESSES LISTED IN PIXEL TABLE

STOP

Figure 3. Fast Area Averaging
convolution in just about every respect. True to form, when subjected to a visual "enlargement test" it shows 45° artifacting a little more severe than cubic convolution does. Considering that it only requires four points, the algorithm does not perform all that poorly. Bilinear interpolation basically performs linear interpolation along two opposite sides of a quadrilateral defined by four pixels, and then performs a third interpolation between these two new points. It then, is based on the assumption that all four points lie in the same plane, a gross assumption which is incorrect whenever the four points are over an image edge.

The interpolation scheme chosen was a four point interpolation which breaks the quadrilateral defined by the four surrounding points into triangles. The most appropriate triangle for interpolation is chosen by the direction of the local image gradient and by the position of the desired point. The interpolation is then controlled by image content and handles 45° edge situations well. See Figure 4.

The context dependent interpolation has been coded and tested at GE DIAL and performs as was expected. The 45° artifacts are completely eliminated. The interpolator does have some remaining problems which could ultimately show up in factor analysis. Angular artifacts remain at 22° to the pixel grid and isolated points are reproduced as diamond shapes. Further use in the generation of factor overlays should determine the value of this method. It is likely that some combination of a linear interpolators and a context dependent interpolator would produce better results.
FOUR POINT CONTEXT DEPENDENT INTERPOLATION


CASE 01 IF ((IC-BI ≥ IA-DI) AND (Y ≥ X))
P = A + X(D-C) + Y(C-A)

CASE 02 IF ((IC-BI ≥ IA-DI) AND (Y < X))
P = A + X(B-A) + Y(D-B)

CASE 03 IF ((IC-BI < IA-DI) AND (X ≤ (1-Y)))
P = A + X(B-A) + Y(C-A)

CASE 04 IF ((IC-BI < IA-DI) AND (X > (1-Y)))
P = C - X(D-C) + (1-Y)(B-D)

Figure 4
B. Feature Skeletonizing

At the end of Phase 1, another capability seen as very useful for the extraction of several terrain elements was the ability to transform those elements characterized by filament type objects into single pixel width lines. Two obvious examples of elements requiring the operation are the delineation of watercourses and roads. There are really two different levels to the program. The first case, where the feature is spatially connected i.e. no breaks, is relatively easy to handle by using a line stripping algorithm to reduce feature width to the point where only one strip of adjacent pixels remain, bisecting the original feature. In general, however, themes are not always well connected. As the desired feature nears the resolution limit of the imagery, or is hard to categorize for some other reason, the feature is usually mapped as a tenuous string of pixels spaced with many gaps. Any widely applicable skeletonizing algorithm should be able to bridge at least small gaps and then shrink the feature to produce the single pixel width map.

The problems are demonstrated vividly when one attempts to extract watercourse alignment. As described in the element extraction section of this report, several methods of extracting large watercourses are feasible and success was achieved with most of the images investigated. The most widely applicable methods rely on image intensity slicing in combination with the unique smoothness or lack of texture that water bodies exhibit to most sensors. Extracting the alignment of very narrow watercourses is a different problem as there is not enough resolution available to make use of texture or any other positively identifying features. While the watercourses can be roughly identified by density slicing, it is difficult to classify them as contiguous objects. Thin or intermittent watercourses are often revealed to the photo interpreter, not by the presence of clearly identifiable water, but only by subtle changes in the surrounding vegetation, or as a linear shadow with no water being imaged.
Prior to a skeletonizing operation, connectivity of the feature must be established if we are to work successfully with narrow watercourses and roads. Some encouraging results in this area were obtained by performing a very low pass filtering operation (box filtering) on unconnected binary theme maps. Filtering in this manner generates a slowly varying grey level output where losses in connectivity have been "bridged" by the slowly varying intensity ramps. The operation transforms strings of isolated pixels into a series of peaks and cols. Level slicing on the smoothed image results in a map much wider than the original feature, and one where connectivity has been established across gaps of width up to the size of the filter. The method has two undesirable effects. First the width of the feature is increased by up to twice the span of the filter size, making subsequent skeletonizing by line stripping methods a more lengthy process. Second, the filtering introduces a straightening of the feature centerline. The straightening can become pronounced and objectionable when filter size approaches the dimensions of feature characteristics such as stream meanders. Figure 5 shows a sample watercourse theme and Figure 6 is the slice where minor gaps have been bridged.

1. Skeletonizing Features via Gradient Zero Crossing

Several attempts were made to utilize linear spatial operators to locate feature centerlines. The results were less than satisfactory, and many problems remain to be addressed, but the use of differentiating operators on grey level images generated from smoothed themes, does present some interesting possibilities.

A binary watercourse theme is first changed by low pass filtering, into a continuous grey level image with a continuous first derivative. In this smoothed image, the watercourse center is characterized by a local intensity maximum which defines a ridge line curve where the local gradient taken perpendicular to the watercourse direction changes sign as it passes through zero. Zero crossing is a uniquely descriptive feature which is independent of the watercourse width, and also independent of the gray level magnitude corresponding to the watercourse. As long as the smoothing operator has established some non zero grey level
Figure 5. Typical Watercourse Theme (Purposely Left Rough to Provide a Difficult Test)
Figure 6. Watercourse Mask Where Connectivity Has Been Established By Slicing A Low Pass Filtered Image.
connection between watercourse pixels, a gradient zero crossing line will exist and can be located. Mapping the zero crossing can be achieved by level slicing all negative (or positive) pixel values and then outlining the area mapped by the slice. Figure 7 is a schematic of the processing sequence.

The zero crossing line has some very attractive features:

- As long as the filter size is greater than one half the watercourse width, gradient zero crossing will occur at the watercourse centerline.
- The zero crossing line tends to establish and maintain connectivity across small gaps in the watercourse.
- The line can be quickly and easily mapped into a single pixel width feature.

Despite the promise of this method, only limited success was achieved when applied to real images. The method was useful on spatially simple watercourses i.e. those without meanders, where the entire stream could be projected in some direction with a one to one mapping, allowing the use of a single gradient direction, as is shown in Figure 8. The method could be used on similar features with limited curvature. Whenever the range of curvature direction was greater than a semicircle, the method was unsuccessful. The problem reduces to one of preserving some of the properties of a direction sensing operator while making the operator work over the image in a directionally invariant manner. There is likely a solution which would make the method universally applicable, but during this project, time limitations brought an end to the investigation.

Most of the obvious adaptations of the gradient method were tried. Taking gradients from two directions and then combining the result into an image where grey scale was related to gradient magnitude failed. In general connectivity was not well preserved, and it was lost at every stream junction point. Likewise, treating orthogonal directions independently failed. This approach produced results which were not totally coincident and thus could not be recombined to form a single centerline.
Reduction of an unconnected watercourse theme to a single pixel width watercourse alignment map by using low pass filtering and zero crossing of gradient.

Figure 7. Processing Sequence for Skeletonizing
Figure 8. Watercourse Skeletonizing by Gradient
On Spatially Simple Stream
The most interesting approach, which also failed, was to use a rotating gradient operator. With a $3 \times 3$ pixel moving operator, four gradients arranged in a semicircle were extracted. Output of the operator was the signed magnitude of the gradient possessing the greatest absolute value. The idea was to always extract the gradient in a direction perpendicular to the watercourse so that the zero crossing line would be preserved, even when the gradient sign was reversed. Unfortunately the operator generated unacceptable discontinuities whenever the watercourse direction was oriented perpendicular to the line defining the semicircle. See Figure 9.

2. **Skeletonizing Features via Golay Processing**

Golay processing is a purely spatial logical operation which is applied to a binary theme map. For each pixel position, the operation examines pixels immediately adjacent to the central pixel, and adds or deletes the central pixel from the output based on the configuration of its adjacent neighbors. Golay transforms were meant to operate on images arranged in a hexagonal array where each pixel has six neighbors. There are several significant advantages to this arrangement when compared to the more commonly used rectangular array. These advantages are: fewer pixels are needed for spatial operators, between neighbors interpixel distances are constant, and a higher order of symmetry exists. Few interactive systems, however, work on hexagonal arrays, making implementation on these systems complex. At CE DIAL, the transforms have been applied in a compromising manner directly to a rectangular array, resulting in a capable program but also one which has some directional artifacts. Figure 10 shows the Golay patterns as implemented at DIAL.

Given a connected watercourse theme, Golay processing is a reasonable means of shrinking the watercourse to the single pixel width line needed to identify alignment. Skeletonizing is accomplished by deleting from the watercourse theme, all pixels with surrounds numbers three, four, and five as labeled in Figure 10. The operation basically removes the outside annular ring of pixels from every contiguous area and thus narrows the feature by two pixels per pass.
Figure 9. Unsuccessful Attempt At Watercourse Delineation
By Zero Crossing Line Of Maximum Local Gradient.
Note That Discontinuities Are Introduced Only Then
Watercourse Is Oriented At 10 o'clock.
Figure 10. Golay Skeletonizing of Watercourse Theme

THE 14 GOLAY SURROUNDINGS ARE BY NUMBER:

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>10</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>01</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>01</td>
<td>11</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>01</td>
<td>02</td>
<td>03</td>
<td>04</td>
<td>05</td>
<td>06</td>
<td>07</td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>10</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>10</td>
<td>09</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>01</td>
<td>10</td>
<td>10</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>08</td>
<td>09</td>
<td>010</td>
<td>011</td>
<td>012</td>
<td>013</td>
<td>014</td>
<td></td>
</tr>
</tbody>
</table>
A problem with the method results from the small neighborhood sampled by the Golay surround, this or any other rotationally invariant operator of 3 x 3 pixels or less, cannot sense the difference between the boundary of a large contiguous area, and a narrow linear feature two pixels wide. The result can be deletion of pixels from both sides of the narrow feature and thus loss of connectivity. Figure 10 shows the same watercourse skeletonized in this manner. There are ways to stop this action. For example, slight smoothing of the feature, a technique which has already been recommended to establish connectivity, can also be used to introduce curvature into long straight boundaries minimizing occurrence of the malfunction. Other techniques, such as slightly randomizing feature boundaries have also been used.

Alternatively, a modification of the process avoids the problem, but introduces other effects. The inability to recognize lines of two pixels width, is avoided if the process is changed so that image raster lines being processed are subjected to immediate updating. With updating, whenever an output pixel is changed, the surround of the subsequent pixel includes that change. The compromise with this approach is that the resulting single pixel width line is no longer centered on the feature. The line is instead shifted to the side of the feature which was last processed. Even with this undesirable effect, the technique might be of value for use on narrow watercourses where the induced offset is small. One very attractive feature of this modification is that two passes over the image are usually sufficient which can mean a significant reduction of execution time. Figure 11 shows a sample result.

Based on the approaches to skeletonizing investigated, the use of Golay pattern transforms of similar operations is viewed as the more promising means to achieve the delineation of narrow watercourses. The properties of rotational invariance, moderate execution time, and a versatile selection of operations make it useful. The main deficiency of the approach is that watercourse gaps of more than one pixel cannot be easily bridged, but this can be overcome by filtering. Figure 12
Figure 11. Golay Skeletonizing With Updated Image
Figure 12. Watercourse Delineation by Skeletonizing the Mask In Figure 6.
represents Golay skeletonizing of the smoothed theme, and is probably the most successful skeletonization exercise. The gaps in this figure are largely a result of the way Golay transforms are currently implemented in GE DIAL with no attempt to compensate for use on a square pixel grid. These deficiencies could most likely be removed with an improved program.

C. Theme Smoothing Method
Categorization of images by computer assisted methods produces, except for the extraction of very easy and distinctly unique classes, results that are spatially complex. The classification may or may not be "noisy", depending on the suitability of the data for extraction of a given class, but will in general exhibit a salt and pepper appearance which might be interpreted as noise. (This effect comes from the fact that with current technology computers perform relatively poorly at spatial tasks.) Most of the commonly used classification schemes are totally aspatial, relying on intensity level in one or more spectral wavelength bands, while completely ignoring feature boundaries, shapes, and texture. Even those methods which aspire to use spatial information, for example, the textural operators described in this report, use spatial information in a very rudimentary manner when compared to the feature extraction capabilities of a human image interpreter.

The interpreter, in contrast, has a relatively poor ability to identify image intensity levels with any real accuracy, and relies very heavily on his spatial abilities, particularly when analysing monochrome imagery. As opposed to a computer classification, which takes place on a per pixel basis, the interpreter performs classification on a per feature basis. This leads to extraction of features by the location of their external boundaries and tends to force the classes into contiguous areas. The interpreter tends to ignore (or accommodate for) minor variances internal to the feature which might affect a computerized classification, and also performs considerable smoothing or straightening of feature outlines.
Deciding which of these two radically different approaches is more accurate is moot, and depends on one's definitions and objectives. What is certain is the fact that the interpreter generally produces a map that is far more suitable to the generation of factor overlays, and if computer assisted methods are to be used successfully, their results must be processed in a manner which changes them into a form which is more compatible with factor overlays. This requires spatial low pass filtering, and occasionally severe filtering. Factor overlays are very simple in spatial structure and contain only enough information to locate and describe a feature of tactical significance. Since overlays are to ultimately be combined, their simplicity becomes even more important. The combination of spatially complex overlays could produce a result containing too much information to be effectively understood and utilized. Production of overlays from computer generated thematic maps generally requires simulating the integration and feature selective filtering performed subconsciously by an image interpreter. Specifically it required the straightening of object boundaries, mapping the object in a contiguous manner, and the removal of small or unimportant features.

In Phase 1 of this project, efforts were restricted to the application of currently available software tools, eliminating the possibility of a truly effective theme filter. The need for one was, however, clear in the earliest stages of the project. Being able to "see the forest through the trees" was not a cliche, but was the definition of a major problem. Extracting forest boundaries meant elimination of high resolution microstructure within the desired feature without objectionably modifying the boundary.

The approach used in Phase 1, was basically to apply an image smoothing filter to the image as a preprocessing function. The smoothed image was then used to generate binary masks of various features after the removal of feature microstructure. Then the element extraction processes, which generally required use of microstructure and high resolution was performed on the original data. Themes generated were then restricted
to an area of the mask to eliminate misclassification errors which would otherwise be widespread and unacceptable. While the overall effort was successful, there were serious problems that were left to be solved. The most serious was that smoothing transformed all discontinuities or feature edges into smoothly flowing ramps. Generation of several of the required masks resulted in a situation where adjacent thematic features did not compliment or butt each other properly, with the area near the feature butt line, corresponding to the high slope region of the ramp, usually not being placed into the proper mask. A second problem was that the smoothing treated all objects equally as opposed to considering the spatial characteristics of individual classes. For example, narrow features such as roads, were obliterated by the heavy smoothing needed to map the forest. Lengthy work arounds were used where several degrees of smoothing were applied, but this was not an effective solution.

As a result of the Phase 1 effort, a series of requirements were identified for a post processing filter to work not on grey level images, but on the themes generated from the images. The advantages of the post processing are many and are basically related to the fact that all resolution is maintained and utilized as needed during the element extraction process. Only after the image has been reduced to the desired themes, is the filtering applied. Post processing solves the mentioned ramp problem by totally avoiding the generation of ramps, and the problem of selective filtering is handled more easily after categorization as the features have been identified.

Processing problems to be solved by a post processing filter:

- **Theme Simplification** - The severity of the filtering action should be user specifiable to achieve the desired amount of boundary smoothing. Capabilities should include extreme low pass filtering.

- **Theme Selective Filtering** - All themes are not created equal, and the filter must treat individual themes in individual manners. Methods must be provided for the restriction of growth, shrinkage or change for any particular theme.
- **Theme Butting** - The filter should be able to annihilate areas between adjacent features of interest which commonly result from classification errors or from the presence of incidental features such as shadows.

- **Contiguous Area Thresholding** - The filter should be capable of filtering by object size to allow removal of islands and/or holes from any designated theme.

In the interest of reasonable execution times, the use of box filtering techniques were nearly dictated for the theme filter. The filter must basically count the number of times each theme occurred in each local neighborhood. Each of the themes is then ranked, based on its local occurrence rate in conjunction with operator designated replacement constraints, as a candidate for being replaced and for being a possible replacement for others. The key theme (the one located in the center of the filter area at any given time) is then replaced by the theme at the top of the ranking hierarchy. In the absence of operator input constraints the key theme is replaced by the one most common in the neighborhood. To perform this operation means tabulating running areal totals for each of the themes being processed, and could amount to an excessive number of calculations with clumsy implementation. Box filtering, with its ability to operate with filters of any reasonable size without appreciable execution time penalties, is ideally suited to the task. Keeping running totals with box filters even required less computation than their usual use, which is keeping running averages. As with the running averages, the minor artifacting introduced by the use of a square spatial operator, is outweighed by the computational efficiencies.

Figure 13 shows the theme filter main program, and Figure 14 shows the data structure used. Note that the program is set up to handle eight themes, but nine different categories. Theme #0 refers to uncategorized pixels or those not in any of the eight themes. In retrospect, even another category marking multiply classified pixels or those contained in more than one theme would have been useful.
Figure 12. Theme Filter
Figure 14. Theme Filter Data Structure
The box filtering approach is not unique in theme processing, but as commonly used it is not sufficient to satisfy the theme processor requirements. Used without replacement constraints, it is not versatile enough to perform all the desired theme manipulation possibilities. Box filtering is simply an effective method of providing information to the real heart of the theme processor, the theme replacement logic.

Interactive control of the theme filter is accomplished by setting theme population thresholds which must be exceeded before a given theme is modified, with separate thresholds for theme growth and for theme shrinkage. The filter can thus be set up to treat themes uniquely when needed. Whenever the shrink threshold is passed, the key pixel is replaced by the most common growable theme. See Figure 15.

Considerable effort went into the operator interface. Its purpose is to assist the operator in the task of setting up fairly complex filtering options. The filter has numerous input parameters, whose input is tedious and has a high potential for error. Also, since the filtering action on a given theme is determined not only by the thresholds directly associated with that theme, but also by the thresholds of its possible replacements, the interaction between these input parameters is not always obvious. The interface is designed to minimize confusion along with tedious repetitive input. The filter is set up to operate in an iterative manner. Allowing either continuing with the filtering or reworking the last operation if undesirable results appear. The filter executes fast enough so that an operator can observe and analyze the result, and then rerun the filter with modified thresholds or box size to better achieve his purpose.

Figure 16 is a verification display for the interface, listing current threshold settings along with a computer generated verbal estimate of the resulting filtering action.
Figure 15. Theme Filter Replacement Logic
THE FILTER OPERATES BY TABULATING THE NUMBER OF TIMES EACH INDIVIDUAL THEME OCCURS INSIDE A MOVING RECTANGULAR AREA. FOR EACH FILTER POSITION, IF THE THEME IN THE CENTER OF THE AREA IS ALLOWED TO SHRINK, IT WILL BE REPLACED BY THE MOST COMMONLY OCCURRING AMONG THOSE ALLOWED TO GROW. WHENEVER A SUITABLE REPLACEMENT IS NOT FOUND, THE THEME IS LEFT UNALTERED.

SELECTIVE THEME REPLACEMENT IS CONTROLLED BY ENTERING THRESHOLDS EXPRESSED AS PERCENTILES OF THE FILTER AREA. A SPECIFIC THEME IS ALLOWED TO SHRINK IF THE NUMBER OF TIMES IT OCCURS IS LESS THAN IT'S SHRINK THRESHOLD, AND IS ALLOWED TO GROW IF OCCURRENCES EXCEED IT'S GROW THRESHOLD. A NEGATIVE THRESHOLD VALUE CAUSES A THEME TO BE TOTALLY REPLACED.

<table>
<thead>
<tr>
<th>THEME NUMBER</th>
<th>SHRINK IF BELOW, %</th>
<th>GROW IF ABOVE, %</th>
<th>FILTERING ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td><strong>100</strong></td>
<td><strong>0</strong></td>
<td>REPLACE</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>0</td>
<td>SMOOTH</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>0</td>
<td>SMOOTH</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>100</td>
<td>NO CHANGE</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>100</td>
<td>SHRINK ONLY</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>GROW ONLY</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>100</td>
<td>REMOVE ISLES</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>85</td>
<td>REMOVE HOLES</td>
</tr>
<tr>
<td>.8</td>
<td>5</td>
<td>95</td>
<td>CLEAN NOISE</td>
</tr>
</tbody>
</table>

CHANGE THEME THRESHOLDS? (Y)ES, OR (N)O

Figure 16. Theme Filter Operator Interface
D. Use of the Theme Filter

In the pursuit of water resources element extraction it was established that the theme filter was effective in solving a host of image categorization problems. The most severely limiting factor in its usage seems to be the limited ability of the analyst to correctly forecast the action of the filter and to be able to specify the set of interrelated thresholds needed to perform the desired function. Several common image categorization problems, along with suggested and proven solutions making use of the theme filter, are listed below:

1. Capabilities

a. General Theme Smoothing

   Application: Spatially simplify theme so that they may be depicted by line maps.

   Procedure: Run filter with default thresholds so that each theme is replaced by the locally most common. For best results, run the filter twice to insure no linear artifacts are generated by the rectangular filter.

b. Annihilation of Certain Themes

   Application: Often it is desirable to totally eliminate irrelevant themes (shadows for example).

   Procedure:
   - Enter a negative threshold for the theme to be annihilated.
   - Leave all other thresholds default

   The theme filter has a replacement flag set by entering a negative threshold. This flag stops the theme from being considered as a possible output.

c. Preserving Certain Themes While Filtering Others

   Application: Very accurately mapped themes, or sparse filament themes (roads, etc.) should not be changed.

   Procedure:
   - Set grow above threshold to 100% so that theme will not grow.
   - Set shrink below threshold to 0% so that theme will not shrink.
d. General Noise Cleaner

Application: Most spatially derived themes are improved by eliminating isolated pixels which often represent noise.

Procedure: Set thresholds for all themes to grow above 33% and shrink below 11%, then run a 3 x 3 pixel filter. The 11% threshold locates totally isolated pixels, while the 33% threshold insures there is a locally dominant class before replacement. By enlarging the filter size this operation can be used to remove larger areas (islands and holes).

e. Resolving Theme Overlap or Underlap

Application: Image classification virtually always results in some error as signatures are so unique. Problem areas often show up as unclassified or multiply classified pixels. A satisfactory solution is to remap these poorly classified pixels on a purely spatial basis.

Procedure:
- Place pixels to be resolved into a separate theme
- Set replacement flag for that theme
- Leave default thresholds for all other themes

2. Processing Example

To demonstrate the filtering capability, one of the panchromatic scenes used in the first phase of this project was recategorized with considerable improvement. The following processing sequence was followed and results are shown in Figure 17.

a. The scene was categorized on the basis of gray level and microtexture. This task was somewhat easier than with the previous effort because the filter will contiguously map locally predominant classes allowing less discriminative signatures to be used.
Figure 17. Comparison of Fort Belvoir/Woodbridge Vegetation Themes Derived by Interactive Computer and Manual Interpretation.
b. Areas categorized as shadows were eliminated by the filter based on operator interpretation. Two passes were used: (1) when a shadow occurred within a short distance to water, the shadow was changed to water producing considerable improvement in the river shoreline and bridge area, and (2) shadows not near water were mapped to the locally most common class.

c. The scene was then smoothed by a 5 x 5 pixel filter to produce a categorization comparable to what must be done by a photo interpreter. The "bare" category was restricted from change so that the roads would be preserved. See Figure 18A and 19.

d. To demonstrate severe filtering, the categories were then processed with an 11 x 11 pixel filter. See Figure 18B and 20. allows virtually any degree of filtering to be achieved with about the same execution time. In working situation, the desired amount of filtering might be determined by scene complexity, desired scale, or intended use.

e. Both of the products were reduced to line maps via Golay processing (essentially automated). Each theme was outlined, the outlines were combined to produce double width lines, and then the lines were shrunk to a single pixel width.
Figure 18. Results of Theme Filtering

Figure 18A
Theme Filter Results

Figure 18B
Simplified Themes
Figure 19. Outlines of Vegetation and Water Themes
Figure 20. Outlines of Simplified Themes
IV. RESULTS

A. Extraction of Watercourses and Associated Elements

1. Watercourses
   a. Extracting Watercourses From Panchromatic and Thermal Imagery

   The Knobbs Creek imagery was selected for analysis and development of techniques because it contained data coverage from both thermal IR and panchromatic photographic sensors. Knobbs Creek is a good test for watercourse classification in that the coverage contains a wide range of watercourse widths. It provides a difficult test as the creek is almost 100% surrounded by large trees. This does not provide, however, the best diversity of surrounds with which to test watercourse identification. The ideal test would involve a scene having many watercourses surrounded by a variety of land cover, and having a wide range of watercourse widths.

   Knobbs Creek, being surrounded by forest, creates problems for both machine and manual analysis. The creek is obscured by forest when watercourse width is less than approximately 5 meters. In these situations, analyst inferred watercourses will be necessary. Further, the determination of dry gap widths is hindered by the encroaching forest and by shadows from adjacent forest which extend across the watercourse. The shadows create an additional problem for machine analysis by completely or partially covering the watercourse (depending on watercourse width or direction). This requires the machine analyst to derive a signature for water and a signature for shadow which must be added together to enable identification of the entire watercourse. Unfortunately, the signature contains areas which are not watercourses (forest shadows within the forest).

   Eight (512 x 512 pixel) subscenes were digitized at GE DIAL from three of the 1:100,000 scale thermal IR transparencies of Knobbs Creek. Four of these were digitized from 4 in. x 4 in. areas in the transparency (ground resolution approximately 2 meter) and four from 1 in. x 1 in. areas (ground resolution approximately 0.5 meter). In addition, four (512 x 512 pixel) subscenes were digitized at GE DIAL from the 1:250,000 scale panchromatic
photographic image of Knobbs Creek. Two of these were digitized from 4 in. x 4 in. areas (ground resolution approximately 4 meter)* and two from 1 in. x 1 in. areas (ground resolution approximately 1 meter)*. These resolutions were chosen for initial tests of watercourse identification to provide an appropriate range of resolution for testing the spatial filters used in watercourse extraction.

The initial step in the investigation was to identify whether gray level density slicing would be sufficient to identify watercourses. Several problems were encountered in this step:

- This thermal IR imagery appears to contain dark and light bands which are apparently caused by automatic gain and offset in the sensor which are adjusted by scan content. This has the undesirable effect of causing forest and watercourse signatures from different areas of the scene to appear similar and thus be recognized simultaneously.

- Both thermal IR and panchromatic photographic digital images exhibit a shading across the image which cause signatures to overlap.

Gray level density slicing of either the thermal or the panchromatic photographic digital images, in order to extract watercourses, produced unacceptable results. Figures 21 and 23 show the watercourse themes extracted from the panchromatic photographic images, and similar poor results were obtained with the thermal images.

At this point, several spatial filters were applied to all digital images. The images resulting from use of these filters have high spatial frequency noise, however, which requires a subsequent smoothing operation. Best smoothing results appear to be produced with the 7 x 7 pixel moving average. Some of the spatial operations tested include: 5-pixel cross min-max texture; the original image minus a 7 x 7 pixel local average; 3 x 3 pixel Laplacian.

* In this report, the term resolution is used to describe the actual dimensions of the smallest resolvable objects in an image.
Figure 21. Watercourse Extraction via Level Slicing The 1-meter Resolution Panchromatic Photographic Image.
Figure 22. Watercourse Identification via Joint Use of the 1-meter resolution Panchromatic Photographic Image and the Derived Texture Image (Smoothed).
Figure 23. Watercourse Extraction via Level Slicing The 4-Meter Resolution Panchromatic Photographic Image
Results when using the spatially processed images jointly with the original digital images (i.e., two dimensional classification) can be summarized as follows:

- Best results for 1 meter resolution panchromatic photographic and 0.5 meter resolution thermal IR images were obtained with the 5-pixel Cross min-max texture operator.
- Best results for 4 meter resolution panchromatic photographic and 2 meter resolution thermal IR images were obtained with the original image minus the 7 x 7 local average.

Figure 22 shows the result of watercourse identification using the 1 meter resolution panchromatic photographic image and its texture version. In Figure 24, watercourse identification is shown for the case when 4 meter resolution panchromatic photographic image and the smoothed result of image minus a 7 x 7 pixel local average are used. Figures 23 and 26 show the actual location of the watercourse. Comparison of actual and machine-extracted watercourses indicates shadow areas in both scenes which are falsely included in the watercourse theme.

Similar results were obtained with the digitized thermal imagery.

2. Extracting Watercourses from Radar Imagery

Four subscenes of 512 x 512 pixels depicting X-Band vertical and horizontal polarization; and L-Band vertical and horizontal polarization were made. Figures 25, 26, 27, and 28 are illustrations of the subscenes as displayed on the cathode ray tube (CRT). The data are shown inverted to a negative. Geographic scene locations are shown in Figure 29.

In Figures 30, 31, 32 and 33 theme results were obtained by sensity slicing the data. Figure 34 subscene is the same as Figure 25. It is repeated to allow for comparison relative to the processes subscenes in Figures 35, 36 and 37. In Figure 35, the subscene illustrates the application of the 7 x 7 moving average operator on the X-Band horizontal polarization data. Figure 36 shows the derived associated microtexture image derived from X-Band horizontal polarization data.
Figure 24. Watercourse Identification via Joint Use of the 4-Meter Resolution Panchromatic Photographic Image and the Result of the Image Minus 7 x 7 Pixel Local Average.
Figure 25 - X-Band Horizontal Polarization Subscene
Figure 26 - X-Band Vertical Polarization Subscene
Figure 27 - L-Band Horizontal Polarization Subscene
Figure 28 - L-Band Vertical Polarization Subscene
Figure 29. SAR Data and Subscene Locations.
Figure 30. Water Theme Sliced From X-Band SAR w/Horizental Polarization.
Figure 31. Water Theme Derived From X-Band SAR w/Horizontal Polarization.
Figure 32. Water Theme Derived From X-Band SAR w/Horizental Polarization.
Figure 33. Water Theme derived from L-Band SAR w/vertical polarization.
Figure 34 - X-Band Horizontal Polarization Subscene (Same as Figure 1)

Figure 35 - X-Band Horizontal Polarization Data Smoothed by a 7x7 Pixel Moving Average Operator

Figure 36 - Associated Microtexture Image

Figure 37 - Microtexture Image Data Smoothed by a 7x7 Pixel Moving Average Operator
Figure 37 shows the results of smoothing the microtexture image data with a 7 x 7 pixel moving average operator. In Figure 38, X-Band SAR horizontal polarization data and the associated microtexture image are shown categorized by a two dimensional partitioning classifier. This procedure appears to provide clearer results than density slicing the SAR data only. Figure 39 illustrates the categorization of water courses and water bodies using the two dimensional partitioning classifier with the X-Band SAR horizontal polarization data and the texture data smoothed by a 7 x 7 pixel moving average operator. It appears that another degree of clarification is achieved with this method. In Figure 40, X-Band SAR horizontal polarization data are smoothed by a 7 x 7 pixel moving average operator and density sliced into a theme for water courses and water bodies. Apparently this procedure provides the best definition of watercourses and water bodies.

3. Watercourse Extraction Using Combinations of Dissimilar Images

A data set of panchromatic, low resolution Thermal IR and SAR (two flight directions) images were digitized at 5 meter resolution and registered for Little River and New Begun Creek water courses. Different combinations of data from the set, including microtexture derived data (from panchromatic images) were used for the extraction of water course themes. The low resolution Thermal IR data did not have a significant contribution to the derivation of the watercourse themes. From the analysis of a combination of the panchromatic images and the microtexture (derived from the panchromatic images) data a conflict between shadows and water existed. Radar imagery was used to resolve some of the shadow conflict. Derived watercourse themes for Little River are shown in Figures 41, 42 and 43.

A comparison of watercourse themes which were derived from different combinations of Chapel Creek (upper Little River area) panchromatic, radar, and the microtexture data (derived from the panchromatic images) are shown in Figures 44, 45, 46 and 47.
Figure 38. Water Theme Derived from w/Horizontal Partitioning of X-Band SAR w/Horizontal Polarization and Associated Microtexture Image Data.
Figure 39. Water Theme Derived From 2-Dimensional Partitioning of X-Band SAR w/Horizontal Polarization Data and Textured Data Smoothed By a 7 x 7 Pixel Moving Average Operator.
Figure 40. Theme Derived From the 7 x 7 Pixel Smoothed Version of the X-Band SAR Image w/Horizontal Polarization.
Figure 41. Little River Watercourse Theme Derived from Panchromatic Photography and SAR Data and from the Microtexture Version of the Photography.
Figure 42. Smoothed Version of Figure 41.
Figure 43. Edged Version of Figure 42
Figure 44. Chapel Creek Watercourse Theme Derived from Panchromatic Photography and SAR data.
Figure 45. Chapel Creek Watercourse Theme Derived from Panchromatic Photography and SAR Data and from the Smoothed Microtexture Version of the Photography.
Figure 46. Smoothed Version of Figure 45.
Figure 47. Edged Version of Figure 46.
4. Watercourse Extracting Using Panchromatic Imagery Digitized of Various Scales

A section of the panchromatic image of the Chapel Creek watercourse was digitized at multiple scales to determine what resolution of the derived micro-texture data contains the most separable watercourse class. Preliminary results indicate that panchromatic resolution between 1 and 2 meters yields microtexture data which best discriminates the watercourse class. This also, to some degree, verifies results for vegetation extraction where similar panchromatic resolutions yielded micro-texture data which contained best separability of vegetation classes.

- Alignment of all Water Bodies
  Given a suitable thematic classification of a watercourse, the alignment or centerline can be extracted by the methods detailed in Section III. Figure 12 shows the alignment of watercourse as extracted by the Golay method.

- Shore Alignment of all Water Bodies
  This is a relatively easy task once water bodies have been categorized. In fact, the recommended procedure is tolerant of classification errors of the omission type and required only a mediocre water body theme. The theme is filtered as required to generate a contiguous map by using the theme filter or a similar operation, and is then outlined by the Golay process. Figure 47 shows a sample alignment theme.

B. Extraction of Dry Gaps and Associated Elements

1. Dry Gaps

Much of the effort involved in producing watercourse factor overlays deals with dry gap width, which is defined as the distance between stream bank shoulders. These width measurements are not usually derivable from a single image. The Terrain Analysis Procedural Guide for Drainage and Water Resources lists two methods of determining dry gap i.e (1) use of topographic information to locate lines of high elevation gradient along the stream bed and (2) locate the shoulder by photo interpretation of stereo pair. While there are interesting possibilities for implementing either of these approaches on an interactive system, no suitable imagery was available resulting in development efforts.
Without elevation information, the extraction of dry gaps is a task best left to the interpreter because of the large amount of judgement required. Dry gap outlines can be quickly entered into digital storage by using a pen/tablet or cursor in conjunction with an image display. Admittedly this is an inelegant solution, but it is also practical. Dry gaps can be located more easily on the basis of vegetation and landform interpretation, than by any unique machine extractable methods.

Once a watercourse and its associated dry gap have been satisfactorily mapped into themes by whatever means, interactive or automated, generation of factor overlays containing watercourse delineation and categorization still entails a series of tasks which are labor intensive and fairly tedious. Tasks which could be done effectively by computer include, the classification of water course segments, dry gap width, encoding of water course delineations by class, and the location of terminal points.

Even with the unfeasability of extracting dry gaps from the available imagery, it was felt that interactive computer methods could be of value in the extraction of these elements rather than not investigate the elements, some analysis was performed with a manually extracted dry gap as a starting point.

To demonstrate some of the potential tasks, a panchromatic image of Tennessee was used. The scene contained an unusually diverse set of watercourses with dry gap widths ranging from less than 3 to greater than 50 meters. Both the watercourse outline and dry gap outline were classified by manual means with an operator moving a joystick controlled cursor over the image.

While the resulting dry gap theme probably contains errors, it is typical of what can be done on the basis of land form and vegetation analysis with no available elevation data. Figure 48 and 49 show the results.

2. Categorizing Watercourse by Dry Gap Width
A sequence which could lend itself to automated processing can be used with watercourse and dry gap themes stored. The first step is to employ a pixel stripping algorithm such as the Golay processor to locate the center line (delineation) of both themes. Figure 50 and 51 show the results.
Figure 48. Manually Interpreted Watercourse
Figure 49. Manually Interpreted Dry Gap
Figure 50. Skeletonized Watercourse
Figure 51. Skeletonized Dry Gap
for this particular image. These results have been touched up to correct some of the previously discussed problems introduced by the Golay skeletonizing algorithm as it is currently implemented on GE DIAL. A correction is needed to the algorithm which assures skeletonizing to a single pixel width line with no loss of connectivity.

The next step basically measures dry gap width along the entire water course by reapplying the Golay process to strip from the dry gap theme, annular rings of pixels corresponding to the desired width increments. In contrast to the previous process, the dry gap is not skeletonized but is allowed to disappear as the stripping continues. Five sequential applications of stripping can be used to yield a set of themes showing areas of dry gap greater than 3, 10, 18, 25 and 35 meters.

These themes are then logically combined with the dry gap center line producing a series of center line segments where each segment maps a dry width interval. Figure 52 is a flow chart describing the algorithm segments.

The desired final result is, of course, not a categorized dry gap center line, but is a categorized watercourse center line. The final steps of the sequence address this problem. A new set of themes must be generated, which categorize the entire area covered by the dry gap into segments by width interval. This is accomplished by applying the theme filter to the dry gap and its center line segments. Filter constraints are set up so that the dry gap theme is totally replaced with the theme corresponding to the closest center line segment. Finally, this set of themes is logically combined with the watercourse center line (or boundaries) to produce the desired categorization. See Figures 53 and 54.

3. Terminal Points of Watercourse Segments
At this point in the suggested processing sequence, we have supossed performed a successful categorization of the watercourse based upon its associated dry gap width. The result is a series of themes, each containing segments of the watercourse delineation (centerline) corresponding to a given dry gap width increment. From any of these themes it is a simple matter to locate end points of the segments.
I GENERATE DRY GAP THEME $A_0$

SKELETOIRIZE THEME $A_0$ TO DRY GAP CENTERLINE VIA GOLAY PROCESSOR

WHILE DRY GAP THEME GT 50. M DO

CALCULATE NEXT WIDTH INCREMENT W

SHRINK DRY GAP THEME $A_n$ BY REMOVAL OF AN ANNULUS w/2 THICK VIA GOLAY PROCESSING

LOGICALLY AND CENTERLINE THEME WITH LOST AREA TO GET SEGMENT CATEGORIZED BY WIDTH

GENERATE A SET OF THEMES SUBDIVIDING THE AREA OF $A_0$ BY GROWING THE CENTER LINE SEGMENTS INTO THIS AREA (THEME PROCESSOR)

STOP

Figure 52. Categorizing a Dry Gap By Width
Figure 53. Dry Gap Categorized by Width and Displayed as Polygons
Figure 54. Combining Watercourse Center Line With Dry Gap Categories. Performing a Logical Intersection Produces Watercourse Segments Categorized Width.
Golay processing is a very straightforward means of finding segment end points. The procedure is to operate on the desired width increment theme by deleting all pixels with surrounds containing more than one pixel. Using the numbering convention in Figure 10, surrounds 2 thru 4 would be set for deletion. The output theme contains only the end points of connected segments in the input theme. The operation is completed in one pass over the image and is independent of the number of segments being processed.

For the end points of watercourse segments with less than three meter dry gaps an additional operation may be desirable. In this case, both terminal points of the segments are not used in factor overlays, only the points where the watercourse transitions to greater than three meter dry gap width. The same Golay operation applied to the entire watercourse delineation serves to locate points where the stream disappears. Subtracting these stream end points from the three meter segment end point leaves only the desired transitional points.

4. Encoding Watercourse Delineation

After categorization, the encoding of watercourses by line dashing, can and should be done by automated means. All the dry gap width symbols specified in the Terrain Analysis Procedural Guide for Drainage and Water Resources can be generated from a series of points equally spaced along the watercourse center line. Figure 55 is an algorithm (untested) which could be used to generate these points and Figure 56 lists methods by which the dot pattern could be used to encode lines with the desired patterns. Note that the feature encoding procedures are all raster (line by line) algorithms as opposed to line following algorithms. The raster processing approach has the advantages of neither being affected by image size, complexity, nor an occasional broken or non contiguous watercourse.
Figure 55. Raster Operating Line Dotting Algorithm
<table>
<thead>
<tr>
<th>DRY GAP WIDTH (meters)</th>
<th>ENCODING PROCEDURE</th>
<th>RESULT</th>
</tr>
</thead>
</table>
| $W < 3$                | 1. GENERATE CLOSELY SPACED DOTS FROM SINGLE PIXEL WIDTH WATER COURSE CENTER LINE.  
                         | 2. GROW DOTS INTO SMALL CIRCULAR AREAS VIA GOLAY PROCESSING | ...... |
| $3 < W < 10$           | 1. GENERATE WIDELY SPACED DOTS FROM CENTER LINE  
                         | 2. GROW DOTS INTO CIRCULAR AREAS VIA GOLAY PROCESSING  
                         | 3. OUTLINE CIRCULAR AREAS VIA GOLAY PROCESSING  
                         | 4. SUBTRACT OUTLINE FROM CENTER LINE | ...... |
| $10 < W < 18$          | 1. GENERATE MODERATELY SPACED DOTS FROM CENTER LINE  
                         | 2. GROW DOTS TO SMALL CIRCULAR AREAS | ...... |
| $18 < W < 25$          | 1. USE UNMODIFIED CENTER LINE | ...... |
| $25 < W < 35$          | 1. USE PROCEDURE FOR $10 < W < 18$  
                         | 2. GROW DASHED LINE VIA GOLAY PROCESSING | ...... |
| $35 < W < 50$          | 1. GROW CENTER LINE VIA GOLAY PROCESSING | ...... |

Figure 56. Encoding Watercourse Delineation
V. CONCLUSIONS

Phase 1 and Phase 2 of the study have concentrated on the feasibility and use of man-machine interactive digital processing techniques for the extraction of terrain data elements from aerial imagery. Interactive digital techniques were developed for vegetation/land cover boundary extraction and for extraction of forest-related data elements. These techniques included the use of spatial crown and texture operators.

Applications of these operators during Phase 1, enabled the extraction of feature roughness at several scales; extraction of vegetation types, wet areas and water bodies; and the determination of forested areas, forest canopy and locations of individual trees. The overall success of Phase 1 suggested the need to further test the operators relative to water resources data element and to improve and test the spatial crown and texture operators at different scales.

The development and coding of software included programs in image resolution and scale changing, feature skeletonizing, and theme smoothing. These programs are implemented on the GE DIAL system, primarily in Fortran language. Limited testing of the programs have been completed in Phase 2 to demonstrate usefulness. The implementation of these programs on systems similar to GE DIAL, is feasible.

Extraction of water resources data elements requires outlining features such as vegetation boundaries and shore line delineation of water bodies. Through spatial binary processing, or Golay processing, features were quickly and flawlessly outlined.

A slight variation of outlining, line stripping, provided a feasible, but somewhat lengthy (multiple pass) mean of categorizing watercourses by the width of their manually mapped dry gaps. Terminal points of watercourse segments can also be located by one pass Golay processing. Efforts to delineate watercourses, the prime water resources element, were only partially successful. The seemingly simple problem was to connect separated pixels of a watercourse theme, and then shrink or skeletonize the feature into a single pixel width centerline. Satisfactory delineation was achieved for certain cases but no universally applicable techniques were found.
The production of usable factor overlays, which are the desired result of terrain element extraction was addressed and handled by writing what is thought to be an important theme filtering program. Its basic purpose was to perform spatial low pass filtering on a categorized image as required to generate overlays which resemble those made by photointerpretation and which are spatially simple enough to be combined into complex overlays. Provisions are made to restrict change of critical or exceptionally accurate themes and to remap on a spatial basis those which are incidental or plagued with categorization errors.
VI. RECOMMENDATIONS

In the performance of Phase 2, areas were identified as candidates for further investigation. Generally, these candidates include work described as testing of existing software/techniques and development of new software/techniques. The recommended areas for future work are:

- Continue to perform sufficient testing of techniques developed in both Phase 1 and Phase 2. The testing should include various sites and ground data to radiate the techniques.
- Limited results were obtained for delineation of line features, i.e., streams and roads. It is believed that the existing Golay algorithm, an already invaluable tool, could be revised and made to perform error free skeletonizing.
- Further testing of the theme filtering program is suggested. This testing is relative to the production of factor overlap, i.e., a comparison of manual interpretation vs. computer. The theme filtering program has proven valuable as a cosmetic noise cleaner and as a tool in the feature extraction process. It allows satisfactory results to be filtered from otherwise unacceptable themes. Although not tested in this respect, it is also seen as a promising means of digitally combining selected themes into complex overlays. While highly effective, the program is somewhat cumbersome to set up, and the operator interface should be rewritten so that commonly used filtering options are menu selectable. With the minor change the program is recommended for general interactive use.
- To date, processing of imagery data has been restricted to small geographical areas. The testing of developed techniques from Phase 1 and 2 to mosaicked images is needed relative to the practical uses of terrain analysis. It is recommended that such work be performed to allow for the expansion of the techniques to larger geographical areas.
APPENDIX A

IMAGE RESOLUTION AND

SCALE CHANGING PROGRAM
IMAGE SCALE AND RESOLUTION CHANGING

THIS PROGRAM PROVIDES A MEANS OF MODIFYING IMAGE RESOLUTION TO SIMULATE ANY SENSOR WITH LOWER RESOLUTION THAN THAT INHERENT IN THE ORIGINAL IMAGE DATA, AND TO PERFORM GEOMETRIC SCALING.

ITS MAIN PURPOSE IS TO MODIFY IMAGES IN A MANNER THAT ALLOWS STANDARD SPATIAL OPERATORS TO BE APPLIED TO ANY IMAGE AT A VARIETY OF SCALES; THIS EXTRACTS SPATIAL INFORMATION AT A VARIETY OF FREQUENCIES WITH MINIMUM COMPUTATION. THE SUGGESTED PROCESSING SEQUENCE IS TO USE THIS PROGRAM TO UPGRADE RESOLUTION TO THE POINT WHERE A DESIRED FEATURE DISPLAYS UNIQUE SPATIAL PROPERTIES, APPLY A STANDARD OPERATOR, AND THEN RESCALE THE OUTPUT BACK TO ITS ORIGINAL SIZE FOR SUBSEQUENT FEATURE EXTRACTION.

FORTRAN PROGRAM MODULES:

BSWINDO ---- A DRIVER THAT CALLS THE MODULES NECESSARY TO PERFORM ANY SELECTED SCALING OPERATION.

WINDO ---- PROVIDES AN INTERACTIVE METHOD OF DEFINING INPUT AND OUTPUT IMAGE SECTIONS. MAKES USE OF THE IMAGE-100 JOYSTICK CONTROLLED RECTANGULAR CURSOR.

SELECT ---- ACCEPTS OPERATOR SELECTIONS OF INPUT AND OUTPUT IMAGE PLANNES OF CHANNELS IN THE IMAGE 100 REFRESH MEMORY.

NMINDO ---- PERFORMS IMAGE MAPPING VIA NEAREST NEIGHBOR INTERPOLATION.

AMINDO ---- PERFORMS IMAGE MAPPING VIA COMPUTATION OF LOCAL APEAL AVERAGES. USED TO SMOOTH OR SHRINK IMAGES.

CMINDO ---- PERFORMS IMAGE MAPPING VIA A SUBPIXEL CONTEXT DEPENDANT INTERPOLATION. USED TO ENLARGE IMAGES.

PDF-11 ASSEMBLER MODULES:

UNSCLAL ---- CALLED BY NMINDO TO REMAP PIXELS ALONG A SCAN LINE.

COLSUM ---- CALLED BY AMINDO TO MAINTAIN RUNNING SUMS ALONG IMAGE COLUMNS.

ROWSUM ---- CALLED BY CMINDO TO COMPUTE RUNNING AVERAGES ALONG IMAGE ROWS.

IMAGE 100 LIBRARY SUBROUTINES:

FIND ---- FREE FORMAT INPUT ROUTINES

IPK ---- CURSOR CONTROL ROUTINES

IPU
INW — VIDEO MEMORY ACCESS ROUTINES
OUTPUT — TERMINAL CONTROL ROUTINE

KEY VARIABLES:

IX1, IY1, IX2, IY2 — COORDINATES OF UPPER LEFT AND LOWER RIGHT CORNERS OF 
THE INPUT IMAGE WINDOW.
OX1, OY1, OX2, OY2 — COORDINATES OF UPPER LEFT AND LOWER RIGHT CORNERS 
OF THE OUTPUT IMAGE.
INCHS(5), OUCHS(5) — CHANNEL ASSIGNMENT ARRAYS. INCHS(N) GETS PROCESSED 
INTO OUCHS(N).
INSERT — A FLAG THAT DETERMINES IF THE OUTPUT IMAGE IS TO 
FULLY REPLACE PREVIOUS OUTPUT CHANNEL CONTENTS, OR 
IS TO BE INSERTED INTO A WINDOW AREA.

MAG — MAGNIFICATION FACTOR
BUF(40) — ALPHA NUMERIC STRING FOR OPERATOR INPUTS

DX(512), DY(512) — ISOGRATIC MAPPING TABLES. TABLE INDEX REFERS TO 
OUTPUT PIXEL LOCATIONS, AND TABLE CONTENTS REFER TO 
CORRESPONDING INPUT PIXEL LOCATIONS.
SUBROUTINE BGWINDO

GENERAL CHANNEL TO CHANNEL WINDOWING PROGRAM

IMPLICIT INTEGER(A-G,S-Z)

DIMENSION BUF(40)

LOGICAL INSERT

COMMON AWINO/IX1,IX2,IY1,IY2,OX1,OX2,OY2,RMAG,INSERT

COMMON /CHANL/ INCHS(5), OUCHS(5)

DATA UTY,'/','Y','/','N','/','Y','/','1','/','C','/','1','/','R','/','R','/'

CLEAR DISPLAY

CALL OUTPUT(27,12,7)

DETERMINE WINDOW MAPPING

CALL WINDOW

CALL OUTPUT(27,12,7)

WRITE(UTY,999) IX1,IX2,IY1,IY2,OX1,OX2,OY2,RMAG

999 FORMAT(7I13,1X,1F15.2)

SELECT CHANNELS

CALL SELECT(INCHS,OUCHS)

ASK FOR MAPPING TECHNIQUE

TECHNI-N

IF(RMAG.EQ.1.0) GO TO 120

WRITE(UTY,1010)

1010 FORMAT('$(C)MPUTED OR (N)EAREST NEIGHBOR INTERPOLATION $')

READ(UTY,1020) Buf

1020 FORMAT(4062)

CALL FRONT(Buf)

IF((Buf(1).NE.N).AND.(Buf(1).NE.C)) GO TO 101

1015 TECHNI=BUF(1)

120 CONTINUE

ASK FOR OUTPUT METHOD

DELTA=X2-OX1

DELTA=Y2-OY1

INSERT=.FALSE.
C IF((DELTA.X GT 510) .AND. (DELTA.Y GT 360)) GO TO 130
     WRITE(UTY,1290)  
     1290 FORMAT('** INSERT OR (R)PLACE OUTPUT **')
     1020 BUF
     1296 CALL FRONT(BUF)
     IF((BUF(1).NE.1) .AND. (BUF(1).NE.1R)) GO TO 129
     130 CONTINUE

      DETERMINE PROCESSING CASE

      NEAREST NEIGHBOR CASE?
      IF((TECHNI.ED.M) .OR. (RMAG.EQ.1.0)) CALL NNANDO

      INTERPOLATION CASE?
      IF((TECHNI.ED.C) .AND. (RMAG.EQ.1.0)) CALL CANDO

      AVERAGING CASE?
      IF((TECHNI.ED.C) .AND. (RMAG.LT.1.0)) CALL AAANDO

      RETURN
      STOP
      END
SUBROUTINE WINDOW

PROCESSING WINDOW DEFINITION VIA CURSOR POSITIONING

IMPLICIT INTEGER(A-Z)
DIMENSION BUF(40)
DIMENSION INTR(5), OUTUR(5)
REAL FLOAT
COMMON /WIND/ IX1, IY1, IX2, IY2, DX1, DY1, DX2, DY2, RMAG, INSERT
DATA UTY/6/
DATA V'N', V'Y'/
DATA NULL'/1/

ASK OPERATOR FOR MAGNIFICATION
WRITE(UTY,5000)
9000 FORMAT('/**ENTER MAGNIFICATION (EG. 2.1)*/')
1000 PRINT(UTY,1000) BUF
1000 FORMAT(402)

IF(BUF(1),ED.X) CALL EXIT
P=0
CALL INTIFP(BUF,50,NUMER)
CALL INTIFP(BUF,50,DENOM)

IF(NUMER,ED.0) GO TO 30
IF(DENOM,ED.0) DENOM=1
RMAG=(FLOAT(NUMER))/(FLOAT(DENOM))

LOCATE WINDOWS WITH CURSOR
IF(RMAG,LT,1.0) GO TO 300
CASE #1
WINDOW WILL BE ENLARGED

ODX=512
ODY=512
OUTUR(2)=256
OUTUR(3)=ODX/2
OUTUR(4)=256
OUTUR(5)=ODY/2

WRITE(UTY,2010) ODX,ODY
2010 FORMAT('**OUTPUT WINDOW = ',13,' X ',13,' */')
WRITE(UTY,1000) BUF
IF(BUF(1),ED.X) CALL EXIT
IF(BUF(1),ED.NULL) GO TO 210
ELSE REPLACE NULL SIZE OUTPUT DEFAULTS

Note: This module must
be replaced for non
Image 100 implementation.
P=0
CALL INTFF(P,BUF,80,IDX)
CALL INTFF(P,BUF,80,IDY)
IF((IDX.LE.0).OR.(IDX.GE.512)) GO TO 201
IF((IDY.LE.0).OR.(IDY.GE.512)) GO TO 201

C
SIZE CURSOR FOR REDUCED OUTPUT WINDOW

0046
OUTCUR(3)+IDX/2
OUTCUR(5)+IDX/2
CALL INK(OUTCUR)
WRITE(UTY,2050)
0060
FORMAT("POSITION CURSOR FOR OUTPUT WINDOW, THEN CR ")
0061
READ(UTY,1000)NOIN
0062
CALL IRK(OUTCUR)
0063
210 CONTINUE

C
SIZE CURSOR TO INPUT AREA

0064
IDX=(FLOAT(IDX))/2
IDY=(FLOAT(IDY))/2
0066
CALL INK(INCUR)
0067
INCUR(3)=IDX/2
0068
INCUR(5)=IDX/2
0069
CALL IRK(INCUR)
0070
WRITE(UTY,2150)
0071
FORMAT("POSITION CURSOR FOR INPUT WINDOW, THEN CR")
0072
READ(UTY,1000)NOIN
0073
CALL IRK(INCUR)
0074
GO TO 400
0075
300 CONTINUE

C
CASE #2
WINDOW WILL BE REDUCED

0066
IDX=512
0067
IDY=512
0068
INCUR(2)=256
0069
INCUR(3)=IDX/2
0070
INCUR(4)=256
0071
INCUR(5)=IDX/2
0072
WRITE(UTY,3010)IDX,IDY
0073
FORMAT("INPUT WINDOW = '"',13,'' X '"',13,'' ")
0074
READ(UTY,1000)IDX
0075
IF(BUF(1).EQ.X) CALL EXIT
0076
IF(BUF(1).EQ.NULL) GO TO 310
0077
C
ELSE REPLACE DEFAULTS:
P=0
CALL INTFF(P,BUF,80,IDX)
CALL INTFF(P,BUF,80,IDY)
IF((IDX.LE.0).OR.(IDX.GE.512)) GO TO 301
IF((IDY.LE.0).OR.(IDY.GE.512)) GO TO 301

C
SIZE CURSOR FOR SMALLER INPUT WINDOW

0086
INCUR(3)=IDX/2
0087
INCUR(5)=IDX/2
CALL IK(INCUR)
WRITE(UTY,3050)
FORMAT('$POSITION CURSOR FOR INPUT WINDOW, THEN CR 
  
  READ(UTY,1000) NOTHIN
CALL IK(INCUR)
CONTINUE

SIZE CURSOR TO OUTPUT AREA

DOX=(FLOAT(IDX))%RAG
ODY=(FLOAT(ODY))%RAG
CALL IK(OUTCUR)
OUTCUR(3)=DOX/2
OUTCUR(5)=ODY/2
CALL IK(OUTCUR).
WRITE(UTY,3150)
FORMAT('$POSITION CURSOR FOR OUTPUT WINDOW, THEN CR 
  
  READ(UTY,1000) NOTHIN
CALL IK(OUTCUR)
CONTINUE

SET WINDOWS IN COMMON

IX1=INCUR(2)-IDY/2
IY1=INCUR(4)-IDY/2
IX2=IX1+IDY
IY2=IY1+IDY

DX1=OUTCUR(2)-DOX/2
DY1=OUTCUR(4)-ODY/2
DX2=DX1+DOX
DY2=DY1+ODY
RETURN
END
SUBROUTINE SELECT(INCHS,OUTCHS)

C
C CCCCCCCCCCCCCCCCCC GENERAL & ELECTRIC CCCCCCCCCCCCCCCCCCCCC
C C SETS UP ARRAYS OF INPUT AND TARGET CHANNELS
C CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC TRW 11/80
C
C
C IMPLICIT INTEGER(A-0,S-2)
C
C DIMENSION BUF(40)
C DIMENSION INCHS(5),OUTCHS(5)
C
C DATA UTY,/, reading
C DATA INCHS/5/
C DATA OUTCHS/5/
C DATA Y,N,X,",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",",","
SUBROUTINE NNUNDO

----------
SCALES AND LOADS A WINDOW VIA NEAREST NEIGHBOR INTERPOLATION
----------

IMPLICIT INTEGER (A-0, S-2)
REAL FLOAT
DIMENSION DX(512), DY(512)
BYTE IB(512), OB(512), ZEROS(512)
BYTE TAB(256)
LOGICAL INSERT
COMMON /WINDOW/ IX1, IY1, IX2, IY2, OX1, OY1, OX2, OY2, RMAG, INSERT
COMMON /CHANNEL/ INCHS(S), OUCHS(S)

----------

SET UP MAPPING FROM INPUT TO OUTPUT COORDINATES

RXINC=FLOAT(IX2-IX1+1)/FLOAT(OX2-OX1+1)
RYINC=FLOAT(IY2-IY1+1)/FLOAT(OY2-OY1+1)

DO 20 J=OY11,OY21
   DO(J)=RYINC*XI+0.5
   K=K+1
   CONTINUE

OY11=OY11+0
OY21=OY21+0
K=0
IF(RYINC.LE.1.0) K=-1.0/(RYINC+2.0)
L=IY1
IF(RYINC.GT.1.0) L=FLOAT(L)+RYINC/2.0

DO 20 J=OY11,OY21
   DO(J)=RYINC*XI+0.5
   K=K+1
   CONTINUE

OX11=OX11+1
OX21=OX21+1
K=0
IF(RXINC.LE.1.0) K=-1.0/(RXINC+2.0)
L=IX1
IF(RXINC.GT.1.0) L=FLOAT(L)+RXINC/2.0

DO 30 J=OX11,OX21
   DO(J)=RXINC*XI+0.5
   K=K+1
   CONTINUE

LENGTH=OX2-OX1

LOOP THRU THE SELECTED BANDS
DO 500 BPTR = 1,5
BNDI = INCHS(BPTR)
BND0 = OUCHS(BPTR)

SKIP UNWANTED BANDS

IF(BNDI .EQ. 0) GO TO 450
ELSE PROCESS LINES

SET UP TABLE LOOKUP

MULTIP = 1
DO 205 J = 0, 255
TEMP = MULTIP * J
TAB(J+1) = TEMP
205 CONTINUE

IF INSERT) GO TO 211
ELSE ERASE LINES ABOVE OUTPUT WINDOW

DO 210 LINE = 0, DY11 - 1
CALL INV(BND0, LINE, ZEROS)
210 CONTINUE
END IF

211 CONTINUE
LASTY = -1
DO 400 LINE = DY11, DY21
NEBDY = DY(LINE)

SKIP PROCESSING FOR DUPLICATED LINES

IF((NEBDY .EQ. LASTY) .AND. (.NOT. INSERT)) GO TO 350
ELSE PROCESS NEW OUTPUT LINE

CALL IRV(BNDI, NEBDY, 1B)

MOVE DATA TO OUTPUT BUFFER

IF (INSERT) CALL LNSCAL(1B, TAB, OB, DX, DX1, LENGTH)
END IF

350 CONTINUE
WRITE OUTPUT LINE TO IAC

CALL INV(BND0, LINE, OB)
LASTY = NEBDY

400 CONTINUE

IF (INSERT) GO TO 411
ELSE ERASE LINES BELOW OUTPUT WINDOW

DO 410 LINE = DY21 + 1, 511
CALL INV(BND0, LINE, ZEROS)
410 CONTINUE
END IF

411 CONTINUE
SUBROUTINE AWFIND

SCALES AND LOADS A WINDOW BY AVERAGING LOCAL AREAS

IMPLICIT INTEGER (A-H,I-L,Z,R,0)
REAL FLOAT

DIMENSION DX(512), DY(512)
DIMENSION SUMB(512)

BYTE OUTB(512), ZEROS(514)
BYTE NVIDIA, S12
BYTE TAB(256)
BYTE NEWB(512), OLDB(512)

LOGICAL INSERT

EQUIVALENCE (ROWSB(2), ZEROS(514))

COMMON /AWIND/ IX1, IY1, IX2, IY2, OX1, OY1, OX2, OY2, RMAG, INSERT
COMMON /CHAIN/ INCHS(5), OUCH(5)

DATA MINLIN/-4/, MAXLIN/509/
DATA NUMPIX/S12/

SET UP MAPPING FROM INPUT TO OUTPUT COORDINATES

FXINC=FLOAT(OX2-OX1+1)/FLOAT(OX2-OX1+1)
RYINC=FLOAT(OY2-OY1+1)/FLOAT(OY2-OY1+1)

DO 20 J=0, OY2-1, OY1+1
   DO 20 K=0, OX2+1-1
      L=1
      CONTINUE

DO 20 J=0X1, OX2
   DO 20 K=0, OX2
      L=1
      CONTINUE

SET OUTPUT LENGTH, AND NUMBER OF COLUMNS & ROWS TO BE AVERAGED
C
LEN=OX2-OK1
NCDL=1.0-RAG
NROW-NCOL,ROW
C
MULTI=1
DO 41 J=0,255
  TEMP=MULTI*J
  TEMP=MIN(TEMP,255)
  TAB(J)=TEMP
41 CONTINUE
C
LOOP THRU THE SELECTED BANDS
DO 500 BPTK = 1,5
  BNDT = INCHG(BPTK)
  BND0 = OUTCHG(BPTK)
  SKIP UNWANTED BANDS
  IF(BND1.EQ.0) GO TO 450
  ELSE PROCESS LINES
    IF(INSERT) GO TO 211
    ELSE ERASE LINES ABOVE OUTPUT WINDOW
    DO 210 LINE = 0,(Y11-1)
      CALL IIV(BND0,LINEL,ZEROS)
      CONTINUE
    END IF
    CONTINUE
  210
  211
  TOPAUL=DY(Y11)-NROW+NROW/2
  TOPAUL=MAX(TOPAUL,MINLIN)
  DO 400 LINE=Y11,Y21
    NEBDY = DY(LINE)
    COMPUTE RUNNING COLUMN SUM ON INPUT LINES UNTIL
    SUMS ARE CENTERED OVER NEXT INPUT LOCATION NEBDY
    BOTAUL=DY(LINE)+NROW/2
    DO 390 NEULIN=TOPAUL,BOTAUL
      COMPUTE LINE LIMITS OF INPUT AREA
      OLDLIN=NEULIN-NROW
      OLDLIN=MAX(OLDLIN,MINLIN)
      OLDLIN=MIN(OLDLIN,MAXLIN)
      NEULIN=MIN(NEULIN,MAXLIN)
    390 CONTINUE
    UPDATE RUNNING COLUMN SUMS IN SUMB
    COUNT=COUNT+1
    IF(COUNT.GT.NROW) CALL IIV(BND1,OLDLIN,OLDB)
compute column sums

CALL IRV(BNDL, NNEW1, NNEW)
CALL COLSUM(NNEW, OLDB, SUMB, NUMPIX)

SET STARTING INPUT LINE FOR FUTURE LOOP

TORDUL = BOTAUL + 1
CONTINUE

PROCESS OUTPUT LINE

IF (INSERT) CALL IRV(BNDL, LINE, OUTB)
CALL ROWSUM(SUMB, ROWDS, NUMPIX, NCOL, AREA)
CALL UNSCAL(ROWDS, TABA, OUTB, DX, CK1, LEN)
CALL IVJ(BNDL, LINE, OUTB)

IF (INSERT) GO TO 411
ELSE ERASE LINES BELOW OUTPUT WINDOW

DO 410 LINE = OY21 + 1, 511
CALL IVJ(BNDL, LINE, ZEROS)
CONTINUE

END IF
END IF
END IF
CONTINUE
CONTINUE
CONTINUE
RETURN

END
SUBROUTINE CMDINDO

SCALES AND LOADS A WINDOW VIA CONTEXT DEPENDANT INTERPOLATION

IMPLICIT INTEGER (A-0, S-Z)
REAL FLOAT
DIMENSION DX(512,2),DY(512,2)
BYTE IB(512),OB(512),ZEROS(512)
BYTE BYTE
LOGICAL INSERT
COMMON /WINDOW/ IX1,IX2,IY1,IY2,OX1,OX2,OY2,PMAG,INSERT
COMMON /CHANL/ INCHS(5),OCHKS(5)
EQUIVALENCE (BYTE,WORD)

SET UP MAPPING FROM INPUT TO OUTPUT COORDINATES

PXINC=FLOAT(IX2-IX1+1)/FLOAT(OX2-OX1+1)
PYINC=FLOAT(IY2-IY1+1)/FLOAT(OY2-OY1+1)

DO 20 J=OY1+1,0Y2
P=POSIT-PYINC*K
INTG=INT(P)
FFRACT=P-POSIT*FLOAT(INTG)
FRCT=1.0-FFRACT
DY(J,1)=INTG
DY(J,2)=FFRACT
K=K+1
20 CONTINUE

DO 30 J=OY1+1,0Y2
P=POSIT-PXINC*K
INTG=INT(P)
FFRACT=P-POSIT*FLOAT(INTG)
FRCT=1.0-FFRACT
DX(J,1)=INTG
30 CONTINUE

Compute mapping tables
READ INPUT DATA

REAL X0, Y0, X1, Y1, S, T, U, V

CINTERPOLATE X AND Y VALUES

U = X0 + (X1 - X0) * T
V = Y0 + (Y1 - Y0) * T

CWRITE OUTPUT TO DEVICE

WRITE OUTPUT DATA

END
GET 4 INPUT PIXELS SURROUNDING INTERPOLATING POINT
PLEFT=DX(PIX.1)
PRIGHT=PLEFT+1
BYTE=IB(PLEFT\LA\BOVE)
A=WORD
BYTE=IB(PRIGHT\LA\BOVE)
B=WORD
BYTE=IB(PLEFT\LB\LO\W)
C=WORD
BYTE=IB(PRIGHT\LB\LO\W)
D=WORD
GET FRACTIONAL PART OF INTERPOLATING POINT LOCATION
X=DX(PIX.2)
Y=DY(LINE.2)
SELECT AND INTERPOLATE ON A PLANE CHOSEN BY LOCAL GRADIENT AND QUADRANT OF POINT IN ABCD
CASE=1
IF(IBS(C-B),LT,IBS(A-D))
CASE=3
IF((CASE.EQ.1).AND.,(Y,LT,X))
CASE=2
IF((CASE.EQ.3).AND.,((128-Y),LT,X))
CASE=4
GO TO (331,332,333,334) CASE
OB(PIX)=A+((XY(D-C)+XY(C-A))/128)
GO TO 340
OB(PIX)=A+((XY(B-A)+XY(D-B))/128)
GO TO 340
OB(PIX)=A+((XY(B-A)+XY(C-A))/128)
GO TO 340
OB(PIX)=C+(XY(D-C)+(128-Y)*X(B-D))/128
END CASES
WRITE OUTPUT LINE TO IAC
CALL IWH(END,LINE,OB)
LASTDY = NEWDY
CONTINUE
IF(INSERT) GO TO 411
ELSE ERASE LINES BELOW OUTPUT WINDOW
DJ 410 LINE = DYL+1,511
CALL IWH(END,LINE,ZEROS)
CONTINUE
END IF
CONTINUE
CONTINUE
CONTINUE
CONTINUE
CONTINUE
APPENDIX B

THEME FILTERING PROGRAM
SUPERVISED THEME FILTER

THIS PROGRAM PERFORMS OPERATOR CONTROLLED LOW PASS SPATIAL FILTERING ON A CATEGORIZED THEMATIC IMAGE. IT TABULATES INDIVIDUAL THEME OCCURRENCES OVER A RECTANGULAR FILTER AREA, AND IN ACCORDANCE WITH OPERATOR SPECIFIED CONSTRAINTS, REPLACES THE CENTER PIXEL OF THE RECTANGULAR AREA WITH THE LOCALLY MOST COMMON THEME.

THE PROGRAM IS PRIMARILY DESIGNED TO TRANSFORM THEMATIC IMAGES PRODUCED BY SUPERVISED OR UNSUPERVISED COMPUTER CLASSIFICATION TECHNIQUES INTO SPATIALLY SIMPLIFIED THEMATIC MAPS, WHICH CLOSELY RESEMBLE THOSE WHICH MIGHT BE INTERPRETED BY MANUAL PHOTO INTERPRETATION. IT PERFORMS THE SPATIAL SIMPLIFICATION NECESSARY TO CHANGE "NOISY" COMPUTER THEMATIC MAPS INTO ONES WHICH CAN BE PRINTED OR STORED AS OUTLINES OR POLYGONS.

BY JUDICIOUS USE OF THESE REPLACEMENT CONSTRAINTS, IT CAN ALSO BE USED TO RENDER CLASSIFICATION SHOWN ON A SPATIAL BASIS: TO FILTER ONLY SPECIFIC THEMES, OR TO TOTALLY REMOVE UNDESIRE Features SUCH AS SHADOWS.

INPUT:  A CLASSIFIED IMAGE WHERE EACH THEME HAS BEEN MAPPED TO A SPECIFIC GREY LEVEL.

          USER SPECIFIED FILTER SIZE.

          USER SPECIFIED CONSTRAINTS FOR EACH THEME

OUTPUT: SMOOTHED IMAGE MAPPED TO THE SAME GREY LEVELS

FORTRAN PROGRAM MODULES:

THMFLIL ---MAIN PROGRAM

THMHLD ---OPERATOR INTERFACE WHICH PROMPTS FOR AND ACCEPTS A SOMEWHAT LENGTHY LIST OF AREA THRESHOLDS WHICH CONTROL FILTERING ACTION

PDP-11 ASSEMBLER MODULES:

THMTOT ---COMPUTES A SET OF RUNNING SUMS, ONE FOR EACH IMAGE COLUMN

ROWTOT ---COMPUTES A LINE OF AREAL SUMS BY TAKING A RUNNING SUM ON THE COLUMN SUMS;

REPLAC ---CHECKS THEME AREA TOTALS FOR EACH FILTER POSITION, CHECKS CONSTRAINTS AND REPLACES CENTRAL PIXEL

IMAGE 100 LIBRARY SUBROUTINES CALLED:

FROM
INTF --- FREE FORMAT INPUT ROUTINES
IPX --- READS BOX CURSOR POSITION
IPV --- READ AND WRITE LINES TO VIDEO MEMORY
OUTPUT --- TERMINAL CONTROL ROUTINE
TALOOK --- TABLE LOOKUP FOR VIDEO VALUES

KEY VARIABLES:
BUF --- ALPHANUMERIC INPUT STRING
THVAL --- ARRAY CONTAINING GREY LEVEL OF EACH THEME
GROPCT --- ARRAY OF HELP PERCENTILES, ONE FOR EACH THEME, FOR A
GIVEN THEME TO GROW OR REPLACE ANOTHER, THE THEME MUST
COVER AT LEAST THIS PERCENT OF THE FILTER RECTANGLE
SHPCT --- AN ARRAY OF AREA PERCENTILES WHICH CAN NOT BE EXCEEDED
IF A GIVEN THEME IS TO SHRINK OR BE REPLACED
ENTAB --- LOOKUP TABLE TO MAP THEMES TO LOWEST GREY LEVELS
DETAB --- LOOKUP TABLE TO MAP PROCESSED IMAGE BACK TO THEME LEVELS
COLSUM(PIXELS, THEMES) --- ARRAY TO STORE RUNNING COLUMN SUMS
THMSUM(PIXELS, THEMES) --- ARRAY TO STORE RUNNING AREA SUMS
NEWB
MIDB --- BUFFERS FOR BOTTOM, MIDDLE AND TOP LINES OF FILTER POSITION
OX1,OY1
OX2,OY2 --- UPPER LEFT AND LOWER RIGHT CORNERS OF IMAGE
SEGMENT TO BE PROCESSED

SUBROUTINE THMFL

PERFORMS SPATIAL THEME FILTERING WITH A RECTANGULAR WINDOW

IMPLICIT INTEGER (A-0..5..2)
INTEGER R
DIMENSION BUF(40)
DIMENSION THVAL(9)

DIMENSION BUFFER(256,16)

DIMENSION CS(668), COLSUM(512,9)

EQUIVALENCE (BUFFER(1,1), CS(1), COLSUM(1,1))

BYTE TS(4608), THSUM(512,9)

EQUIVALENCE (TS(1), THSUM(1,1))

BYTE ENTAB(256), DICTY(256), PPETAB(256), ZERO(256)

BYTE LINES(156)

BYTE LIDB(512), HIDE(512), OLDB(512)

EQUIVALENCE (HDB(1), LINES(1))

EQUIVALENCE (LIDB(1), LINES(1))

EQUIVALENCE (OLDB(1), LINES(1:25))

COMMON/EXCL/SU(S), BONUS-EXUL/BONUS, RETTSK, NEXTSK, RAUX, RTSK

EQUATION X(10), SY(10), DIVIDE(5), TIME(4), RDGPT(17)

COMMON/TRID/PROT,LINES(4), PAPER, PARCH(4), MAHR(4)

COMMON/TRND/FIELD,ROPS(15), LRECL, NRECL, IERR1, NDIM1, IERR2, NDIM2

COMMON/TRNS/FPNX(15), IPNL, FNPX, FNB(15), THRESH, FL

COMMON/SCENE/SID(15), C(14), EPS(2,2), IMEC(2,2), CF, RFCFLAT, RFCFLDR

COMMON/PARAM/HAUNAL, LRES(2), HSIG(2,4), HSF(2,4)

COMMON/PSNSP/10

END 1-100 GLOBAL COMMON

DIMENSION PFCNT(9), SHPECT(9), GROCENT(9), SHCENT(9)

DATA GROFCT /100, 000, 000, 000, 000, 000, 000, 000, 000, 000/

DATA SHPFCT /100, 100, 100, 100, 100, 100, 100, 100, 100, 100/

IMAGE LIMITS

DATA MMIOIN'/4, MAXLINES/509, MINPIX/6, MAXPIX/511/

PROCESSING BUFFER SIZE

DATA BUFLEN/512/

THEME INFORMATION

DATA THEMFIN/0,1,2,4,8,16,32,64,128/

DATA THEMFIN/

DATA X''/''", R''/''", C''/''", NUL''/

DATA UTY/6/

DATA NCOL'/3, NROVL/2/

LIST INSTRUCTIONS

CALL OUTPUT(27,12,7)
issue preliminary instructions to operator

designate temporary storage space

generate look up tables to encode and decode theme levels to processing indexes
0005 73 CONTINUE
0020 74 CONTINUE
0038 75 CONTINUE
0059 76 CONTINUE
0068 77 CONTINUE

C
C WRITE(UTY,1000) NROW,NCOL
0082 1000 FORMAT('FILTER SIZE = ',I3,' X',I3,' ') C
C CALL OUTPUT()
0084 1004 READ (UTY,1001) BUF
C
0092 1006 FORMAT(4X,N2)
0094 1008 IF(BUF(1),EQ,0) GO TO 125
0096 1009 IF(BUF(1),EQ,0) GO TO 500
0098 1011 IP=0
0100 1013 CALL INTFP(IP,BUF,GO,NCOL)
0102 1015 CALL INTFP(IP,BUF,GO,NCOL)
0104 1017 IF(NCOL.GT.101).OR.(NCOL.LT.1) GO TO 100
0106 1019 IF(NROW.GT.101).OR.(NROW.LT.1) GO TO 100
0108 125 CONTINUE
0109 C
C AREA=NCOL*NROW
0111 WRITE(UTY,1400)
0122 1400 FORMAT('PROCESS (C)URSORED AREA OR (S)UBSCENE? S')
0124 C
0126 1402 CALL OUTPUT(?)
0128 1404 READ (UTY,1001) BUF
0130 1406 IF(BUF(1),EQ,0) GO TO 900
0132 1408 IF(BUF(1),EQ,0) GO TO 145
C
C ELSE GET LIMITS FROM 1-100 COMMON
0140 Dx1=IMAGC(1,1)+1
0150 Dx1=MAX0(Dx1,MINPIX)
0160 Dy1=IMAGC(2,1)+1
0170 Dy1=MAX0(Dy1,MINLIN)
0180 Dx2=IMAGC(1,2)+1
0190 Dx2=MIN0(Dx2,MAXPIX)
0200 Dy2=IMAGC(2,2)+1
0210 Dy2=MIN0(Dy2,MAXLIN)
0220 1417 GO TO 149
C
C THEN GET LIMITS FROM CURSOR POSITION
0230 145 CONTINUE
0240 CALL INTFP(CURSOR)
0250 01+CURSOR+1+1=IP(3)
0260 02+CURSOR+1+1=IP(5)
0270 01+CURSOR+1+1=IP(3)
0280 02+CURSOR+1+1=IP(5)
C
0290 149 CONTINUE

C operator selection of processing parameters

C GET THRESHOLD TO CONTROL FILTERING ACTION
0300 150 CALL THRESH(NCOL,NROW,NTHMN,NTHMC,SHRPT,SHRNT)
C
C DETERMINE PROCESSING WINDOW
0310 140 CALL UTY,1400)
0400 1404 FORMAT('PROCESS (C)URSORED AREA OR (S)UBSCENE? S')
0500 1406 IF(BUF(1),EQ,0) GO TO 900
0600 1408 IF(BUF(1),EQ,0) GO TO 145
C
C ELSE GET LIMITS FROM 1-100 COMMON
0700 Dx1=IMAGC(1,1)+1
0800 Dx1=MAX0(Dx1,MINPIX)
0900 Dy1=IMAGC(2,1)+1
1000 Dy1=MAX0(Dy1,MINLIN)
1100 Dx2=IMAGC(1,2)+1
1200 Dx2=MIN0(Dx2,MAXPIX)
1300 Dy2=IMAGC(2,2)+1
1400 Dy2=MIN0(Dy2,MAXLIN)
1500 1417 GO TO 149
C
C THEN GET LIMITS FROM CURSOR POSITION
1600 145 CONTINUE
1700 CALL INTFP(CURSOR)
1800 01+CURSOR+1+1=IP(3)
1900 02+CURSOR+1+1=IP(5)
2000 01+CURSOR+1+1=IP(3)
2100 02+CURSOR+1+1=IP(5)
C
2200 149 CONTINUE

C selection of processing window
initialize summing buffers

DO 160 I=1,1,INLEN
160 CONTINUE

DO 1st 1=1,BUFLEN
131 Lines1=0
132 CONTINUE

SET UP NEGATIVE START LOCATIONS FOR BOX

BOTLP=0;1-(X+1)/2
TOPLP=BOTLP+INLEN
LINSTP=TOPLP-H10V/2

BOTLAM=BOTLP
TOPLAM=TOPLP

DO 399 LINE=LINSTR3

READ TOP, Bot, AND MILD LINES OF BOX (WITH REFLECTION)

BLREAD=BTFLN
IF(BOTAML.LT.OY1) BLREAD=BOTAML+OY1+G1Y=1
IF(BOTAML.GT.OY2) BLREAD=BOTAML-OY2+OY2
TLREAD=TOPLAM
IF(TOPAML.LT.OY1) TLREAD=TOPAML+OY1+OY1=1

CALL IFB(RNBH,TLREAD,OLDB)
CALL IFB(RNBH,LINES,HDIB)
CALL IFB(RNBH,BLREAD,NEB)

IF(TOPAML.EQ.BTR) GO TO 305
IF(BOTAML.GE.BOTLAM) GO TO 305
END IF

305 CONTINUE

BOTAML-BOTAML+1
TOPAML-TOPAML+1

U CODE INPUT VALUES TO o-MAXTHM

CALL TALOOK(LINES,ENTSBLINES,1536)

UPDATE STORED COLUMN SUMS

IF(TOPAML.LT.BOTSTR) GO TO 318
GOTO 328

THEN PRELOAD SUMMING BUFFERS

DO 315 PIX=01,0X2
       B=NEWB(PIX)
C

0163
0164
0165

315
CONTINUE
GO TO 325

ELSE ADD THEMES FROM NEW LINE AND SUBTRACT THEMES FROM OLD VIA ASSEMBLER ROUTINE

CONTINUE

320

FORTAN REPLACED

DO 322 PIX=0,X,0.2

E+NEB(PIX)

INDEX=PIX+BUFSL

CS(INDEX)=CS(INDEX)+E

T=OLDB(PIX)

INDEX=PIX+BUFSL

CS(INDEX)=CS(INDEX)-T

CONTINUE

322

CALL THMTOT(NHB,OLDB,CS,PIX1,NUMPIX,BUFLEN)

update column sums

END IF

325
CONTINUE

COMPUTE AREAL SUMS

IF(LINE.LT.0Y) GO TO 395

170

ONE=1

DO 330 INDEX=1,NUMTH+1

173

CALL ROWTOT(COLSUM(PIX1,INDEX),

THMTOT(PIX1,INDEX),NUMPIX,NECL,ONE)

update area sums

330
CONTINUE

FOLLOWING CODE EXECUTED BY ASSEMBLER ROUTINE REPLACED

REPLACE OUTPUT WITH MOST HEAVILY WEIGHTED THEME

DO 350 PIX=0,X,0.2

KEYIND=MIDB(PIX)+1

IF(THMTOT(PIX,KEYIND),GT,SHRTOT(KEYIND))

GO TO 344

ELSE PIXEL CAN BE CHANGED

MAXTOT=0

DO 36 INDEX=1,NUMTH+1

THT=THMTOT(PIX,INDEX)

IF(THT.LE.MAXTOT) GO TO 344

IF(THMTOT(INDEX,LT.0) GO TO 344

IF(THT.LT.GRHT(INDEX))GO TO 344

ELSE REPLACEMENT POSSIBLE

MAXTOT=THT

MIDB(PIX)*INDEX=1

execute replacement logic

344
CONTINUE

346
CONTINUE

END IF
In the context of a FORTRAN IV program:

```fortran
4.1
4.0
S.
M-4j
1. Ina
0~ +

write output time
```

```fortran
C
C C348
C C350
C CC
C CCCCCCCC
C END OF REPLACEMENT CCCCCCCC
C
C 0176
C NTHEME={NUMTHEME+1}
C DATA=NUMTHEME
C CALL REPLAC(MIDB,TS,01,X,NTHEME,
C GROUND,SMRTNT,BUFFNR)
C
C DECODE AND WRITE OUTPUT LINE
C
C 0178
C CALL TALOOK(MIDB,DETAB,MIDB,512)
C CALL IIM(BNDD,LINE,MIDB)
C
C 0180 395
C END IF
C 0181 399
C CONTINUE
C
C 0182 400
C WRITE(LUTY,4000)
C FORMAT(/'$H(CHANGE THRESHOLDS, (R)ECYCLE, OR E(X)IT)'/)
C 0184
C CALL OUTPUT(7)
C 0185
C READ(LUTY,1001)BUF
C 0186
C IF(BUF(1),ED.C) GO TO 100
C 0188
C IF(BUF(1),ED.F) GO TO 70
C 0190
C IF(BUF(1),ED..) GO TO 900
C 0192
C GO TO 400
C
C 0193 900 CONTINUE
C 0194
C RETURN
C
C 0195 END
```
SUBROUTINE THRDLD(NCOL, NFOW, NUMTHM, GROPC1, GROCNT, SHRPT, SHRCON)

OPERATOR INTERFACE FOR THEME FILTERING

IMPLICIT INTEGER (A-Z)

REAL BARBAR

DIMENSION BUF(40), DAT(6), TIM(6)
DIMENSION SHRPT(9), GROPC1(9)
DIMENSION SHRCON(9), GROCNT(9)
DIMENSION ACTION(6, 9)

DATA ACTION

DATA CHATHM/B/
DATA Y /'Y'/
DATA N /'N'/
DATA NULL /' '/
DATA UTY /'Y'/

C C SET THEMES NOT TO BE PROCESSED FOR NO CHANGE

IF(NUMTHM.GE.MAXTHM) GO TO 90
ELSE SET NO CHANGE
DO 90 TH=NUMTHM,MAXTHM-1
90 GROPC1(TH+2)=0
CONTINUE
90 CONTINUE

DISPLAY AND CHANGE THEME THRESHOLDS

C C

CONTINUE
100 CONTINUE

IF(NUMTHM.GE.MAXTHM) GO TO 101
CALL ORTHM(NCOL, DAT, TIM)
CALL UTHM(NCOL, DAT, TIM)
CALL TIME-TIM
WRITE (U, 1010) NCOL, DAT, TIM
1000 PRINT(*, 'THEME THRESHOLD PERCENTILES')

END IF

WRITE (U, 1010)
1010 WRITE (U, 1010)
change percentiles to numeric count

:compute approximate filtering action for operator verification

sense for any keyboard inputs and override defaults
C      END IF
0092  170    CONTINUE
C      END IF
0093  180    CONTINUE
0094  190    CONTINUE
0095    IF (CHANGE.NE.0) GO TO 190
0096    CHANGE = 0
0097    GO TO 190
0098  193    WRITE (UTY, 1930)
0099  1930   FORMAT (//)
0100  195    WRITE (UTY, 1950)
0101  1950   FORMAT ('CHANGE THRESHOLD VALUES? (Y)ES, OR (N)O     ')
0102  1960   FORMAT (A2)
0103  1960   FORMAT (A)
0104    IF (CHANGE.NE.Y) AND (CHANGE.NE.N)) GO TO 195
0107    CHANGE = 1
0109    ELSE
0109    CHANGE = 1
0110    GO TO 102
0111  198    CONTINUE
0112    RETURN
0113    END
THREE ROUTINES FOR RAPID EXECUTION OF THEMES FILTERING

.TITLE THSUN
.GLOBAL THSUN
.GLOBAL THMTOT
.GLOBAL THIMPL

FIRST ROUTINE DOES RUNNING SUM OF THEMES INTO A SET OF ACCUMULATING BUFFERS, ONE PER THEME

FORTRAN CALLING SEQUENCE:
CALL THSUN (HEB, OLDB, SUMB, START, NUMPIX, BUFLEN)

HEB = BYTE ARRAY CONTAINING NEXT LINE FOR AVERAGING AREA
OLDB = BYTE ARRAY CONTAINING LINE TO BE SUBTRACTED FROM AREA
SUMB = WORD ARRAY SUMB (BUFLEN, THM) CONTAINING COLUMN SUMS OF THEMES
BUFLEN = NUMBER OF PIXELS IN THE LINES

THMTOT:
UPDATE SUMB BY ADDING NEW LINE AND SUBTRACTING OLD

MOV 2(R5), R0  | NEW LINE ADDRESS
MOV 4(R5), R1  | OLD LINE ADDRESS
MOV 6(R5), R2  | SUM BUFFER ADDRESS
MOV #14(R5), R3  | GET START PIXEL
MOV #14(R5), R4  | GET BUFFER LENGTH
ASL R4  | IN WORDS
MOV #12(R5), R5  | SET LOOP COUNTER

ADD R3, R0  | POINTS TO 1ST NEWB PIXEL
ADD R3, R1  | POINTS TO 1ST OLDB PIXEL
ADD R3, R2  | POINTS TO 1ST SUMB PIXEL

LOOP: MOV B (R1)+, R3  | GET THEME NUMBER FROM OLD LINE
       MOV R4, R3  | R3 = BUFLEN-THM
       ADD R2, R3  | TARGET ADDRESS
       DEC R3      | DECREMENT THE SUM
       MOV B (R0)+, R3  | THEME NUMBER FROM NEW LINE
       MOV R4, R3  | R3 = BUFLEN-THM
       ADD R2, R3  | TARGET ADDRESS
       INC R3      | INCREMENT THE SUM
       TST 0, R2+   | INCREMENT TO SUMB (NEXTPIX, THM)
       SUB R2, LOOP
       PXI 15, R5  |
2ND REPLACES THE KEY PIXEL WITH MOST COMMON LOCAL THEME
WHICH PASSES THE GROWING AND SHRINKING CONSTRAINTS

FORTRAN CALLING SEQUENCE:
CALL REPLAC(MIDB,SUMD,START,NPIX,NTHM,GRO,SHR,BUFLEN)

MIDB = BYTE ARRAY CONTAINING CENTER LINE OF BOX
SUMD = BYTE ARRAY SUMD(BUFLEN) CONTAINS THEME COLUMN SUMS
NPIX = NUMBER OF PIXELS TO BE OUTPUT
NTHM = NUMBER OF THEMES TO BE PROCESSED
GRO = ARRAY GRO(NTHM) CONTAINS POPULATION THRESHOLDS WHICH
MUST BE EXCEEDED BEFORE A THEME IS ALLOWED TO GROW
SHR = ARRAY SHR(NTHM) CONTAINS POPULATION THRESHOLDS WHICH
MUST NOT BE EXCEEDED IF A THEME IS ALLOWED TO SHRINK
BUFLEN = LENGTH OF BUFFERS IN PIXELS

REPLACE:

80 000070 010567 000244
81 000070 015567 000004 000240
82 000074 015567 000004 000240
83 001012 017567 000006 000224
84 001110 017567 000010 000230
85 001116 017567 000012 000230
86 001214 016667 000014 000220
87 001312 016667 000016 000214
88 001410 017567 000020 000210
89 001416 016565 000002
90 001416 016565 000002
91 000152 066767 000166
92 000156 065905
93 000160 162767 000202 000154
94 000166 066767 000152 000146
95 000174 010567 000160
96 000200 066767 000002 000152
97 000200 066767 000002 000152
98 000200 066767 000002 000152
99
100
101 000206 005205
102 000210 111581
103
104 000212 016700 000136
105 000216 060100
106 000220 00100
107
108
109 000222 062767 000001 000112
110 000230 070167 000122
111 000234 066767 000102
112
113 000240 121110
114
THMSM MACRO D1013 19-NOV-81 21:11 PAGE 1-9

115 000242 002030
116
117
118 000244 005000
119 000246 005001
120 000250 016702 000076
121 000254 016703 000062
122
123 000260 012206
124 000262 111304
125 000264 042704 177400
126
127 000270 066703 000062
128
129 000274 020004
130 000276 002006
131
132 000300 020604
133 000302 003004
134
135 000304 005705
136 000306 002406
137
138 000310 019400
139 000312 110115
140
141 000314 005201
142 000316 020157 000026
143 000322 003756
144
145 000334 020567 000030
146 000336 002736
147
148 000340 016704 000002
149
150 000404 003000
151

BGE PXTEST

THLOOP:

MOU (R2)+,SP

MOVB (R3),R4

BIC #177402,R4

THTEST:

MOU (R3),R4

ADJ BULLEN,R3

GET THM GROW THRESHOLD

GET THM GROW SUM

SET SUM POINTER TO SUMB(PX;NEXTHM)

JUMP OUT IF THM NOT A MAXIMUM

JUMP OUT IF TOTAL BELOW GROW THRESHOLD

JUMP OUT IF THM IS TO BE REPLACED

MOU R4,R0

MOVB R1,(R5)

CHANGE MAXIMUM TO NEW MAX

HAND OUTPUT THE THM NUMBER

THTEST:

INC R1

CMP R1,NTHM

BLT THLOOP

PXTEST:

CMP PS,STOP

BLT EXLOOP

RESTORE STACK POINTER
DATA NEEDED TO AVOID LINE SHORTENING
DOUBLE PRECISION ARITHMETIC TO AVOID OVERFLOW

.TITLE RONOT
.GLOBAL RONOT

FORTRAN CALL: CALL RONOT(IBUF,OBUF,LENGTH,AVELEN,DIVISR)

IBUF = WORD ARRAY CONTAINING INPUT DATA
OBUF = BYTE OUTPUT ARRAY
LENGTH = NUMBER OF PIXELS TO BE OUTPUT
AVELEN = NUMBER OF ELEMENTS TO BE AVERAGED
DIVISR = DIVISOR TO COMPUTE AVERAGE

SET LOOP COUNTERS

MOV LENGTH,STOP2
ADD LENGTH,STOP2
MOV IBUF,STOP2
MOV LENGTH,STOP3
ADD OBUF,STOP3
MOV SB,SAVE

INITIALIZE POINTERS

MOV AVELEN,RO
INC RO
MOV RO,R1
THIS = 2K((AVELEN+1)/2)

MOV IBUF,FORSTR
SUB RO,FORSTR
STARTING ADDRESS FOR LEADING ELEMENT

MOV FORSTP,BACSTR
SUB AVELEN,BACSTR
STARTING ADDRESS FOR TRAILING ELEMENT

MOV IBUF,OUTSTR
SUB RO,OUTSTR
STARTING ADDRESS FOR OUTPUT BYTE

MOV IBUF,OFFSET
ADD IBUF,OFFSET
TO ACHIEVE LOW END REFLECTION, SUBTRACT ADDRESS FROM OFFSET

MOV RO,OFFSET
SET THE POINTERS

MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBUF,RO
MOV IBU
PROCESSING LOOP FOR THE FIRST SEGMENT OF THE LINE

LOOP1:

MOV R4, R0  ; LEADING ELEMENT ADDRESS
ADD #2, R4

CMP R0, IBUF  ; DOES IT POINT TO VALID DATA YET?
BGT FOROK

NEG R0  ; IF NOT, REFLECT ADDRESS
ADD OFFSET, R0

FOROK:

ADD (R0) + R2  ; ADD LEADING ELEMENT INTO ACCUMULATORS
ADC R3  ; WITH DOUBLE PRECISION

MOV SP, R0  ; TRAILING ELEMENT ADDRESS
ADD #2, SP

CMP R0, FORSTR  ; ARE WE STILL PRELOADING ACCUMULATORS?
BLT TEST1

CMP R0, IBUF  ; INTO VALID DATA YET?
BGT BADCOK

NEG R0  ; IF NOT, REFLECT ADDRESS
ADD OFFSET, R0

BADCOK:

SUB (R0) + R2  ; SUBTRACT TRAILING ELEMENT FROM ACCUMULATORS
SBC R3  ; WITH DOUBLE PRECISION

CMP R5, OBUF  ; IS OUTPUT POINTER INTO OBUF YET?
BLT TEST1

MOV R2, P1  ; IF YES, COMPUTE AVERAGE
MOV R3, R0
DIV DIVUSR, R0

ADDB R0, R5  ; (AND PUT IT INTO OBUF

TEST1:

INC R5  ; INCREMENT OUTPUT POINTER
CMPL SP, IBUF  ; IS ALL INPUT PAST IBUF YET?
BLE LOOP1

PROCESSING LOOP FOR THE MAIN SEGMENT OF THE LINE

LOOP2:

SUB (SP) + R2  ; SUBTRACT TRAILING ELEMENT NORMALLY
SBC R3

ADD (R4) + R2  ; THEN ADD LEADING ELEMENT NORMALLY
ADC R3

MOV R0, P1  ; COMPUTE THE AVERAGE
DIV DIVUSR, R0

MOVB R0, R5  ; OUTPUT THE AVERAGE

CMP SP, IBUF
BGT LOOP2
PROCESSING LOOP FOR THE LAST SEGMENT OF THE LINE

LOOP3:  SUB  (SP)+, R2  ; SUBTRACT TRAILING ELEMENT NORMALLY
       DCC  R3
       ADD  -(R4), R2  ; THEN ADD LEADING ELEMENT BY REFLECTION
       DCC  R3
       MOV  R2, R1  ; COMPUTE THE AVERAGE
       MOV  R3, R0
       DIV  DIVISOR, R0
       MOVB  R0, (RS)+  ; OUTPUT THE AVERAGE
       CMP  STOP3, R5
       BGT  LOOP3
       MOV  SAVE, SP  ; RESTORE STACK POINTER
       RTS  FC

DIVISOR: +0
STOP1: +0
STOP2: +0
STOP3: +0
AVLEN: +0
LENGTH: +0
FORSTR: +0
HALFSTR: +0
OUTSTR: +0
IBUF: +0
DBUF: +0
OFFSET: +0
.END