DIGITAL IMAGE-PROCESSING AND ANALYSIS TECHNIQUES
FOR SeaMARC II SIDE-SCAN SONAR IMAGERY

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF
THE UNIVERSITY OF HAWAII IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN GEOLOGY AND GEOPHYSICS
MAY 1987

BY
Thomas Beckett Reed, IV

Dissertation Committee
Donald M. Hussong, Chairman
William T. Coulbourn
Gerard J. Fryer
Alexander Malahoff
Edward J. Weldon

Return to
School of Ocean and
Earth Science and Technology
LIBRARY
We certify that we have read this dissertation and that in our opinion it is satisfactory in scope and quality as a dissertation for the degree of Doctor of Philosophy in Geology and Geophysics.

DISSERTATION COMMITTEE

[Signatures]

[Signatures]

[Signature]
ACKNOWLEDGMENTS

One comment which all members of my committee agreed upon after reading my dissertation, was my overt usage of the pronoun "we". This was no attempt at royalty. Rather, it was an open admission of the plurality of effort which accompanied this research. To my entire committee I am indebted for their extraction of significance from my sometimes random concatenations of psychology and geophysics. Particularly I am indebted to my chairman and advisor, Don Hussong, who not only had the temerity to hire "another Ivy-league wing-nut", but who also dared to curb my somewhat eufluistic literary style. I also am indebted for a major part of my training and assistance with algorithms to the entire SeaMARC staff, especially Grant, Karen, and Diane, who tolerated with such good patience endless neophyte questions and attempts at the necromancy of Harris JCL and programing.

To P. B. Humphrey, C. deMoustier, J. Sender, and J. C. Farre I am indebted for excellent critical reviews and assistive comments upon the manuscript in its early stages. To S. P. Dang I am much indebted for his fine photographic work and assistance in drafting. To S. S. Davis go my sincere thanks for proof-reading, collation, and general support in the production of this dissertation. This research was supported by the Office of Naval Research under contract N00014-82-C-0380.

"Far better is it to dare mighty things, to win glorious triumphs, even when checkered by failure, than to take rank with those poor spirits who neither enjoy nor suffer much, because they live in the gray twilight that knows not victory nor defeat." TBR
The recent growth in the production rate of digital side-scan sonar images, coupled with the rapid expansion of systematic seafloor exploration programs, has created a need for fast and quantitative means of processing seafloor imagery. Computer-aided analytical techniques fill this need. A number of numerical techniques used to enhance and classify imagery produced by SeaMARC II, a long-range combination side-scan sonar and bathymetric seafloor mapping system are documented. Four categories of techniques are presented: 1) preprocessing corrections and enhancements, 2) filtering, 3) feature extraction, and 4) image classification.

Preprocessing transformations produce a data set which is a geometrically and radiometrically accurate representation of the seafloor, corrected for undesirable characteristics of the data acquisition system and the medium in which it operates. Radiometric corrections include removal of gain changes, multiples, beam pattern variations, and scattering variations. Geometric corrections include total sensor attitude and terrain topographic rectifications, and removal of along-track aliasing. Filtering and enhancement operations include contrast stretching, high and low pass filtering, noise cleaning, and edge enhancement and delineation. A novel, faster algorithm for median value filtering is presented.
An introduction to the concept of "feature vectors" is provided, along with a review of possible methods of extracting informative features from images. One such method, based upon Gray-Level Co-occurrence Matrices (GLCM), is developed in depth. Synthetic images are generated to test the theoretical sensitivity of the GLCM method to wavelength, rotation invariance, noise level, and mixed-wavelength signals. An alternative to the a priori texel (texture element) subdivision of images is presented in the form of ReGATA - Region Growing And Texture Analysis. This routine simultaneously provides a texture map of spatial resolution superior to that obtainable with arbitrarily assigned texel boundaries, and minimizes the possibility of mixed texture signal due to combining two or more textures in an arbitrarily assigned texel.

Computer classification of these textural features extracted via the GLCM technique results in transformation of images into maps of seafloor texture. These maps may either be interpreted in terms of the theoretical relationships shown between texture signatures and wavelength, or converted to geologic maps by correlation of texture signatures with ground truth data.

These techniques are applied to SeaMARC II side-scan sonar imagery from a variety of geologic environments, including lithified and non-lithified sedimentary formations, volcanic and sedimentary debris flows, and crystalline basaltic outcrops. Application of the above processing steps provided not only superior images for both subjective and quantitative analysis, but also the critical ability to discriminate between outcrops with radically distinct lithologies but similar image intensity.
Lastly, documented FORTRAN code and usage instructions for the processing routines described in this paper are provided in the appendices.
TABLE OF CONTENTS

ACKNOWLEDGEMENTS ........................................................................................................... iii

ABSTRACT ............................................................................................................................... iv

LIST OF TABLES ....................................................................................................................... ix

LIST OF ILLUSTRATIONS ....................................................................................................... x

1 INTRODUCTION ..................................................................................................................... 1

1.1 General Introduction and Statement of Purpose ................................................................. 1
1.2 Previous Work ..................................................................................................................... 4
1.3 Characteristics of Side-Scan Sonars .................................................................................. 7
1.4 SeaMARC II System .......................................................................................................... 10
1.5 Requirements for Image Processing .................................................................................. 17

2 PREPROCESSING CORRECTIONS ....................................................................................... 21

2.1 Objectives .......................................................................................................................... 21
2.2 Preprocessing Corrections: Purposes and Definitions ....................................................... 22
2.3 Radiance Transformations ............................................................................................... 23
2.4 Contrast Stretching ........................................................................................................... 25
2.5 Background Subtraction .................................................................................................... 36
2.6 Geometric Rectifications — Layover ............................................................................... 42
2.7 Geometric Rectifications — ”Point Migration” ............................................................... 50

3 IMAGE FILTERING .............................................................................................................. 58

3.1 Introduction to Filtering .................................................................................................... 58
3.2 The Box Filter ................................................................................................................... 58
3.3 Median-Value Filter ......................................................................................................... 59
3.4 Edge Detection ................................................................................................................ 61

4 FEATURE EXTRACTION ...................................................................................................... 69

4.1 The Feature Vector ............................................................................................................ 69
4.2 Texture .............................................................................................................................. 72
4.3 The Auto-Covariance Function ....................................................................................... 73
4.4 The Power Spectral Method ............................................................................................ 74
4.5 Laws' ”texture energy transform” ................................................................................... 75
4.6 The gray-level methods ................................................................................................... 75
4.7 The Gray-Level Co-Occurrence Method ....................................................................... 77
4.8 GLCM Features ............................................................................................................... 82
4.9 Evaluation of image texture with the GLCM ................................................................. 84

5 CLASSIFICATION ................................................................................................................ 87

5.1 Introduction ....................................................................................................................... 87
5.2 Un-supervised Classification ............................................................................................ 87
5.3 Supervised Classification.................................................................................................. 89
# TABLE OF CONTENTS (continued)

6 THEORETICAL INVESTIGATION OF TEXTURE ........................................ 95
   6.1 Method ......................................................... 95
   6.2 Wavelength Sensitivity ........................................ 95
   6.3 Stationarity .................................................. 106
   6.4 Noise Sensitivity ............................................. 109
   6.5 Rotation Invariance .......................................... 112
   6.6 Wavelength mixing ........................................... 115

7 APPLICATIONS .......................................................... 121
   7.1 General ......................................................... 121
   7.2 Discrimination of Basalt and Dolomite ........................ 130
   7.3 Comparison of SeaMARC II data with ground truth ............. 133
   7.4 Classification of Lima Basin .................................. 135
   7.5 Region Growing --ReGATA ...................................... 138
   7.6 Applications of ReGATA ....................................... 146
   7.7 ReGATA Classification of Lima Basin ......................... 152
   7.8 Qualitative Interpretation ................................... 154
   7.9 Quantitative Interpretation .................................. 164
   7.10 Speculative Interpretation .................................. 167

8 SUMMATION ..................................................................... 170

APPENDICES ....................................................................... 173

BIBLIOGRAPHY ..................................................................... 252
## LIST of TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. VCV matrix, eigenvalues, percentage variance per eigenvector, and eigenvectors</td>
<td>.101</td>
</tr>
<tr>
<td>2. Texture vectors for 20m pattern with 0, 12.5, 25, and 50% noise</td>
<td>.113</td>
</tr>
<tr>
<td>3. Feature vectors for the four classes shown in the thematic map in Figure 49</td>
<td>.151</td>
</tr>
</tbody>
</table>
### LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>SeaMARC II towing configuration</td>
<td>8</td>
</tr>
<tr>
<td>2.</td>
<td>Imaging system geometry</td>
<td>11</td>
</tr>
<tr>
<td>3.</td>
<td>Angular dependence of backscatter strength</td>
<td>14</td>
</tr>
<tr>
<td>4.</td>
<td>SeaMARC II angle measurement sub-system</td>
<td>16</td>
</tr>
<tr>
<td>5a.</td>
<td>Unprocessed SeaMARC II side-scan image.</td>
<td>27</td>
</tr>
<tr>
<td>5b.</td>
<td>Gray level histogram for image in Figure 5a</td>
<td>27</td>
</tr>
<tr>
<td>6a.</td>
<td>Image after histogram equalization</td>
<td>31</td>
</tr>
<tr>
<td>6b.</td>
<td>Equalized histogram</td>
<td>32</td>
</tr>
<tr>
<td>7.</td>
<td>Image containing multiple gain changes</td>
<td>33</td>
</tr>
<tr>
<td>8a.</td>
<td>Image after histogram equalization</td>
<td>35</td>
</tr>
<tr>
<td>8b.</td>
<td>Equalized histogram</td>
<td>35</td>
</tr>
<tr>
<td>9.</td>
<td>Image containing errors</td>
<td>37</td>
</tr>
<tr>
<td>10a.</td>
<td>Pixel intensity vs. look-angle</td>
<td>41</td>
</tr>
<tr>
<td>10b.</td>
<td>Image after background subtraction</td>
<td>41</td>
</tr>
<tr>
<td>11.</td>
<td>Slant-range correction geometry</td>
<td>44</td>
</tr>
<tr>
<td>12.</td>
<td>Unprocessed side-scan image</td>
<td>47</td>
</tr>
<tr>
<td>13.</td>
<td>SeaMARC II bathymetry for Figure 12</td>
<td>48</td>
</tr>
<tr>
<td>14.</td>
<td>SeaMARC II image after layover correction</td>
<td>49</td>
</tr>
<tr>
<td>15.</td>
<td>Schematic diagram of migration</td>
<td>51</td>
</tr>
<tr>
<td>16.</td>
<td>Unprocessed image showing &quot;hashmarks&quot;</td>
<td>53</td>
</tr>
<tr>
<td>17.</td>
<td>Image after background subtraction and migration</td>
<td>56</td>
</tr>
<tr>
<td>18a.</td>
<td>Raw SeaMARC II side-scan image</td>
<td>62</td>
</tr>
<tr>
<td>18b.</td>
<td>Image after median value filtering</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>19.</td>
<td>SeaMARC II image of a lava flow.</td>
<td>66</td>
</tr>
<tr>
<td>20.</td>
<td>SeaMARC II bathymetric map of the lava flow.</td>
<td>67</td>
</tr>
<tr>
<td>21.</td>
<td>Edge map of the lava flow image.</td>
<td>68</td>
</tr>
<tr>
<td>22.</td>
<td>Example of Euclidean distance measurement</td>
<td>90</td>
</tr>
<tr>
<td>23.</td>
<td>A two-class clustering example</td>
<td>93</td>
</tr>
<tr>
<td>24.</td>
<td>Examples of Brodatz textures</td>
<td>96</td>
</tr>
<tr>
<td>25.</td>
<td>Synthetic side-scan image of 24 sine waves</td>
<td>97</td>
</tr>
<tr>
<td>26.</td>
<td>Plots of textural parameters vs. wavelength</td>
<td>99</td>
</tr>
<tr>
<td>27.</td>
<td>Classification matrices of Fig. 25</td>
<td>102</td>
</tr>
<tr>
<td>28.</td>
<td>Plot of textural parameters vs wavelength</td>
<td>104</td>
</tr>
<tr>
<td>29.</td>
<td>Plot of textural parameters vs wavelength</td>
<td>105</td>
</tr>
<tr>
<td>30.</td>
<td>Synthetic side-scan image of five sine-waves</td>
<td>107</td>
</tr>
<tr>
<td>31.</td>
<td>Classification statistics for Figure 30.</td>
<td>108</td>
</tr>
<tr>
<td>32.</td>
<td>Synthetic side-scan images with noise.</td>
<td>110</td>
</tr>
<tr>
<td>33.</td>
<td>Classification matrices for noise images</td>
<td>111</td>
</tr>
<tr>
<td>34.</td>
<td>Synthetic side-scan image with rotation.</td>
<td>114</td>
</tr>
<tr>
<td>35.</td>
<td>Classification matrix for rotated image.</td>
<td>114</td>
</tr>
<tr>
<td>36.</td>
<td>Self-reflected synthetic side-scan image.</td>
<td>116</td>
</tr>
<tr>
<td>37.</td>
<td>Plots of textural features vs wavelength.</td>
<td>117</td>
</tr>
<tr>
<td>38.</td>
<td>Combined 2.5 km signal with 20 to 480 m.</td>
<td>119</td>
</tr>
<tr>
<td>39.</td>
<td>Plots of textural parameters vs wavelength</td>
<td>120</td>
</tr>
<tr>
<td>40.</td>
<td>Texture classification of lava flow</td>
<td>123</td>
</tr>
<tr>
<td>41.</td>
<td>Lava flow subjected to enhancement routines</td>
<td>126</td>
</tr>
<tr>
<td>42.</td>
<td>Average angular backscatter for lava flow</td>
<td>127</td>
</tr>
<tr>
<td>43.</td>
<td>Texture classification of enhanced image.</td>
<td>129</td>
</tr>
<tr>
<td>44.</td>
<td>Composite image of dolomite and lava flow</td>
<td>131</td>
</tr>
<tr>
<td>45.</td>
<td>Textural classification of composite image</td>
<td>132</td>
</tr>
</tbody>
</table>
46a. Side-scan mosaic of Lima Basin. . . . . . . .136
46b. Three-class thematic map of Lima Basin mosaic . .136
47. Portion of CDP-1 seismic line . . . . . . . . . .139
48. Three-class ReGATA map of lava flow . . . . . .147
49. Four-class ReGATA map of lava flow . . . . . .149
50. Lima Basin side-scan mosaic . . . . . . . . . .153
51. 12-class thematic map of Lima Basin mosaic . .155
52a. Detail of side-scan mosaic showing knick point. . .157
52b. Detail of 12-class ReGATA map with knick point. . .157
53a. Detail of side-scan mosaic showing dolomite. . . .160
53b. Detail of 12-class ReGATA map with dolomite. . . .160
54a. Detail of side-scan mosaic showing drainage channel .162
54b. Detail of 12-class ReGATA showing drainage channel. .162
55a. Detail of side-scan mosaic showing "mud volcanoes". .163
55b. Detail of 12-class ReGATA map corresponding 55a. . .163
56. Plots of texture vs class for the 12-class map. . . .165
Remote sensing may be broadly defined as the acquisition of information about an object from a distance or without direct physical contact. In this general sense, many of the techniques of geophysics -- magnetics, gravity, seismics, etc.-- may be thought of as remote sensing. More often, however, the term remote sensing refers to the acquisition of synoptic areal imagery of surface terrains from platforms aloft. During the Civil War, photographers were sent aloft in balloons to record troop movements and artillery placement. Balloons soon gave way to airplanes, and these in turn to satellites as the instrument platform. The instruments themselves evolved from black and white still cameras to color cameras, multi-spectral scanners, and other devices. Air- and space-borne platforms and their imaging devices have resulted in rates of data acquisition so rapid that subjective interpretation and editing is no longer sufficient to deal with the reams of data being delivered. Fortunately, the concomitant advance of the digital computer simultaneously relieved the human image-interpreter of significant data handling burdens, and made possible a more quantitative approach, specifically, digital image processing and computer-aided analysis.

The medium for this copious remote sensing imagery is electromagnetic radiation, including the ultra-violet through infrared for passive devices and the microwave bands for active sensors such as imaging radars. Analysis of imagery spanning this spectrum provides valuable information regarding surface conditions, cover, and temporal
and spatial variability. Unfortunately, water is all but opaque to electro-magnetic energy as received at a remote platform. As more than 70% of the surface area of the Earth is covered with water, our ability to gather any synoptic perspective of a large area of the earth is severely hampered. For remote sensing of the seafloor, acoustic methods, to which the seas are transparent are usually used. For acquisition of areal imagery of the seafloor surface, the sensor platform is the surface vessel, the medium acoustic energy in the water, and the sensor is a side-scan sonar.

Side-scan sonar devices have been utilized in surveys of seafloor features since 1958 [Chesterman et al., 1958]. These surveys are the underwater analogs of LANDSAT, SEASAT, and SLAR studies of earth surface terrains. To date, however, the interpretation of these surveys of the seafloor has been primarily qualitative. Thus, for the purposes of geologic studies side-scan sonars have disclosed the location and, to some extent, the morphology of seafloor features, but have provided little of the type of quantitative information necessary for lithologic identification. This quantitative imprecision - in contrast to the successes presently being enjoyed by the terrestrial remote sensing groups - stemmed from both technical hardware problems resulting in a lack of seafloor image fidelity, and the problem of the Gestalt of perception involved with dealing with acoustic images instead of "the thing in itself". Our major perceptual sense is visual; visual perception does not yield knowledge per se of "the thing in itself" but rather inferential knowledge via perception of interaction of electro-magnetic energy with the shape and surface roughness of the object.
Because of this familiarity with ascertaining something about objects from their appearances, we can at least make reasonable inferences about the nature of things from photographs of them, i.e. we can interpret aerial photographs, etc. Sonar images on the other hand are representations of what the object sounds like, i.e. the interaction of acoustic energy with the object surface roughness, specifically at that single wavelength of ensonification. Due to our un-familiarity with that imaging modality, subjective interpretation of side-scan sonar imagery is fraught with additional difficulties. Now, however, the application of objective digital image processing techniques, free (we hope) of prior conceptions of image-object relationships, to images of superior quality and uniformity, makes possible a more quantitative approach in the analysis and interpretation of synoptic seafloor imagery.

In this report an introduction to side-scan systems in general is provided, as well as a brief description of the SeaMARC II (Sea Mapping and Remote Characterization) system which produced our data. Next, a suite of digital processing techniques for the correction, enhancement, and classification of sonar images is provided. Although these techniques are described in reference to the long range SeaMARC II system with which the images shown here were acquired, they are equally applicable to other digital side-scan sonar data, including images from short range systems. Finally, applications of these techniques in the enhancement and classification of a variety of seafloor images are presented.
1.2 Previous Work

Pertinent work in digital image processing is voluminous. Excellent review texts covering general theory as well as diverse applications include Pratt [1978], Andrews [1968], and Ballard and Brown [1982]. Methods directly applicable to remote sensing data are presented by Schowengerdt [1983], Hord [1982], and Moik [1980]. Duda and Hart [1973] present a classic text on pattern recognition. Hartigan [1975] develops the general theory and presents many FORTRAN algorithms for cluster analysis.

Background work for sonars and underwater acoustics in general are provided by Urick [1983]. DeMoustier [1986] and Tyce [in press] provide up to date reviews of acoustic swath mapping systems.


Paluzzi [1976] and Paluzzi et al. [1976] provided a first account of the application of digital image processing techniques to side-scan sonar images. Processing operations included analog-to-digital conversion, partial geometric rectifications, contrast enhancement, and various filtering operations. Walker [1978], Clifford [1979], and Prior
et al. [1979] presented different methods of producing ground speed corrected and geometrically rectified sonar images, although not all are digitally based. Lowenstein et al. [1980] emphasized the importance of producing "mosaics" consisting of overlapping passes through an area and stressed the necessity of geometric and navigational corrections in rendering these mosaics meaningful. Teleki et al. [1981] also produced corrected mosaics, but stressed that total geometric correction would require knowledge of bathymetric information on a pixel by pixel basis, which was not available to them. More recently, Luyendyk et al. [1983] described the results of subjecting digitized side-scan images to the VICAR (Video Image Communication And Retrieval) image processing system at California Institute of Technology's Jet Propulsion Lab. They concluded that the slant-range correction and contrast stretching routines were most successful, but that the sonar images in general possessed S/N (signal-to-noise) ratios too low to benefit from more computationally complex techniques such as inverse filtering. Farre and Ryan [1985] attempted to produce stereo side-scan pairs from SeaMARC I imagery, although they seem to have been caught in the tautology of geometric corrections based upon bathymetry which was derived from interpretation of un-corrected side-scan data.

Finally, in the way of quantitative image analysis and pattern recognition, Pace and Dyer [1979] conducted an analysis of seafloor image texture, based on the measurement of second order gray-level statistics of the image. By combining transformations of their own derivation with those previously utilized in the analysis of sub-aerial imagery [Haralick et al., 1973], they were able to distinguish a number of sedimentary bottom types with greater than 65% confidence. Reut et
al.,[1985] demonstrated an ability to distinguish imagery of six classes of homogeneous sediment type (mud, clay, sand, gravel, cobbles, and boulders) on the basis of image cepstral (log of the image power spectrum) analysis.
1.3 Characteristics of Side-Scan Sonars

A side-scan sonar system is a device for acoustically imaging an area of the seafloor. A transducer array, which acts as both source and receiver for this imaging system, is oriented and shaded so as to radiate a fan-shaped beam with a broad vertical angle and a narrow horizontal angle (Figure 1). The transducer is driven by a pulsed continuous wave (CW) signal, generally of fixed frequency between 3 and 650 kHz [Clerici and Konecny, 1980]. Superior range resolution is acquired by using pulses of short duration, as $R = CT/2$, where $R$ is the range resolution, $T$ is the pulse length in seconds and $C$ is the propagation velocity in meters per second. Signal strength and detection probability, however, are increased by using longer period pulses. The scale of bottom roughness to which the system is most sensitive is directly proportional to the system wavelength. Along-track resolution is directly proportional to the ratio of the system wavelength to the effective aperture of the transducer array.

The transducer array can be housed in a fitting attached to the ship's hull, or pulled behind the ship in a "towfish" deployed near the surface or at depth. Deployment at depth decouples the transducers from the motions at the sea surface, minimizes the effects of refraction due to sound speed variations near the surface, and increases the power that can be transmitted (the cavitation threshold increases as the square of the ambient pressure [Ulrick, 1983]).

The beam produced by the transducer ensonifies a trapezoidal area of the seafloor extending from the array's nadir to the distance athwartships at which the acoustic energy just grazes the bottom.
Figure 1. SeaMARC II towing configuration. The neutrally bouyant towfish is towed at about 100-m depth behind a depressor weight and is stabilized by a long rope used as a drogue. Electronics in the towfish acquire and transmit the side-scan image, attitude data, and the angle of incidence (θ) and range (R) of reflectors. These data are transmitted to the ship, recorded, and later converted to horizontal distance and depth values needed to produce a bathymetric contour map and side-scan image. After [Blackinton, 1983]
Depending on oceanographic conditions, this grazing distance will be around 3 to 4 times the height of the towfish above bottom. Recording the reflections and backscattering of the sound from the ensonified area yields an acoustic image of this portion of the seafloor perpendicular to the transducer. By translating the transducer along the ship's track, and timing the pulses so that consecutive beams will ensonify contiguous areas of the seafloor, an image is constructed of an area of the seafloor. Separate transducers are typically used to simultaneously image both sides of the ship's track. The major long-range (5 km or greater swathwidths) side-scan sonar systems presently being used are GLORIA Mk2 [Laughton, 1981], Swathmap [Andrews and Humphrey, 1980], and SeaMARC II [Blackinton et al., 1983]. Characteristics of the SeaMARC II system pertinent to this report are further described below.
1.4 The SeaMARC II System

SeaMARC II combines a conventional side-scan sonar with a bathymetric mapping system in a single unit towed at depths of 100 meters or less at speeds up to 10 knots. In water depths greater than 1 km, the system produces 10 km wide data swaths, permitting 100% coverage of over 3000 km$^2$ of seafloor per day. A complete system description can be found in Blackinton et al., (1983).

Two parallel, inclined arrays are mounted on each side of the SeaMARC II towfish. The port arrays operate at 11 khz and the starboard at 12 khz. This frequency difference reduces contamination of the signal on one side by reflected energy from the other side (cross-talk). In the water, the outgoing signal may be considered as an annulus of sound expanding as it propagates away from the towfish. Although the radial propagation speed of the annulus in the water column is constant, the speed at which it sweeps the bottom decreases from infinity at normal incidence to the speed of sound in water at grazing incidence. Consequently, in order to generate pixels that represent areas on the seafloor of equal cross-track dimension, the reflected signal must be sampled non-linearly in time. By assuming a nominally flat bottom, and calculating the rate at which the annulus will have swept the bottom for cross-track distances from nadir to 5 km athwartships, the returning signal is divided into 1024 unequal intervals of time, each representing a 5-m-wide swath of the seafloor. Sampling rates are high for the near-nadir pixels and decrease in proportion to the cosine of the grazing angle (Figure 2). If the seafloor is in fact flat across-track, then
Figure 2. A cross-section of the water column indicating the geometry between the imaging system and the terrain. The depression angle, $D$, is the angle between a line horizontal to the earth's surface passing through the sensor and the axis of the sonar beam as it encounters the terrain surface. The compliment of this angle is the look angle, $L$, measured as the inclination of the beam from a line connecting the sensor with the earth's center. The incidence angle "$i$" is measured at the intersection of the axis of the beam and the plane containing the normal to the local true slope $(\tau)$. The grazing angle $(G)$ is the compliment to this angle, and is the more commonly used expression in sonar work. Finally, the apparent slope $(\alpha_{ap})$ is simply the component of the true slope of the local feature in the direction of the plane of ensonification.
this process will produce an image that is geometrically correct in horizontal range. Cross-track topographic variations will however, result in image distortion. Specifics of this type of distortion, and remedies for it, are presented in the Preprocessing Section.

Integration of the signal energy during each sampling interval yields the 1024 pixel values per ping for each side of the image. The echo strength from the portion of the signal representing each 5-m-wide pixel is full-phase rectified, filtered, digitized to 8 bits, and recorded on magnetic tape. The side-scan data are also displayed immediately on shipboard monitor records, with the un-sampled time series plotted in slant range and the sampled pixel amplitudes plotted in a corrected horizontal range projection, at 4 bits. Due to the near infinite speed of the sound annulus at near nadir incidence, and the resulting impractically high sampling rate required of the pixel sample generator, the samples that would have constituted the first 40 pixels per side are discarded. This results in a 400-m-wide (port plus starboard) zone of no information beneath the ship, and a final data set for each side of 984 pixels extending from 200 to 5120 m athwartships.

The along-track distance between sonar pulses will be a function of the ship's speed and the pulse repetition rate. At a typical rate of 1 pulse per 10 seconds, and a ship speed of 8 knots, this spacing is approximately 40 meters. The line spacing distance is used as the nominal along-track pixel dimension for the side-scan data, yielding a digital image with a theoretical ground resolution of 5 by 40 m. We stress, however, that the area of the seafloor ensonified during each pulse is generally much greater than the assumed size of the pixel,
particularly in the along track direction. The effect of this approximation is discussed below.

Three types of amplitude gains are applied to the SeaMARC II sidescan data. The first is a time varying gain (TVG) which is intended to correct for attenuation due to inverse square law spreading and absorption of the signal in the water ( \( e^{-at} \), where 'a' is the absorption coefficient and 't' is time [Urick, 1983].). The second is an angle varying gain, (AVG) which is intended to correct for both the non-uniformities in the transmitted beam pattern and the change in backscattered signal intensity observed at different grazing angles (Figure 3). These two gains are applied automatically, and are not subject to operator intervention. The final gain is a scaling factor which is controlled by the operator in response to the image quality on the ship's monitor recorders.

The assumptions that the seafloor is flat, that the length of the seafloor ensonified by any portion of the beam is the product of the inter-ping period and the speed of the ship, and that the gains have all been applied correctly, are all compromises that produce a distorted image of the seafloor. Our preprocessing steps seek to correct and minimize these distortions, examples of which are presented later in this paper.

The SeaMARC II bathymetric information is acquired through the same transducers as the side-scan data but is processed with different hardware and software. On transmission the transducer pairs on each side of the towfish are driven in parallel. On reception, each row in the pair is sampled independently. Therefore, any signal incident upon
Figure 3. Angular dependence of the strength of signal backscattered from the bottom as a function of angle of incidence at various frequencies (after Brekhovskikh et al., 1982, fig. 1.34).

a) scattering from a smooth bottom; b) scattering from a very rough bottom.
the transducer at any angle off normal to the transducer face is detected at the two rows with a different phase lag (Figure 4), from which the depression angle (θ) of the reflector is calculated. By measuring the round trip travel time and assuming a sound speed of 1500 m s\(^{-1}\), the slant range (R) to the reflector can be calculated. These values for R and θ are converted to across track distance and depth for each reflector, and contoured to produce a bathymetric map. Absolute accuracy of 2-3% of the water depth is nominal; relative accuracy is significantly better.
Figure 4. SeaMARC II angle measurement sub-system. Separation of arrays A and B by spacing D causes any signal arriving at some angle $\theta$ off normal to the transducer face to ensonify the two arrays at different times. Detection of this time lag allows calculation of the angle of arrival.

$$\sin \theta = \frac{C}{FD} \times \frac{\phi}{360^\circ}$$

$C = \text{ACOUSTIC VELOCITY}$

$D = \text{ARRAY SPACING}$

$F = \text{SIGNAL FREQUENCY}$
1.5 Requirements for Image Processing

The technical requirements for quantitative analysis of any form of remotely-sensed, map-format imagery are threefold. First, the image information must be in a digital format. In digital images, the images are composed of discrete picture elements or "pixels". Each pixel is defined by three parameters: (1) its location in some coordinate system; (2) its dimensions on the surface being imaged; and (3) a number that is the average "brightness" of the small area of the seafloor that the pixel represents. This brightness is due to that portion of the integrated backscattered and specularly (mirror-like) reflected energy from the seafloor within the pixel's projected boundaries that is detectable within the system's bandwidth. This energy is a continuous function, but in digital format it must be quantized to finite numbers, contained within some range of resolution. Five bits resolution, providing 32 gray levels, are required to produce a visually continuous image [Schowengerdt, 1983], though 8 bits, resulting in a pixel brightness resolution of 256 levels, are preferable both for numerical analysis and computer hardware and display considerations. Of the long-range systems, only GLORIA MKII and SeaMARC II data are recorded in digital format.

The second technical requirement for quantitative analysis is signal calibration. To calculate acoustic parameters such as impedance or the absolute reflectivity of a surface, or to quantitatively compare the back-scatter strengths of two distinct areas, the output of the sensor should be calibrated through the use of some fixed energy reference source [Swain and Davis, 1978]. Periodic calibration during a survey
also allows the interpreter to account for systemic problems, chiefly beam pattern variations. At this writing, SeaMARC II is being modified to provide this calibration; however, because no currently operational side-scan sonar system has this calibration, only relative measurements of reflectivity can be made. Furthermore, systemic changes must be recognized and accounted for in post-processing.

Third, for the purposes of any interpretation based upon average pixel intensities, the imaging system should produce a uniform response as the system scans from its nadir to its maximum slant range across a uniform terrain. That is, pixels representing similar surfaces should have approximately the same pixel value regardless of their position in the image. Figure 2 demonstrates the difference in the intensity of backscattered signals from different grazing angles, for different frequencies. Clearly, a change in intensity of the order of tens of dB's presents a significant problem to overcome prior to quantitative, or even meaningful qualitative, interpretation. Airborne radar is equally susceptible to this variation of intensity, although satellite based remote sensing systems avoid this problem by surveying from such great altitudes that only a small range of look angles is necessary to scan a large swath of terrain. As the long-range marine systems mentioned above produce swath widths that are primarily greater than water depth, a range of look angles greater than 45° from either side of nadir must be accommodated.
The usefulness and necessity of processing side-scan data in digital format have only recently begun to be properly appreciated. For long-range side-scan systems such as SeaMARC II, which is used to survey large areas of seafloor with 100% coverage, digital data are required to produce mosaics of adjacent tracks providing overlapping images. In order to produce a mosaic, each pixel must be accurately located via iterative re-navigation of ship tracks so that overlapping seafloor images co-register. The brightness of each pixel must then be adjusted so that adjacent tracks have comparable character and can be abutted in a mosaic without producing disconcerting boundaries between the data strips. The point of this adjustment is not aesthetics, but rather avoidance of mis-interpretation of track-to-track intensity variations as significant changes in bottom character. Only with digital data is such level of adjustment practicable.

At the same time that iterative data reduction is becoming routine, researchers are beginning to appreciate the need for quantitative interpretation of the side-scan information. Just as subjective interpretation of aerial photography was for many years considered sufficient for most geologic purposes, so too was subjective interpretation of analog, chart-paper images of the seafloor considered sufficient for the purposes of most marine geologists. Here a quandary has developed, in that as the resolution of seafloor imaging systems has increased, more objects have been discerned about which we know less. Ideally, identification of these unknown terrains could be made on the basis of quantitative analysis of the image, rather than the costly alternative of sampling each new target. Furthermore, spurred by the development of seafloor exploration programs, especially those relating to ferro-manganese crusts and polymetallic sulfides, a need has emerged
for the superior resolution, processing speed, objectivity, and discriminatory capabilities of digital systems augmented by computer-aided analysis.
2. PRE-PROCESSING CORRECTIONS APPLIED TO SeaMARC II DATA

2.1 Objectives

In this report the original side-scan sonar data have been slant-range corrected to a nadir reference datum to produce images wherein the along-track and across-track scales are equal. Ship navigation has been merged with side-scan pixel data to produce images free of distortions due to variable towing velocity. Examination of fish attitude data, recorded every second, indicates that sensor pitch, roll, and yaw are of small magnitude and vary slowly for the images under consideration, and hence present only minor distortions. However, for the production of multi-image mosaics total correction of attitude variations is applied, and the data are converted from a "line and pixel" coordinate system to a Universal Transverse Mercator projection. Given the above in combination with the SeaMARC II bathymetry as the standard SeaMARC II data set, the objectives of the author are threefold:

(1) to produce from the raw images a data set which is a geometrically and radiometrically correct representation of the seafloor,
(2) to enhance the qualitative and quantitative interpretability of these corrected images via contrast enhancements and filtering operations, and
(3) to extract from these corrected images statistical features that can be recognized by a computer, and on the basis of which the
images can be classified into regions of distinct acoustic and, hopefully, lithologic texture.

Our image processing can be considered as four classes of operations, each of which is described in a section below. These are (1) preprocessing corrections, (2) filtering operations, (3) feature extraction, and (4) image classification. The desired result is a thematic classification of the images based upon a concatenation of a textural interpretation with ground truth data. Although the four classes of operations are discussed distinctly, there are many overlaps. For instance, binary thresholding, a type of filtering in which a two-tone picture is produced, may be viewed as a crude classification scheme, in which all pixels below a certain threshold are in one class and all those above it are in another. The point is not to develop a rigorous image processing lexicon, but rather to present a suite of techniques which will provide future users with both base images of superior fidelity, and non-subjective methods for interpretation of these improved images.

2.2 Preprocessing Corrections: Purposes and Definitions

A side-scan sonar image is a two-dimensional display of pixels, each with an associated intensity, which attempts to model a physical realization of a four-dimensional process, namely the interaction of sound with the seafloor. The dimensions of the process are the three cartesian coordinates of space which give the position and orientation of any reflector with respect to the sonar, and the fourth dimension of
acoustic character which includes all geologic information. Consequently, the image is an approximation of the bottom with various sources of errors, including system and operator-induced artifacts, poor contrast, and random noise. Prior to meaningful analysis, these errors must be minimized. The purpose of preprocessing corrections is to produce an image that is a geometrically and radiometrically correct representation of the seafloor. Geometrically correct implies that features are not only in their correct locations, but that they are also represented by the same spatial distribution of picture elements irrespective of the slant-range at which they are imaged. Radiometrically correct implies that a single bottom type should produce a uniform response throughout the image. Only when these two criteria are met can meaningful subjective or quantitative interpretations be conducted.

Pixel radiance transformations include both restoration and enhancement. Restoration implies a transformation that attempts to recreate the original "true image" or remove unwanted noise from the raw image, in a fashion comparable to inverse filtering. Enhancement, on the other hand, seeks only to produce a "better" image; that is, one in which some specific feature of interest is more easily discernible by man or machine. Both are described below.

2.3 Radiance Transformations

Radiance transformations can be of three types: pixel-by-pixel transformations, neighborhood transformations, or spatial transformations. Pixel-by-pixel transformations are the simplest,
requiring knowledge only of the intensity of the pixel and the derivation of a look-up table, which relates the original intensity to the desired intensity. This type of operation has the advantages of being fast enough to implement in real time, computationally inexpensive, and often single-valued, such that any pixel of intensity "I" is always transformed to an intensity "J", within a single application. The histogram modification routines presented below are examples of pixel-by-pixel transformations.

Neighborhood operations require, as the name implies, the evaluation of some function over a neighborhood of pixels surrounding the pixel of interest, in order to determine the scalar transformation to be applied to the value of that pixel. Although the evaluation of the function is constant in method throughout time and space, similar input intensities can result in distinctly different output intensities, as neighborhood operators can be functions of both intensity and relative position of pixels. Substitution of the mean of the neighborhood for the value of the pixel at the center is an example of a neighborhood operation which is not position dependent. Edge detection and point migration as described below are examples of neighborhood operations which are position dependent. In either case, the transformations are rarely single-valued.

Finally, spatial transformations are pixel intensity modification operations wherein the most significant controlling variable is the physical location of the pixel, usually with respect to data indexing (row or column position) or some fiducial reference point in the image. The general assumption is that the image data contain some problem which
is position-dependent, and that by spatial processing we may remove the problem and acquire an image free of spatial dependence. Background subtraction, as described below, is an example of a spatial pixel transformation.

It should be clearly pointed out, however, that no amount of pixel radiance transformations or any other form of image processing can increase the actual information content of the data. This statement at first may seem contradictory, as many transformations appear to make the image more meaningful to the observer, but that is simply a matter of semantics, in that "meaningful" is a function of the observer, and not the image. That an image can be enhanced is de facto evidence that the information visible after the processing step was present in the data prior to any transformation. Furthermore, enhancement of one characteristic is usually accomplished at the expense of another. In image processing, as elsewhere, the second law of thermodynamics still prevails -- there's no free lunch. Furthermore, but for those routines which remove some specific obfuscating errata, enhancement routines which produce "superior" images for human interpretation do not in general produce better data for computer-aided interpretation. If the information is in the data, the computer algorithms can see it without enhancements.

2.4 Contrast Stretching

Contrast, the range of gray levels present in an image, is one of the most important image properties. Schowengerdt [1983] listed
several possible definitions of contrast, including the difference between the maximum and minimum gray levels, their quotient, and the image standard deviation. As S/N may be defined as a function of the image standard deviation [Schowengerdt, 1983], it is clear that higher contrast is an asset. One relative measure of image contrast is the degree to which the image utilizes the range of values available on the imaging and display systems. This use of range can be demonstrated by a gray level histogram, which describes the image in terms of the distribution of the pixels throughout the available gray levels. A histogram is a plot of number or relative frequency of pixels verses pixel intensity value (I.V.). It provides no information regarding spatial distribution of pixels in the image. A side-scan sonar image and its gray level histogram are shown in Figure 5. Note that in SeaMARC II images strong returns (high I.V.) are plotted as dark and weak returns as light, so that the images are the opposite of what one would expect from air photos. Characteristically, because of measures taken during data acquisition to avoid saturation of the recording device, raw histogram plots are skewed and polymodal. The smooth curve in Figure 5b shows most data around I.V. 30, with little utilization of the higher intensity values. To increase the contrast of the image the spread of the histogram must be increased.

A linear stretch is one simple means of increasing the dynamic range of the image [Pratt, 1978]. This procedure entails evaluation of the histogram maximum and minimum, taking their difference, and calculating the new value for the pixel as its fractional part of the optimum dynamic range. Specifically,
Figure 5a. Unprocessed SeaMARC II side-scan imagery presented in plan perspective. Total swath width is 10 km.

5b. Gray level histogram for image in Figure 5a.
Although equation (II-1) will increase the dynamic range of the image, it does so with no regard for the statistical distribution of gray levels in the original image histogram; all contrasts are stretched equally.

Another approach is to redistribute the pixel intensity values over a range of the histogram proportional to their percentage of the original image. This redistribution is implemented by a nonlinear contrast stretch known as "histogram equalization" [Pratt, 1978]. The premise behind this technique is that the highest contrast will be obtained when each value of the dynamic range is utilized equally. Were this the case, then the number of pixels at each value of the histogram would be equal, hence the name.

Contrary to its name, histogram equalization does not produce a flat histogram, but rather one in which the values are separated by amounts proportional to their importance in the original histogram, so that the pixel density along the intensity axis of the new histogram is equal. Hence the contrast between gray levels of frequent occurrence is expanded, while the contrast between those of frequency of occurrence less than the histogram average is compressed. Due to the Gaussian nature of most histograms, histogram equalization usually results in saturation at the extremes where there is less data and expansion of the
means where there is more of the original data. As a result, several input values will be mapped into a single output value, voiding the possibility of comparisons of absolute pixel intensity values between different areas.

An additional caveat in the utilization of histogram equalization is that the histogram of the original image be unimodal [Pavlidis, 1982]. Histogram equalization of a bi-modal histogram will tend to push the data in the two peaks toward each other, resulting in the mixing of the two previously separate intensities. In the image, histogram equalization of an image with a bi-modal image will result in "snowy" blacks and "dirty" whites. Consequently, caution must be exercised in application of this technique to images of strongly bi-modal character, such as those of strongly reflective rock outcrops surrounded by flat-lying sediments which reflect little non-specular energy.

Histogram equalization as described so far is monotonic, in that several input values may be mapped to a single output value, but no input value results in more than one output value. If a truly flat histogram is desired, then this monotonicity must be sacrificed. Pavlidis [1982] described two non-monotonic methods for flattening the histogram. Hummel [1977] displayed images whose histograms have been flattened according to those two rules and concluded that although the histograms are markedly different, the images are virtually indistinguishable from that produced by the monotonic transformation, and hence are not justified computationally. Furthermore, from a statistical perspective, as the non-monotonic transformations do not
treat each input gray level equally, the output S/N is decreased. Consequently, in this paper histogram equalization refers only to the monotonic transformation.

Figure 6a shows the result of applying histogram equalization to the image in Figure 5. Although the equalized histogram (Figure 6b) is still not flat, the contrast and interpretability of the image has increased dramatically. Muted features which would likely be overlooked in Figure 5a are clear in the enhanced image. The FORTRAN code for implementation of our version of histogram equalization, HISTEQ, may be found in appendix A1.

It has been suggested [Frei, 1977] that as the human visual system has a logarithmic response, a hyperbolic gray level transformation would produce the most pleasing image. Pratt [1978] has continued in this theme, producing a number of transform rules for modifying histograms according to Gaussian, catenary, hyperbolic, and exponential distributions. As the choice of transformation that will produce the "best" transformed image will be a function of seafloor character, instrument characteristics, and the prejudices of the interpreter, histogram modification should be considered more of an enhancement than a restoration when used simply to increase image contrast and "interpretability". However, histogram equalization does also possess attributes resulting in restorational capabilities. Equalization of an image histogram results in equalization of local contrast along an image. This equalization is tantamount to removing first order statistical image differences, particularly operator-induced gain
changes. Figure 7 shows a side-scan image in which the manual gain

Figure 6a. Image as shown in figure 5a after histogram equalization.
6b. Equalized histogram.
Figure 7. a) SeaMARC II side-scan image containing errors: 1) gain changes; 2) nadir specular reflections; 3) surface multiples; 4) irregular shading due to improper AVG.

b) Image histogram.
settings have been changed several times, resulting in abrupt changes in image intensity. Although merely a nuisance to the human observer, these changes would produce major confusion for machine based analyses. Figure 8 shows the same image after histogram equalization. Although image quality is subjective, the removal of these spurious intensity changes can only aid interpretation, both subjective and objective. For this reason, Laws [1980] considered histogram equalization a mandatory preprocessing step for textural research in un-calibrated imagery, and hence a form of image restoration.
Figure 8a. Image as shown in figure 7a, after histogram equalization.

Note absence of gain changes, and pronounced shading problem.
2.5 Background Subtraction

The next three preprocessing routines deal with the removal of specific artifacts. Consider Figure 9, in which there are three artifacts in the image. The first is the irregular band of high intensity pixels nearest to the ship's track on the starboard side. We assume these high intensity values are spurious because of the low probability of there being such a narrow, highly reflective feature parallel to the ship's track and located at its nadir for a distance of more than 15 km. The anomalously high intensity of these pixels are most likely due to the contribution from near normal incidence specular returns, which will only occur within the first few degrees from nadir. The predominance of these high intensity returns on the starboard side might then be ascribed to an overall slope of the bottom (down to the port). The second artifact consists of two lines of high intensity pixels parallel to and equidistant from the ship's track on port and starboard sides. These lines of high amplitude pixels are assumed to be spurious by the same logic as above, and have a time delay consistent with their being surface peg-heel multiples of the first bottom echo from the nadir position. This strong first arrival ensonifies the transducer, then continues up to the water surface and is reflected back down to the transducer to ensonify it a second time. Similar dark linear features roughly parallel to the peg-heels but at the outside edges of the image are the first bottom multiples. Again, as with the operator gain changes shown in Figures 7 and 8, once these phenomena are understood they can be properly interpreted by the observer, but would be sources of confusion that must be corrected prior to machine
Figure 9. SeaMARC II side-scan image, containing three types of errors: 1) high-amplitude near nadir specular returns; 2) peg-leg and first bottom multiples; and 3) swath of anomalously low intensity pixels on starboard side due to faulty beam pattern.
processing. The peg-leg and first multiples should not, however, be
summarily removed from the image and disregarded, as they do impart some
useful data; their presence in the image indicates a bottom sufficiently
smooth to produce strong and coherent specular reflections.

The final artifact to be observed in Figure 9 is the swath of low
intensity pixels parallel to and located approximately 3 km athwartships
of the ship's track on the starboard side. This diminution of intensity
is caused by an irregularity in the beam pattern, which the angle
varying gain (AVG) has not been able to correct. The AVG is designed to
correct for beam pattern irregularities and the variation of
backscatter intensity due to change in the angle of incidence over an
average bottom. However, system problems or significant differences
between the bottom being surveyed and the average bottom for which the
AVG was designed can result in severe image degradation. Chavez [1980]
reported similar problems for SLAR (side-looking airborne radar)
imagery. He suggested a correction method based upon subdivision of the
image into non-overlapping regions, calculating a modified estimate of
the means of the boxes, and modifying the pixel values by the ratio of
the mean estimate to a derived estimate of the average image intensity.
Unfortunately, any such area-variable transformation minimizes the
usefulness of comparing measures of either average intensity or variance
between different areas. We have developed a different method here
which results in the rectification of the shading problem as well as the
ghosting and specular problems.
In the author's method, a parameter set is calculated for both port and starboard sides, consisting of an average pixel intensity and standard deviation of strips of the images representing strips of the bottom which would be subtended by 0.25° bins of the beam athwartships. These strips parallel the track of the ship, and span the range of angles from nadir to the least depression angle. These averages should be taken over a substantial portion of a mosaic so that variations in I.V. due to local geologic variations will cancel. The transformation is accomplished by multiplying every pixel in the image located at angle increment "I" by a factor P(I) where

\[ P(I) = \frac{\bar{I}_V}{\text{AVE}(I) + K \times \text{SIG}(I)} \]  

(II-2)

and \( \bar{I}_V \) is the average intensity value of the entire image under consideration, AVE(I) and SIG(I) are the mean and standard deviation for that angle, and K is a weighting factor.

Chavez [1986] suggests a similar operation based upon calculation of the average of each column (e.g. I=1,984) in the along-track direction. Clearly, as the problems which this technique seek to correct are angle variant, his method will only work for data from constant water depth.

The theory behind our operation is that variations in pixel intensity due to geologic variability will be randomly distributed relative to the track of the ship as long as the data set is sufficiently large. Thus, intensity variations due to geologic features will add destructively and only contribute a background level to the
parameter set for that image. As the parameter sets are normalized by the overall average of the image, the gain of the filter is independent of the geology or original gain settings. However, spurious features related to the operating system vary relative to fairly fixed depression angles, adding constructively in the along-track summation, and yielding an estimate of systematically induced cross-track errors. Hence the specular reflections (constrained to near-nadir positions), the peg-leg multiples (constrained to a cross track distance proportional to twice the fish depth), the surface multiples (constrained to a depression angle of 30 degrees), and the beam pattern variations (constrained to fixed depression angles by the shading of the transducers), will all contribute significantly to the average cross track profile. One caveat to this method is that it does not take into account ray-path bending, so that a parameter file generated from a shallow water bottom would not be appropriate for images from significantly (> 1000 m.) deeper waters, or vice versa. A simple solution to this problem which we utilize is to generate parameter files over the range of depths encountered in the survey area, and index them by depth, so that the routine may access them as a look-up table on the basis of the nadir depth of the portion of the image under consideration.

A typical profile calculated in such a fashion for the starboard side data of the image in Figure 9 is shown in Figure 10a. Application of this profile and the corresponding profile for the port data as spatial operators to the image shown in Figure 9, and subsequent histogram equalization, yield the image shown in Figure 10b. The minimization of the three artifacts is evident, while the true geologic
Figure 10a. Plot of average pixel intensity versus look-angle for the starboard half of the image shown in Figure 8a.

Figure 10b. Image as shown in Figure 9 after background subtraction. Note removal of specular reflections, surface multiples, and shading problems.
features are minimally affected. We shall refer to this correction as "background subtraction" as it is largely a correction based upon removal of the average image background.

In areas of relatively constant along-track geology, background subtraction may also allow a means of estimating and correcting for the angle-variant backscattering as seen before in Figure 3. Examples of utilization of the background subtraction algorithms for estimating the backscatter curve of uniform outcrops are presented in the APPLICATIONS SECTION. The algorithm for calculation of the parameter files to be used in correcting the data is AVPR03. Documentation and FORTRAN code are found in appendix A2. The code for application of the parameter files with the modification for variable depth is called ROLDAVD, and is found in appendix A3.

2.6 Geometric Rectifications - Layover

Geometric corrections to the side-scan images include both along-track and across-track rectifications. The across-track correction deals with pixels which have been placed at the incorrect cross-track distance, a phenomenon known as "layover" to the airborne remote sensing community. Reflected side-scan data from each "ping" (outgoing pulse) are acquired sequentially in time, i.e. linear in slant-range. To convert this cross-track "slant-range" image to a plan perspective without a priori knowledge of the bottom topography requires that one assume that cross-track topographic variations about some reference datum, usually the nadir depth, are insignificant. The cross-track
positions of the pixels are estimated from knowledge of the travel time, approximate sound velocity, and nadir depth. Violation of this flat-bottom assumption, as often happens, will result in topographic features being incorrectly positioned, or "laid over" (Figure 11). Because of this flat-bottom assumption, reflections from points A and B in Figure 11, representing off-nadir troughs and peaks, are erroneously rotated along arcs of radii equal to their slant ranges to points A' and B' on the reference datum, and consequently imaged spuriously. As a result, the inward sloping faces of the trough and cliff have been foreshortened and their boundaries misplaced.

This displacement is contradictory to our experience from photographic aerial imagery, wherein the taller objects appear to lean outward, away from the viewer. Furthermore, this distortion becomes more significant closer to nadir, again at odds with our experience with aerial imagery in which the distortion decreases with a decrease in parallax view angle. Finally, in mosaics containing parallel tracks with opposite look directions, similar topographic features seen in different tracks will be displaced in opposite directions, resulting in difficulties in co-registration, not to mention any quantitative analysis.

As an example of the magnitude of this effect, consider an abyssal hill rising 200 meters above a reference datum of 2500 m water depth, and located 2500 m athwartships. The flat bottom assumption will cause the peak of the hill to be imaged 200 m closer to the track than it actually is. This distortion increases with water depth and depression
Figure 11. Slant-range correction geometry. Nominal slant-range correction assumes the bottom is flat (reference datum). Topographic deviations from this reference datum will cause features to be imaged incorrectly, as shown by the compression of the inward-sloping faces of the trough and cliff. See text for details.
angle, so that the same hill at a reference datum of 5000 m water depth would be misplaced by almost 500 m.

Naraghi et al. [1983] describe the same layover problem experienced in side-looking airborne radar (SLAR) images, and an approximate method of correction based upon use of ancillary digital elevation data. Specifically, they generated a synthetic SLAR image from the USGS Digital Terrain Data Set, co-registered the salient features of this image with similar features in the actual image via local correlation techniques, determined the shifts necessary to produce a match for the features they could correlate, and used these shifts to re-locate the rest of the pixels. Although reasonably effective on a large scale (+10 km), this method has the undesirable requirement of introducing the uncertainties of a previously smoothed, unrelated terrain data base, as well as the limitation of using a small number of shifts for many local problems.

As SeaMARC II gathers explicit cross-track bathymetric information with each line of side-scan data, this distortion caused by cross-track relief can be corrected on a local, pixel-by-pixel basis. Under the flat bottom assumption, side-scan pixels are placed at a cross-track distance \( X_{FB} \) equal to \((SR^2 + ND^2)^{1/2}\) where SR is the slant range (sound velocity * arrival time/2) and ND is the nadir depth (reference datum). Our correction of this pixel position is accomplished by locating the side-scan pixel for which the slant range equals that of the nearest bathymetry point, \( X, Z \). (Recall that SeaMARC II bathymetry coordinates
are calculated from measurements of depression angle and time to a reflector, and hence are independent of the side-scan image based on the flat bottom assumption.) When a match is found, that pixel is placed at a cross-track position of the bathymetry point, X, and all those pixels between it and the last pixel so located are stretched or compressed accordingly. The results of this transformation are shown in Figures 12, 13, and 14. Note how the scarp which strikes across the original image in Figure 12 at approximately N45°W is conformed to follow the contours of the bathymetry (Figure 13) after the pixel relocation (Figure 14). The most significant change in geometry is demonstrated by the secondary scarp (arrowed in the raw image and the bathymetry) which is shifted athwartships by up to 1 km and rotated clockwise by almost 20°. An obvious benefit of this rectification is that features are correctly placed on the image for interpretation and survey targeting. A more subtle benefit is the removal from the image of spurious cross-track compressions and rarefactions due to topography which might otherwise be interpreted by both man and the computer algorithms as variations in geologic character.

The only assumption required by the author's layover correction method is that the side-scan data be in the correct sequence athwartships, i.e. that there be no reflector at a cross-track distance \( X_r \) and elevation above the reference datum \( Z \) such that the travel time associated with it would be less than that for any reflector located at some \( X < X_r \). This "correct sequence" assumption is a reasonable assumption for most geometries and bottom types, considering the
Figure 12. Unprocessed side-scan image corrected for slant-range according to the "flat-bottom" assumption.
Figure 13. SeaMARC II bathymetry for the image shown in Figure 12.
Figure 14. SeaMARC II image after layover correction, as described in the text. A 1 km² grid spacing has been superimposed upon the sidescan images for the purposes of comparison. Topographic features such as the ridge (arrowed in the images and bathymetry map) have been shifted by up to one km.
relatively high altitude (height above bottom) - typically greater than 20% of the swath-width - at which SeaMARC II is deployed. The FORTRAN code for implementation of this layover correction is called DELAY2. Often the bathymetry data will require smoothing prior to application of the layover routine. This is accomplished via routine ZFIL/T2, which smooths the data over an operator-specified radius about points in an equally spaced grid. Code and documentation for these two routines are presented in appendix A4.

2.7 Geometric Rectifications - "Point Migration"

The final artifact to be discussed under preprocessing rectifications we refer to as "point migration". As previously mentioned, each pixel is calculated as the integrated intensity of backscattered energy from a 5-m-wide area of the seafloor. The along-track length, y, of any of these areas will be given by $y = BW \times SR$, where BW is beam width and SR is the slant range. From antennae theory [Kinsler et al., 1982] beam width (BW) in radians for a line array is approximately given by the ratio of the wave length of the system to the antennae length. Therefore the along-track dimension of the area ensonified during each ping will increase from nadir to the maximum slant range. This spreading remains uncompensated when the pixels are plotted. Instead, the along track size of each pixel is assumed to be the inter-ping spacing, typically on the order of 40 m. As the beam at most slant ranges will possess a greater along-track dimension, any isolated strong reflector of linear dimension less than the nominal pixel length will be ensonified several times, as shown in Figure 15.
Figure 15. Schematic diagram of the migration of a point reflector, caused by multiple ensonification.
The result of this phenomenon in the side-scan image is demonstrated by the image shown in Figure 16, where point reflectors have been increasingly elongated in proportion to their distance from nadir. The multitudinous "hashmarks" throughout the image are generally interpreted as isolated point reflectors that have been repeatedly ensonified by several adjacent pings and thus plotted as linear features. It is also reasonable to assume that strong reflectors, orders of magnitude larger in area than the beam width, will have been expanded by several additional pixels on either side in the along-track direction in a similar fashion, although this effect is not as immediately conspicuous or deleterious to image quality.

Were this problem of point migration simply a function of beam pattern, we might solve it in a fashion similar to that in which hyperbolic reflectors in seismic data are migrated back to their correct positions by assuming that all points at the same depth have behaved as point hyperbolic reflectors and should all be migrated equally [Hagedoorn, 1954]. However, the extent to which point reflectors in the side-scan data seem to be elongated seems to be a function of both reflector strength and strength in relation to the background. Furthermore, the shadows behind the point reflectors are elongated also, so these too will have to be migrated.

We correct this along track elongation of features in the imagery by developing a migration routine developed by the author wherein the extent of migration (i.e. the number of pings fore and aft of the point of stacking) is a function of both the slant range to the point and the
Figure 16. Unprocessed SeaMARC II side-scan image showing multiple "hashmarks" — short lines parallel to the ship track — caused by the phenomenon of migration.
difference in intensity between the pixel at that slant range and the average intensity of a neighborhood centered about that ping at a lesser slant range. Reflectors at large slant ranges and of high contrast to their neighbors are replaced with a weighted sum of the pixel intensities from pixels fore and aft at the same slant range. From seismic migration theory, one would expect that pixels would be stacked along a hyperbola centered about the high contrast pixel, but with a half beamwidth of $1^\circ$ the slant range would have to exceed 32 km before the curvature of the hyperbola would exceed the pixel dimension of 5 m.

By migrating only those reflectors with high contrast, we minimize the impact of this non-linear, anisotropic low-pass filter upon the rest of the data where hashmarks are absent.

Figure 17 shows the result of application of both background subtraction and the inverse migration routine to the image shown in Figure 16. Note in particular the lower left and upper right portions of the images. The along-track stretching has been greatly diminished, in favor of roughly equidimensional points that are several pixels deep. FORTRAN code and documentation for implementation of the routine MIGRATE are presented in appendix A5.

Both the inverse migration and the background removal techniques emphasize the caveat that one should avoid surveying parallel to linear features. Any linear features oriented parallel to the track will be spuriously enhanced relative to other features, and from the point of
Figure 17. Figure 16 after background subtraction and migration. Note specifically the compression of the hashmarks in the upper right and lower left portions of the image.
view of both the operator or any computer algorithm will be indistinguishable from system-induced errata.

As much as is presently possible, the images have been corrected for the limitations of our observational techniques. After the above pre-processing rectifications, side-scan sonar images have been corrected within the limits of the data, features correctly sized and located, and pixel intensity variations rendered more representative of actual changes in bottom character. Further processing steps are means of extracting quantitative information from the digital images.
3. Image Filtering

3.1

Filtering results in alteration of the importance of certain spatial frequencies relative to the frequency content of the original image. In general, these manipulations may be categorized as high-pass, low-pass, and high-boost [Schowengerdt, 1983]. Low- and high-pass filters are discrete neighborhood transformations. The low-pass filter produces a smoothed version of the original image by replacing each pixel by the average of it and its neighbors. The actual filter, or kernel, is an m-by-n matrix of coefficients, which is convolved with a portion of the image of the same size as the matrix for each point in the original image, by overlapping translations. A high-pass filter is designed to enhance high spatial frequencies, which specifically results in edge crispening. The high-pass filter is easily implemented as an m-by-n unitary gain low pass filter, subtracted from the product of the original value of the pixel at the center of the filter neighborhood and the product of m and n.

3.2 The Box Filter

Both low- and high-pass filters can be implemented simultaneously with the box-filter algorithm [Schowengerdt, 1983]. Conventional implementation of matrix convolution requires m times n multiplications and additions for each pixel, for both high and low pass filtering. However, by calculating column sums over m pixels for all columns of the
image, the box-filter is evaluated by adding n column sums for the first pixel, and subtracting the first column sum and adding the subsequent one for the remaining pixels. Averaging after the "drop one, add one" operation mitigates the possibility of error accumulation due to round-off errors. This boxcar filtering results in a computational savings on the order of n, as well as the generation of a simultaneous high pass image. Subtraction of each summation of n column sums from the value of the pixel at the center of the filter, weighted by the product of m and n, produces a high pass filter. If the center value is weighted by a factor greater than m times n, then a high boost filter results [Schowengerdt, 1983]. This increased weight on the center value produces an image similar to the original with a degree of edge enhancement. FORTRAN code and documentation for our routine MULTILFT, which produces simultaneous low-pass, hi-pass, and hi-boost images, are presented in appendix B1.

It is important when implementing these linear filters that the sampling of the filter match the sampling of the original terrain by the data. That is, if the pixels are not square, then the filter matrix must be proportional to the pixel axial ratio, lest aliasing, or spatial deformation of the image due to un-equal along- and across track sampling, result.

3.3 Median Value Filtering

High- and low-pass filters can also be implemented non-linearly. Median filtering is a frequently used form of non-linear low-pass
filtering [Pratt, 1978]. Median value filtering results in the replacement of the center pixel of an m-by-n window with the median value of that window. As this routine can not benefit from methods such as the box filter algorithm, it is usually computationally expensive. However, median filtering is particularly adept at noise removal. Pratt [1978] compares the effects of mean and median filtering on a variety of signals corrupted by noise. The performance of the median filter is in general superior to that of the mean value filter, as aberrant values are totally excluded, so that signal and noise are effectively separated, rather than averaged in the result. Specifically, a one-dimensional median value filter of length \( L \) will only remove signals of length \((L-1)/2\), where \( L \) is an odd integer greater than 1. Therefore, noise of known spatial wave number can be identified and removed.

Median filtering has previously been costly computationally due to the utilization of sorting routines, which cannot benefit from boxcar type manipulations, to find the median value of the filter kernel at every point in the image. The author presents a new approach to median value filtering, based upon histograms rather than sorting. Consider the histogram of a small portion of the image equivalent to the area spanned by the filter kernel. For a filter of length \( n \), the median value of the pixels in the image under that filter is the intensity in the histogram at which the \( k^{th} \) non-zero entry occurs, where \( k \) is equal to half the filter length. This histogram can be "rolled along" using the drop one, add one technique so that re-computation of the entire
histogram at every point in the image is not necessary. As the histogram method requires only $2 + \frac{n}{2}$ operations per point compared to $n!$ for median value filtering based on sorting, the computation savings is phenomenal. On the Harris 800 computer, median value filtering of images of 2000 by 200 pixels, such as those shown in Figure 18, required 11 minutes CPU time with the routine based on sorting for a filter of length 15, while only 1.5 minutes CPU time for the histogram based method. Our histogram-based median value filtering routine, HIMEFILT, is provided with documentation in appendix B2.

Raw and median-filtered images demonstrate the effect of median value filtering (Figures 18a and b). Note the size-specific removal of hashmarks from the lower right portion of the image. The filter used in this example was a one-dimensional filter of length 15, oriented athwartships. Thus, any feature of cross-track dimension seven pixels or less has been eliminated in the filtered image, and any feature of eight pixels or more was un-effected.

3.4 Edge Detection

Edge detection is an end-member of high-pass filtering. The resultant image is either an edge-enhanced version of the original, an edge map of all edges satisfying some criterion, or a numeric or symbolic display of the edges in a scene, such as rose diagrams used in wind or current direction studies. If an edge is defined as a discrete gray level transition, then detection of edges in a digital image is almost an impossibility, as all pixel boundaries between non-identical
Figure 18a. Raw SeaMARC II side-scan image.

18b. Image after median value filtering with kernel of length 15.
pixels are discrete transitions. Therefore, for any discrete transition
to qualify as an edge, it must meet some criteria, usually a magnitude
threshold or a directional specification, or both. Most edge detection
routines are based on evaluation of the local image gradient and
direction, and require finite difference expressions for the derivative.
Pratt [1978] provides a comparison of the effectiveness of some of the
routines, to which Ballard and Brown [1982] add more recent
developments.

The simplest gradient-based, edge detector can be described by
considering an image $G(i,j)$ where $i$ and $j$ are the row and column indices
respectively and $G(i,j)$ is the pixel value of the $i$th, $j$th pixel. The
vertical edge strength ($GV$) is computed as

$$GV = G(i-1,j) - G(i,j)$$ (III-1)

and the horizontal edge strength ($GH$) as

$$GH = G(i,j+1) - G(i,j).$$ (III-2)

The total edge magnitude equals

$$G'(i,j) = (GV^2 + GH^2)^{1/2}$$ (III-3)

and the direction of the edge, $\theta$, is given by

$$\theta = \tan^{-1}(GV/GH).$$ (III-4)
As the evaluation of any edge strength from two adjacent pixels yields the strength of the edge at the "crack" between the pixels, the method described by equations III-1 through III-4 is referred to as a crack-edge operator [Ballard and Brown, 1982].

In this paper we borrow a superior, though computationally more expensive, routine known as the Sobel edge operator [Pratt, 1978]. For this operator $GV$ is defined, over an image $G(i,j)$ as above, as

$$GV = G(i-1,j-1) + 2G(i-1,j) + G(i-1,j+1)$$
$$-(G(i+1,j-1) + 2G(i+1,j) + G(i+1,j+1))$$

(III-5)

and the horizontal edge strength $GH$ as

$$GH = G(i-1,j+1) + 2G(i,j+1) + G(i+1,j+1)$$
$$-(G(i-1,j-1) + 2G(i,j-1) + G(i+1,j-1)).$$

(III-6)

The total edge magnitude and orientation are defined as for the crack edge operator.

The Sobel operator is symmetric about the central pixel $G(i,j)$, therefore, the new edge magnitude pixel is at the same location as the central pixel. As the edge operator matrix is square, the pixels must also be square. This requirement arises because the edge operator evaluates $dG/dy$ vs. $dG/dx$, so the increments of distance must be equal. To account for this requirement for SeaMARC II images, which have a
typical pixel axial ratio of 1 to 8, square pixels are created by averaging non-overlapping boxes of eight pixels in each ping.

After the edge magnitudes are evaluated for the image, then edge existence is determined by satisfaction of magnitude and direction criteria. These criteria require that an edge exists only if the magnitude exceeds a certain threshold and the direction is within one angle quantization unit of the nearest existing edge. Superior results were obtained if the image was first median value filtered to remove spurious noise points. A raw side-scan image, and its associated bathymetric map, are shown in Figures 19 and 20. Figure 21 is the edge map produced by applying both median value filtering and Sobel edge detection to the side-scan image shown in Figure 19. Future work with edge thinning and linking routines will allow generation of simple closed curves that may be used to delimit more accurately the boundaries of formations for both computer-aided and subjective interpretation. Our routine for edge filtering, SOBEDGE, is provided in appendix B3.
Figure 19. SeaMARC II side-scan sonar image of a lava flow, located 200 km NW of Vancouver. The linear feature in the top right corner is the Queen Charlotte transform fault scarp, with an associated down drop of several hundred meters. Note the 100-meter vent from which the flow has emanated, and its acoustic shadow. Also note the anomalously low reflectivity of the flow in the first kilometer about the ship track.
Figure 20. SeaMARC II bathymetric map of the lava flow shown in figure 14. Note the 100-m plus vent in the lower right and the 300-m plus transform scarp.
Figure 21. Edge map of the lava flow image, resulting from the Sobel edge operator and magnitude thresholding.
4. Feature Extraction

4.1 The Feature Vector

The chief purposes of remote sensing imagery are detection and discrimination. Synoptic imagery displays the areal distribution of terrain surface types, but usually does not describe the individual terrain surfaces explicitly. Although human observers are clearly capable of interpreting such imagery, the results are subjective, not necessarily repeatable, and -- like Rorschach tests -- often more indicative of the interpreter than the object. Statistical analysis of image data provides an objective and repeatable means of identifying, distinguishing, and labeling surface types. Key to this concept of numerical analysis is the term feature vector, a numerical descriptor of N dimensions which will condense the discriminatory information contained in the image plane into a single vector for each surface type. The feature vectors from all surface types in the image supplant the image plane by describing the image in Euclidean N-dimensional feature space.

As an example, gray-tone images might be described by intensity alone, in which case the feature vectors would be one-dimensional and areas of the image between designated thresholds would be classified as distinct (I \leq 50 = class 1, I > 50 = class 2). Feature vectors are usually multi-dimensional, such as from multispectral LANDSAT imagery, and the discriminant surfaces are hyper-planes in N-dimensional space.
For single spectral images such as those produced by side-looking sonars and radars, classification of imagery via intensity thresholding as described above is rarely viable. One reason for the limited usefulness of absolute intensity is the presence in such images of random image noise, or "speckle". This phenomenon takes the form of isolated pixels of intensity values largely un-correlated to their neighbors. In radar imagery this random noise is due to degradation of the signal due to phase interference (coherent fading). In sonar imagery, the most probable causes are ambient noise, water column transmission anomalies (piscine enteric activity?), and electronic noise. The second reason for the limited usefulness of pixel intensity for classification is the limitation of dynamic range imposed by finite recording capacity (usually only 8 bits). As changes in water depth or surface reflectivity can cause variations of echo strength on the order of several tens of dBs, gain settings must be changed, with a consequent loss of any single-valued relationship between intensity as recorded on the image and target strength. Finally, any remote sensing system which sweeps a wide range of incidence angles in scanning across its image swath must deal with the strongly non-linear variations in back-scatter strength with change in angle of incidence (vis. Figure 2). Although angle-varying gains are installed in most systems to account for this back-scattering variation, they can only correct the image in an approximate sense. Local neighborhood standardization techniques (zero mean, unit variance) can remove intensity variations due to the above problems, but at the same time standardization also removes any value
the first order (point) statistics may have had for classification purposes.

Consequently, we look to second-order statistics. First, a note on the concept of order. Order of an image statistic refers to the connectivity of the pixels (or pixel neighborhoods) which the statistic represents. It can be visualized as the geometry required to connect the points under consideration for the evaluation of the statistic. While first order statistics estimate the probability of a monopole landing on a pixel of intensity I, second-order statistics estimate the probability of dropping a dipole on an image and connecting two image cells of specific intensities. It has been shown [Julesz, 1973] that patterns which do not differ in their second-order statistics are perceived as indiscriminable without cognition; the contrapositive has not yet been proven, though it seems to hold.

For both subjective and automated image-interpreters, there are two types of second-order features: abstract and objective. The objective features include those in which the degree of fit between the data and some a priori geometric parameterization is measured. Duda and Hart [1973] describe the Hough transform, a means of determining the existence, though not the location, of features which may be parameterized as curves. Although objective features such as this can be of some use in description of the image, they are of little use as image features. For the classification of single spectral imagery such as side-scan data, texture -- an abstract second-order feature -- is more useful.
4.2 Texture

Texture is an innate property of all objects, which characterizes the closely interwoven relief of the surface. It is not a function of image intensity, but is nonetheless strongly inter-related to it, i.e. an image of constant intensity has no texture, and an image of highly variable intensity must have a strong texture, and vice versa. Texture is strongly stationary and independent of illumination. Although it is quite easy for human observers to recognize and describe texture in empirical terms (smooth, coarse, rippled, etc.), until the past decade texture has been extremely resistant to quantification. Furthermore, measures of image texture have yet to be related to actual object dimensions; texture yields discrimination, but not, as yet, definition.

Most of the current techniques for analysis of image texture can be divided into two categories, the structural and the statistical approach. In the structural approach, an attempt is made to characterize the image texture in terms of its basic elements, which are referred to as primitives. These primitives are the simplest constructional units of the image. For example, in an image of ivy, the leaf would be the primitive. The theory is that if one can determine the parameters which, when applied to some primitive rule, will generate the image, then these parameters may be used to characterize the image in a classification study. Because we do not wish to prejudice our numerical classification of a remotely sensed terrain with a priori
conceptions of the primitives that might compose any area of the seafloor, we will not investigate this method further.

In analysis of image texture, the image must first be divided into non-overlapping sub-sets, referred to as "texels", over which the features are to be evaluated. As texture is inherently a neighborhood property about an image point, texture measures are dependent on the size of the observation neighborhood employed in the analysis. Although it is imperative to restrict the size of the neighborhood to minimize the chance of a mixed texture signature or "mixel", it is at the same time necessary to have a sample large enough to provide a statistically stable estimate of the texture. The signature derived from the analysis of a texel is then used to classify that portion of the image, without necessarily understanding explicitly what either the signature or the resulting classes represent. Several methods have been proposed over the past decade for extracting statistical texture signatures.

4.3 The auto-covariance function

The discrete autocovariance function (acvf) would seem to be a natural means of estimating a second-order process such as image texture. For the discrete, two-dimensional case this is defined as:

$$\text{acvf}(k,k') = \frac{1}{NN'} \sum_{i=1}^{N-k} \sum_{j=1}^{N'-k'} (X_{i,j} - \bar{X})(X_{(i+k,j+k')} - \bar{X}),$$

for $k = 0, 1, 2, ..., N-1$ (IV-1)
where $\bar{x}$ is the sample mean of the whole series and $k$ is the separation between the two points, or lag [Jenkins and Watts, 1968].

Unfortunately, the autocovariance functions of natural scenes are quite similar, as they are dominated by low spatial frequencies. Images of man-made textures, such as agricultural fields, often contain strong periodicities, however few sensor systems have pixels small enough to detect furrow spacings. Furthermore, the acvf as a feature vector is quite unwieldy, and utilization of but its first few moments would be significant only if the acvf were evaluated over a large window; that is, a large number of lags. As the size of the window over which a feature is evaluated increases, so does the probability of encountering two or more significantly different textures, thus producing a feature vector un-representative of either of the textures encountered.

4.4 The power-spectral method

A second method for analyzing image texture [Pratt, 1978] is based on analysis of the texture's 2-D Fourier power spectrum. In this technique, the discrete transform,

$$F(u,v) = \frac{1}{n^2} \sum_{x,y=0}^{n-1} f(x,y) e^{-\frac{i\pi}{n} (ux+vy)}, 0 \leq u,v \leq n-1 \quad (IV-2)$$

is evaluated over subsets of the image and features computed from ring- and wedge-shaped portions of the power spectrum.
Implicit for the efficacy of this transform to estimate the spectrum is the assumption that the input picture \( f(x,y) \) is periodic. As this condition is not met, due to the edges of the window over which the transform is evaluated, spurious "ringing" (the Gibbs effect) is introduced into the spectrum and can strongly limit the usefulness of these power spectral features. Tapering the window can reduce this effect, but as a digitized image is often highly discontinuous, edge effects can still be a problem.

4.5 \textit{Laws' "texture energy transform"}

A third technique introduced by Laws [1980] attempts to describe the image in terms of a texture energy transform. This transform entails convolving the original image with twelve "basis" functions (feature specific band-pass filters). The filters are 2-dimensional separable masks sensitive to spots, edges, and oscillations. A weighted average and standard deviation, which Laws terms a "texture energy transform", are calculated over a window about each pixel of each of these 12 convolved images. Principal component analysis of these 24 "energy transforms" produces the statistical signature with which the image texture is classified.

4.6 \textit{The gray-level methods}

A family of techniques is based upon extraction of second-order gray-level statistics from the differences, run-lengths, and co-occurrences of gray levels. These we shall respectively refer to as the
Gray Level Difference Method (GLDM), the Gray Level Run-length Method (GLRLM), and the Gray Level Co-occurrence Method (GLCM) [Connors and Harlow, 1980]. Each method defines a secondary matrix from the image data from which the texture statistics are extracted. For the GLRLM, the entries to this matrix express the probability of occurrence of a continuous linear feature of length $L$ and gray level $i$ in the direction $0$ in the image. For the GLDM, the matrix is similarly the probability that two pixels separated by a distance $d$ in the direction $0$ will differ in intensity by an amount $D$. Finally, the GLCM method determines the probability of occurrence of a pixel of intensity $i$ neighbored by a pixel of intensity $j$ in the direction $0$ at distance $d$. Clearly, the two latter techniques will be closely related.

Both theoretical [Connors and Harlow, 1980] and applied [Weszka et al., 1976] comparisons determined that the GLCM was comparable to the GLDM and superior to the GLRLM and the power spectral method. In a comparison study, Laws [1980] achieved 87% correct classification of Brodatz textures with his texture energy transform method, while only 73% correct classification with the GLCM. However, violation of pixel volume requirements detailed below may have biased the results against the GLCM.

In a similar study of the effectiveness of the GLCM, utilizing scenes of the same textures as those used by Laws [1980], Vickers and Modestino [1982] achieved a maximum correct classification accuracy of 98%. Similar successes in the range of 80 to 89 percent correct classification were achieved by Haralick et al. [1973] for aerial
photographs, LANDSAT imagery, and photomicrographs of sandstones. Furthermore, as outlined by Laws [1980] the texture energy transform method is not rotation invariant (it gives different results for different look angles) and requires a priori knowledge of the frequencies present to be optimally successful. Due to the preponderance of literature opinion in favor of the efficacy of the GLCM method, and its relative ease of implementation, in this study we have utilized the gray level co-occurrence method for our texture analyses.

4.7 The Gray-Level Co-occurrence Matrix method

As stated previously, the GLCM requires the creation of a secondary matrix from which second-order gray-level statistics are extracted. Specifically, let \( F(x,y) \) represent the digital image over a rectangular domain \( x=1,2,\ldots,L, \ y=1,2,\ldots,M, \) quantized to \( N \) gray levels. Each gray level co-occurrence matrix is a square matrix of dimension \( N \), whose entries \( S(i,j : d) \) express the number of times there occurs in the image a pixel of intensity \( i \) neighbored by a pixel of intensity \( j \) in the direction \( \theta \), at distance \( d \). That is:

\[
S(i,j : d) = \# \left\{ f(x_1,y_1), f(x_2,y_2) : f(x_1,y_1) = i, f(x_2,y_2) = j, (x_2,y_2) - (x_1,y_1) = d \right\}
\] (IV-3)
As a simple example, if the image in the texel is given by

```
1 1 2 3 3
1 2 2 3 4
2 3 3 4 4
1 2 4 4 4
1 1 3 3 3
```

and the gray level co-occurrence matrix is to be evaluated in the horizontal direction for a lag of 1, then the GLC matrix would be

```
   1 2 3 4
1: 4 3 1 0
2: 3 2 3 1
3: 1 3 8 2
4: 0 1 2 6
```

Clearly, the matrix is symmetric, as it is evaluated for the absolute value of the lag; i.e. neighbors to the left and right. Non-symmetric matrices can be generated by considering positive and negative lags separately, although this pairing requires twice the storage space and feature extraction time for a debatable benefit. In this study we will consider only the symmetric matrices.

The elements of the matrix are normalized by dividing each entry in the matrix by the total number of possible entries for that direction and lag. In order to insure that the texture signature of any given
non-isotropic texture is not significantly altered by the angle at which it is imaged, these matrices are evaluated for values of $\theta$ equal to 0, 45, 90, and 135 degrees. Symmetry considerations allow neglecting the respective supplementary angles, as:

$$M(i,j \not\in \theta, d) = M(i,j \not\in \theta + \pi, d) \text{ transpose.} \quad (IV-4)$$

To further insure rotation invariance, Haralick et al. [1973] have suggested taking both the average and the range of the statistics that are extracted at each angle of evaluation. In general, evaluation of the matrices in the four directions given above would allow true rotation invariance for rotations of $\pi/2$ radians, and approximate rotation invariance at $\pi/4$. The author's modification is to impose a normalized, inverse Euclidean-distance weighting scheme upon each neighborhood of dimension $2d$ so that the square neighborhood becomes a circle of radius $d$ and hence rotation invariant.

Specifically, evaluation of texture with the GLCM method for the given angles and $d$ equal to unit spacing requires comparison of the intensity of every pixel in the image to its eight nearest neighbors. Of these eight nearest neighbors, only those four on the north-south and east-west axes are actually at distance $d$. Those pixels on the diagonals are at distance $d/2$. The intensity at the correct distance $d$ along the diagonals may be calculated as follows:

$$F'(\text{unit distance}) = (\sum_{i=1,4} W_i F_i) / \sum_{i=1,4} W_i \quad (IV-5)$$
where
\[ w_i = \frac{1}{\sqrt{(x-x_i)^2 + (y-y_i)^2}}. \]

\( F_i \) are the four pixel values at locations \((x_i,y_i)\), and \((x,y)\) are the coordinates of the point being interpolated.

Haralick et al. [1973] suggested 14 features which can be extracted from the gray level co-occurrence matrices. Evaluation of these 14 features over four directions and several lags makes for a rather unwieldy data set to use directly as a feature vector, to say nothing of being able to visualize the resulting class clustering. Furthermore, for a fixed number of samples, there is an optimal measurement complexity, beyond which increasing the vector dimensionality does not necessarily increase classification accuracy (Schownegerdt, 1983). As classification time for a feature vector of dimension \( N \) is proportional to \( N^2(N-1) \) (Swain and Davis, 1978) smaller feature vectors are better.

Fortunately, some of the features in this set are strongly correlated with each other. The Karheunen-Loev transformation or Principal Component Analysis provides a means for detecting this correlation and reducing the dimensionality of the data set. Appendix CI contains the algorithm we have used for data reduction with PCA.

Briefly, principal components are the eigenvectors of the variance-covariance matrix for the feature vectors from the region under consideration. As the variance-covariance matrix is by nature symmetrical, the eigenvectors will be orthogonal, and hence provide a
new uncorrelated feature set along \( n \) orthogonal axes, with greater feature separability. A new set of \( n \)-dimensional feature vectors can be generated by using the eigenvectors as loadings on the old feature vectors:

\[
Y_1 = a_1X_1 + a_2X_2 + \ldots + a_nX_n
\]

\[
Y_2 = b_1X_1 + b_2X_2 + \ldots + b_nX_n
\]

\[
\vdots
\]

\[
Y_n = m_1X_1 + m_2X_2 + \ldots + m_nX_n
\]  \( (IV-6) \)

where \( Y \) is the transformed feature vector and the \( n \) coefficient sets \( a \) through \( m \) are the eigenvectors.

A simpler approach, utilized in this paper, is to consider the magnitudes of the components of the eigenvectors with the largest eigenvalues. Large eigenvalues indicate those eigenvectors which account for a large percentage of the variance of the data, the equivalent of maximum separability along the transformed axes. The components of the eigenvectors indicate the significance of the original individual features which composed the raw feature vector. By selecting only those original features with large components in the eigenvectors
which account for more than a heuristically chosen significant percentage of the variance of the data, (for example, 80%) one may inexpensively reduce the dimensionality of the data, both in terms of which features need to be calculated and in terms of clustering feature vectors of smaller dimension with greater separability.

4.8 GLCM features

Five of the original 14 features which have shown low correlation in PCA analysis are utilized here. They are:

1) Angular Second Moment: \( \text{ASM} = \sum_{i,j} S(i,j)^2 \)  

\( \text{ASM} \) is a measure of image energy. As the square of the sum of several numbers is bigger than the sum of their squares, this measure will take its largest value for a fixed number of entries when all entries occur at one value of \( i \) and \( j \). Hence this is a measure of image coarseness.

2) Contrast: \( \text{CON} = \sum_{i,j} (i-j)^2 S(i,j) \)  

\( \text{CON} \) is then a weighted measure of the deviation of entries from the main diagonal, and hence a measure of image contrast.
3) Entropy: \( \text{ENT} = - \sum_{i,j} S(i,j) \log(S(i,j)) \)

This feature is largest when all pixel pairs occur with equal frequency, such as in a perfectly random image. Its value decreases as image structure increases.

4) Angular Inverse Difference Moment:

\[
AIDM = \sum_{i,j} S(i,j) / (1 + (i-j)^2)
\]

AIDM is a measure of local homogeneity.

5) Correlation: \( \text{COR} = \sum_{i,j} (i-\mu_x)(j-\mu_y)S(i,j)/(\sigma_x \sigma_y) \)

where \( \mu_x, \mu_y, \sigma_x, \) and \( \sigma_y \) are the means and standard deviations for the row and column sums, respectively. COR is a measure of the linearity of the image.

The means and ranges for these five statistics may be evaluated for one or more distances of separation. The resulting set of statistics forms a feature vector of dimension \( N \), which describes the texture of that portion of the image over which the features were evaluated.
4.9 Evaluation of image texture with the GLCM

To evaluate the texture of an entire image, the image is first divided into small rectangular cells referred to as texture elements or "texels". To optimize the potential of this texture routine, these texels should contain at least $NG^2$ pixels, where $NG$ is the number of gray levels to which the image has been quantized [Pratt, 1978]. Although there is no theoretical basis for this rule, one can intuitively appreciate that as the co-occurrence matrix is intended to approximate the joint-probability distribution for the image, a larger sample will provide a more stable estimate. Furthermore, with an intensity range of 8 bits, pixel-to-pixel intensity transitions of small magnitudes (i.e. 1-2) are more often representative of background image gradients than local image texture. Consequently, the number of pixels pairs sampled should be large in relation to the available range of pixel intensity levels in order to maximize the possibility of measuring real image texture variations.

For images quantized to 8 bits, resulting in 256 gray levels, not only would the evaluation and storage requirements of several 256 by 256 co-occurrence matrices become prohibitive, but the texels would be so large as to be meaningless for all but the most uniform images. Aside from simple resolution considerations, this loss of significance is due to the fact that as texel size increases, the probability of creating a cell of mixed textures (a "mixel") similarly increases.
To minimize the size of the texels without seriously violating the pixel volume requirements, the data are rescaled to a smaller number of intensity values, generally 16 or 32. Historically this rescaling was done via equal-interval quantization [Haralick et al., 1973], which is a modified version of histogram equalization, wherein the output number of gray levels is significantly smaller than the input number. As this step also removes any first order statistical differences, such as spurious variations in the image mean, it has been considered mandatory by most researchers for work with uncalibrated imagery [Laws, 1980; Harlick et al., 1973].

The author has found, however, that if the histogram for the area is bimodal, or if all intensity values are not represented, histogram equalization will result in contrast aliasing. Consider a texel composed solely of equal numbers of pixels of I.V. 0 and 1, on a scale of 0 to 255. Histogram re-quantization to 16 levels will result in an equal number of pixels of D.N. 4 and 12, hence a contrast alias. In general, any re-quantization scheme which is a function of the contrast of the original scene will result in an alteration of the scene's second order statistics. As all raw texels will not have equal contrasts, the degree of alteration of the second order statistics will vary between texels, obfuscating any real textural differences. Thus we re-quantize our data without regard to contrast, by dividing each input intensity by the maximum of the desired range of quantization, and rounding to the nearest integer. For the above example, the transformed texel would consist entirely of pixels of value 1, which is an acceptable re-quantization for an image that was essentially noise to start with.
Following this re-quantization, co-occurrence matrices are evaluated for each texel, and the desired feature vectors are then extracted. The remaining task is to subdivide the image into texturally distinct regions on the basis of each texel's n-dimensional feature vector. Our FORTRAN code for implementation of the gray-level co-occurrence method of texture analysis is called GLCM8. The code and documentation for utilization are presented in appendix D1.
5. **Classification:**

5.1

The object of classification is to assign each pixel or texel of an image, defined by its feature vector, to a certain class, in such a way that the specified error criterion is minimized. In order to do this the computer must be able to determine the classes that exist in an image. This list of classes must be exhaustive; all classes present in the image or feature data must be represented. These classes should be both separable in feature space (the domain spanned by the feature vectors) and of informational value. Having determined the classes present in the data, the computer must determine to which class every pixel or texel is most likely to belong. The desired result is a map in which the pixels or texels of the original image have been replaced with symbols, representing the thematic group to which the feature vector for that point belongs. There are, in general, two approaches to achieving this end: supervised and un-supervised classification.

5.2 **Un-supervised Classification**

Un-supervised classification is best described as clustering. The texels, or specifically their associated feature vectors, of a sub-set of the image are submitted to a clustering algorithm to determine the natural groupings in the data that exist in n-dimensional feature space, where n is the number of statistical features (extracted from the co-occurrence matrices). By submitting a small, heterogeneous sample
population of texels, it is hoped that representative natural groups may be determined. Although these numeric groups may or may not have discernable properties in the image or in the physical world, they may still be used as the classes into which all remaining feature vectors may be classified. Meaningful classes could then be extracted with the aid of ground truth and historical data.

The clustering algorithm utilized in this work is a modified version of the classic K-means algorithm [Hartigan, 1972]. The traditional algorithm is initialized by assuming the existence of K classes, with K to be specified by the operator, for which K approximate initial mean vectors are chosen. The initial K mean vectors may either be chosen randomly, assigned by the operator, or defined by the data. In the latter approach, which we utilize, the scalar sum of each vector’s components is calculated, and the vector is initially assigned to that class K for which its sum represents the kth fractional part of the range of the sums. The mean vector is the vector that defines the centroid of the K-th cluster. Every texel of the training data set is then assigned to that class to which its feature vector is least distant. Distance is often measured in terms of the Euclidean distance, defined as

$$ED_{aM_j} = \left[ \sum_{i=1}^{n} (a_i - M_{i,j})^2 \right]^{1/2}$$ (V-1)
where a is the length-n feature vector, M is the length-n centroid vector of the j-th cluster, and ED is their Euclidean distance (Figure 22). After all feature vectors have been assigned to that cluster from which they are minimally distant, then the cluster means are recomputed, and the process iterates until there is no significant change in the pixel assignments from one iteration to the next. In Hartigan’s approach, K classes are assumed at the start. If, at the end of an iteration, one class is found to have no members, the routine terminates, usually with an unsatisfactory classification. The modification developed in this paper, in the event of an empty class, is to search the remaining classes for the class with the highest variance, split it, re-partition the vectors, and continue the iterations. A further modification we have added is to compute the intra- to inter-class variance ratio (F-statistic) for each of the classes at convergence. Any class whose F-statistic exceeds a threshold T is split, increasing the number of classes to K+1, and the algorithm re-initialized. The algorithm for this clustering step, CLUSTER1, may be found in appendix E1.

5.3 Supervised Classification

Supervised classification can be used when one has supplementary information indicating both the existence and location of homogeneous, representative samples of every class in the image. This supplementary information usually comes in the form of historical data (maps, reports, etc.) or ground truth data (samples, photographs, etc.). It is important that all classes are represented and that the samples used to represent
Figure 22. Graphic example of Euclidean distance measurement. $M_1$ and $M_2$ are the centroid (mean) vectors for the two classes. $X$ is a vector to be classified. The Euclidean distance is measured as shown by the dashed lines. Assuming equal class variances, the vector $X$ would be assigned to the class represented by $M_1$. 

$$D = \frac{1}{2} \left( (X - m_1)^2 + (X - m_2)^2 \right)$$
them contain no "mixels". Once a representative data set is obtained, the average textural vector and its standard deviation are calculated for each class; all other texels may be placed in the class to which they are most similar, according to one of several classification routines. Several, currently available routines are the minimum-distance classifier [Scholz et al., 1979], the Bayes Maximum-Likelihood approach [Duda and Hart, 1973], and ECHO, Extraction and Characterization of Homogeneous Objects [Kettig and Landgrebe, 1976]. In their comparison of these and several other classification schemes in the classification of LANDSAT MSS data for agricultural areas, Scholz et al. [1979] detected a slight but consistent superiority of the minimum distance approach over the other methods. We implement a modified version of this classifier for our data. In the classic approach, the centroid vectors are determined for all known classes in the image. Then all remaining texels are placed in the class from which their feature vector is minimally distant. Distance measures often used include the Euclidean distance defined above, and the city block distance, defined as

\[ D = \sum_{i=1}^{n} |x_i - M_i| \]  

(V-2)

where, as in equation (V-1), D is the distance, x is the feature vector, M is the centroid vector, and n is the number of features in the vector. Readers noting the similarity between minimum distance classification and the K-means algorithm should recall that no re-computation of class
centroids occurs during minimum distance classification, as it is assumed that the centroids are already known.

The assumption generally made in implementing this routine is that the classes have approximately equal variances, so that the line, or the hyper-plane in n-space, separating the populations of any two classes would be located half way between the two centroids. If this assumption of equal variances is invalid, mis-classification will result.

The author's modification to the minimum-distance supervised classification routine accounts for the potential problem of unequal variances by normalizing each comparison of feature vector and class centroid by the variance of each feature in that class. This normalization requires a priori knowledge of class variances, presumably derived simultaneously as the class mean vectors in the un-supervised clustering. Figure 23 demonstrates the minimum distance classifier, and our modification. The algorithm for implementing this classification is MINDIST (appendix E2).

An approach used in satellite remote sensing employs un-supervised classification to determine the natural texture classes, refines these classes with any ground truth data available, and uses these refined classes as the known classes in a supervised classification scheme. If the ground truth, or operator knowledge, has allowed the true nature of the refined classes to be determined, the final result may be considered a thematic map, relating known statistical signatures to specific lithologies. The implication of correlation of surface types with
remotely sensed signatures is that once a statistical signature has been related by sampling to a specific lithology, similar lithologies may be tentatively identified at other locations simply by their textural signature, acquired through the remotely sensed image, augmented by a minimal amount of sampling. A caveat to this conclusion for systems such as SeaMARC II which map terrains from variable altitudes is that identical terrain surfaces in significantly disparate (>1000 m) water depths may yield distinct textural signatures, due mainly to the variations in ray paths and beam spreading concomitant with change in water depth. Empiric evidence has yet to decide the significance of this effect, due chiefly to the absence of verifiable ground truth.
6. THEORETICAL INVESTIGATION OF TEXTURE ANALYSIS

6.1 Method

Most studies of the efficacy of methods of analysis of image texture utilize images from an album of natural textures compiled by Brodatz [1966], which are referred to as "Brodatz textures" (Figure 24). Although such standard images are useful for comparison studies, our purpose is not to prove the superiority of any particular texture analysis method, but rather to investigate the robustness of the GLCM method and to quantify the nature of the textural signatures obtained from it. To this end, we have generated synthetic patterns in the format of side-scan images. The generating function for these synthetic images is a sine wave, of variable amplitude, intensity, wavelength, orientation, and noise content. The physical analogues of these sine waves might be ripple marks or sand waves. Properties of the GLCM texture analysis method which we wish to investigate include wavelength sensitivity, stationarity, noise sensitivity, rotation invariance, and the effects of mixing patterns of two different wavelengths.

6.2 Wavelength sensitivity

Figure 25 displays a synthetic side-scan image consisting of 24 sine-wave patterns, ranging in wavelength from 20 to 480 m in increments of twenty meters. The format is identical to that for an actual side-scan image, and each ripple pattern consists of six lines of 984 5-m
Figure 24. Examples of Brodatz textures: (a) tree bark, (b) calf leather, (c) wool, (d) beach sand, (e) pigskin, (f) plastic bubbles, (g) herringbone weave, (h) raffia, (i) wood grain
Figure 25. Synthetic side-scan image consisting of 24 sine waves of amplitude 255 and wavelength from 20 to 480 meters.
pixels. In each ripple the amplitude range is 0 to 255, so that first order statistics of the 24 ripples are identical. Although the ripples are visually distinguishable on the basis of wavelength; their textural distinction remains to be shown.

For each of the images analysed in this theory section, the ripple image is first subdivided into texels containing 6 lines and 49 pixels. As each pattern is generated with six lines, no mixed texels result. Upon each of these texels GLC matrices are evaluated at unit lag and for the four directions of 0, 45, 90, and 135 degrees. From these four GLC matrices, the five statistics previously discussed in the FEATURES section -- CON, ASM, ENT, ENR, and COR and a sixth of our own derivation -- are evaluated. The sixth feature we refer to as isotropy. It is calculated as the difference of orthogonal GLC matrices, and is a measure of the isotropy of the image. The standard feature vector evaluated for the remainder of these tests consists of the means of each of the five statistics evaluated over the four directions, and ISO, which requires no averaging. The texels are classified on the basis of their feature vectors via our modified K-means clustering algorithm.

Figure 26 shows the relationships between these six texture statistics and wavelength. Three of the statistics -- CON, ASM, and ISO -- are monotonic, single-valued functions of wavelength. The third statistic, ENT, is nearly so, and the statistics ENR and COR are scattered, although still visibly related to wavelength. A vector consisting of the statistics CON, ASM, ENT, and ISO is an optimal measure of dominant image wavelength.
Figure 26. Plots of the six textural parameters CON, ASM, ENT, ENR, COR, and ISO versus wavelength, as derived via the GLCM method described in the text from the synthetic image shown in Figure 25.
Table 1 depicts the results of principal component analysis of the 24 feature vectors plotted in Figure 26. The eigenvalues indicate that the sixth eigenvector, which accounts for 77% of the variance, is the most important. The largest loadings in eigenvector 6 are on components 1-3 and 6, in accordance with estimation from the plots of texture vs. wavelength.

Figures 27a, b, and c show in three parts the results of textural analysis and classification of the image shown in Figure 25. The three parts correspond to sections of Figure 25 containing sine waves of wavelength equal to 20-160m, 180-320m, and 340-480m, respectively. The 8 by 20 matrices show the classification of the 160 texels in each third of Figure 25. Note that in Figure 27a all the texels in each row are classified similarly, so that the classification accuracy is 100%. Classification accuracy decreases to approximately 50% for the portion of Figure 25 classified in Figure 27b, and to effectively zero for the last third classified in Figure 27c. The reason for this decrease in separability can be found in the behavior of the curves shown in Figure 26. The curves for all of the features used in the classification flatten with longer wavelengths. The third feature, entropy, also becomes non-single valued, with inflection points at 250 and 350 m. This decrease in discriminability with increase in wavelength points to a possible relationship between texel size in relation to signal wavelength and the viability of the texture signature. The classification accuracy drops dramatically after the fourth row of Figure 27b, corresponding to a wavelength equal to 240 m.
TABLE 1. VCV MATRIX, EIGENVALUES, PERCENTAGE VARIANCE PER EIGENVECTOR, AND EIGENVECTORS FOR 240 TEXTURE VECTORS.

VARIANCE-COVARIANCE MATRIX

\[
\begin{array}{cccccc}
0.0695 & -0.0760 & 0.0891 & 0.0282 & 0.0138 & 0.0771 \\
-0.0760 & 0.0690 & -0.0788 & -0.0204 & 0.0891 & 0.0714 \\
0.0282 & -0.0204 & -0.0788 & 0.0891 & 0.0714 & 0.0788 \\
0.0138 & -0.0232 & -0.0206 & -0.0093 & 0.0714 & -0.0245 \\
0.0771 & -0.0949 & -0.0829 & 0.0140 & 0.0297 & 0.1045 \\
\end{array}
\]

THE 6 EIGENVALUES ARE:

0.0000 0.0001 0.0020 0.0360 -0.0093 0.0140

PERCENTAGE OF TOTAL VARIANCE CONTRIBUTED BY EACH EIGENVALUE

VARIABLE: 1 2 3 4 5 6

CON :0.4352 0.1807 0.7254 -0.1546 -0.1860 -0.4396
ASM :0.8204 -0.0774 -0.1883 -0.1535 0.0303 0.5109
ENT :0.1498 0.8494 0.1900 0.0746 0.0726 0.4574
ENR :0.1058 0.2367 -0.3520 -0.7796 -0.4283 -0.1328
COR :0.0097 0.0409 0.0289 -0.4539 0.8758 -0.1560
ISO :0.3221 0.4269 -0.5268 0.3649 0.0934 -0.5428
Figure 27. Classification matrices of the synthetic side-scan image.

Each wavelength pattern in the synthetic image was divided into 20 texels. The three matrices a, b, and c represent classification of portions of the synthetic images containing the 20 to 160 m, 180 to 320m, and 340 to 480 m wavelength patterns respectively. Note the marked decrease in classification success after the fourth row in b, corresponding to a wavelength of 240 meters.
As 240 m is approximately equal to the texel dimension, (49 pixels * 5 m per pixel) signals with wavelengths greater than the texel size will be only partially sampled, leading to higher intra-class variance and lower classification accuracy. To further dramatize this, we present in Figures 28 and 29 plots of texture features verses wavelength similar to Figure 26, but for texels of 120 and 500 meters square, respectively. For the 120 m texel, Figure 28 shows that texture features CDN, ASM, and ISO flatten entirely for wavelengths 360 m and greater yielding no discrimination capability, and that feature ENI is single-valued only up to the 120 m wavelength, and then experiences marked cyclicity. Figure 29 on the other hand, for the 500 m texel, shows that all four features CDN, ASM, ENT, and ISO have become more linear and separably single-valued at longer wavelengths.

From this increase in linearity in the relationship between texture features and wavelength one might conclude that, as usual, bigger is better, but for the omnipresent increase in mixel probability as texel size increases. For the purposes of minimizing confusion as to sources of error, further analyses are limited to the first section of Figure 25 containing the perfectly discriminable patterns of wavelength equal to 20-160 m.
Figure 28. Plot of the six textural parameters versus wavelength for a texel of 120 meters on a side.
Figure 29. Plot of the six textural parameters versus wavelength for a texel of 480 meters on a side. Note the general similarity to the plots in Figure 26.
6.3 Stationarity

As mentioned above (section 4.9), the pixel intensities within the individual texels are re-quantized to 16 intervals, without standardization. For the purposes of classification therefore, the individual features extracted from the gray level co-occurrence matrix must be independent of absolute intensity. This is particularly important in images wherein the gains have been changed, resulting in scalar offsets in the image intensity data.

To investigate the behavior of our six texture features (CON, ASM, ENT, ENR, COR, and ISO) we have generated the synthetic image shown in Figure 30. The five patterns are sine wave of constant amplitude (64) and wavelength (100 m) whose average intensities range from 32 to 160 in steps of 32. Figure 31 shows in part (a) the result of textural analysis and classification of the five patterns. The near-perfect column-wise segregation of the image indicates a greater variation within the patterns than between them. Specifically, patterns 2-5 are indistinguishable, irrespective of an almost three-fold variation in average intensity. Inspection of the class mean vectors, shown in Figure 31b reveals that classes 1 and 2, by which pattern 1 is discriminated from the other four, differ from classes 3 and 4, their counterparts, only in the feature COR. Exclusion of the feature COR from the feature vectors should then result in a feature vector which is independent of absolute intensity shifts. Classification of the five patterns with the reduced feature vector resulted in the texel classification shown in Figure 31c. The five patterns of different
Figure 30. Synthetic side-scan image consisting of five sine-wave patterns of constant amplitude (64) and wavelength (100 m), but different average intensities. Average intensities from top to bottom are 32, 64, 96, 128, and 160.
A) 

TEXEL CLASSIFICATION 

<table>
<thead>
<tr>
<th>TEXEL #</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAT.#</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>-------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

B) 

TEXTURE FEATURES

CLASS MEAN VECTORS, CON ASM ENT ENR COR ISO

<table>
<thead>
<tr>
<th>PAT.#</th>
<th>CON</th>
<th>ASM</th>
<th>ENT</th>
<th>ENR</th>
<th>COR</th>
<th>ISO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.9808</td>
<td>1.4375</td>
<td>2.7836</td>
<td>0.3595</td>
<td>1.2317</td>
<td>0.5625</td>
</tr>
<tr>
<td>2</td>
<td>-0.9268</td>
<td>1.4062</td>
<td>2.8881</td>
<td>0.3041</td>
<td>1.4476</td>
<td>0.5937</td>
</tr>
<tr>
<td>3</td>
<td>-0.9808</td>
<td>1.4375</td>
<td>2.7836</td>
<td>0.3595</td>
<td>1.8892</td>
<td>0.5625</td>
</tr>
<tr>
<td>4</td>
<td>-0.9268</td>
<td>1.4062</td>
<td>2.8881</td>
<td>0.3041</td>
<td>1.9078</td>
<td>0.5937</td>
</tr>
<tr>
<td>5</td>
<td>-0.8616</td>
<td>1.3661</td>
<td>2.9342</td>
<td>0.2719</td>
<td>1.8305</td>
<td>0.6339</td>
</tr>
</tbody>
</table>

C) 

TEXEL CLASSIFICATION -- REDUCED FEATURE VECTOR

| TEXEL # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|
| PAT.#   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| ------- |---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|
| 1       | 4 | 3 | 5 | 4 | 5 | 4 | 5 | 3 | 5 | 3 | 5 | 4 | 5 | 3 | 3 | 5 | 3 | 4 | 3 |
| 2       | 4 | 3 | 5 | 4 | 5 | 4 | 5 | 3 | 5 | 3 | 5 | 4 | 5 | 3 | 3 | 5 | 3 | 4 | 3 |
| 3       | 4 | 3 | 5 | 4 | 5 | 4 | 5 | 3 | 5 | 3 | 5 | 4 | 5 | 3 | 3 | 5 | 3 | 4 | 3 |
| 4       | 4 | 3 | 5 | 4 | 5 | 4 | 5 | 3 | 5 | 3 | 5 | 4 | 5 | 3 | 3 | 5 | 3 | 4 | 3 |
| 5       | 4 | 3 | 5 | 4 | 5 | 4 | 5 | 3 | 5 | 3 | 5 | 4 | 5 | 3 | 3 | 5 | 3 | 4 | 3 |

Figure 31. (a) Classification matrix for Figure 30. Note that only pattern 1 is discriminable from the others. (b) class mean vectors for the classes mapped in Figure 31a. (c) classification matrix for Figure 30 after the feature COR has been deleted from the feature vector.
average intensity are indistinguishable, consistent with our expectations.

6.4 Noise sensitivity

We next investigate the effects of random noise upon textural analysis and classification. Figure 32a shows the first 8 patterns of Figure 25. Figures 32b, c, and d demonstrate the visual effect of corruption of the patterns in Figure 32a by 12.5, 25, and 50 percent random noise, respectively. Even with 50% random noise, the general oscillatory nature of the signals is still evident. Figures 33a, b, and c show the textural classifications and mean class vectors for the three noise corrupted images. At 12.5% noise, classification accuracy is reduced to 50%. At 25% noise only the shortest wavelength signal is perfectly discriminated, and the second and third less so, yielding an accuracy of approximately 25%. Finally, with 50% random noise, the signals are texturally indistinguishable.

Insight as to the noise sensitivity of individual texture feature components is gained by comparing the components from vectors representing similar wavelengths in the uncorrupted and noisy images. For example, in Table 2, which gives the feature vectors for the 20 m pattern with 0, 12.5, 25, and 50 percent noise, CON varies by less than 5% of its non-noise corrupted value. At the other extreme, the fourth component, ISO, changes by more than 75%, perhaps indicating that a smaller dimensional feature vector is superior for noisier images.
Figure 32.  

a) synthetic side-scan image with 20 to 160 m wavelength patterns,  
b) image as in (a) but with 12.5% random noise,  
c) image as in (a) but with 25% random noise,  
d) image as in (a) but with 50% random noise.
### Figure 33a.

Classification matrix for Figure 32b, containing 12.5% noise.

### Figure 33b.

Classification matrix for Figure 32c, containing 25% noise.

### Figure 33c.

Classification matrix for Figure 32d, containing 50% noise.
6.5 Rotation Invariance

The fourth property for investigation is rotation invariance. The top half of the image in Figure 34 shows five patterns of wavelength 160, 80, 40, 20, and 10 m. In the bottom half of Figure 34 the same five patterns are shifted by one lag per line number within each pattern to produce a 45° rotation. The rotation appears less than this in the image because in the absence of navigation data, the plotting routine used herein assumes an inter-line spacing of 35 m, resulting in elongation of the image along-track and diminution of the apparent angle.

As mentioned above in the Feature section, rotation invariance is usually approximated by averaging features over various look directions. This averaging is more effective if the co-occurrence matrices in the diagonal directions are evaluated over a modified version of the original image data, filtered with the inverse-distance weighting scheme described above. Even so, analysis of the original six texture components indicated that while CON, ENT, ENR, and ISO were invariant under rotation, ASM and COR were not. Classification of the ten patterns in Figure 34 required 6 classes (Figure 35). Classes 1 through 4, and 6 represent the 160, 80, 40, 20, and 10 m rotated and un-rotated patterns. Class 5 represents the void space generated by the offsetting of the pixels in the rotated image.
TABLE 2. TEXTURE VECTORS FOR 20m PATTERNS WITH 0, 12.5, 25, AND 50% NOISE.

<table>
<thead>
<tr>
<th>% NOISE</th>
<th>CON</th>
<th>ASM</th>
<th>ENT</th>
<th>ENR</th>
<th>COR</th>
<th>ISO</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>1.2558</td>
<td>0.2219</td>
<td>3.2783</td>
<td>0.3285</td>
<td>1.4253</td>
<td>1.5375</td>
</tr>
<tr>
<td>12.5</td>
<td>1.2449</td>
<td>0.2346</td>
<td>3.6135</td>
<td>0.3041</td>
<td>1.4476</td>
<td>1.1522</td>
</tr>
<tr>
<td>25.0</td>
<td>1.2161</td>
<td>0.2092</td>
<td>4.1630</td>
<td>0.3595</td>
<td>1.8892</td>
<td>0.9054</td>
</tr>
<tr>
<td>50.0</td>
<td>1.2500</td>
<td>0.2359</td>
<td>4.4314</td>
<td>0.3041</td>
<td>1.9078</td>
<td>0.4348</td>
</tr>
</tbody>
</table>
Figure 34. Synthetic side-scan image. The five patterns in the lower half of the image are identical to those in the upper half, but for translation by one pixel per line of the pattern to produce a rotated image. The five patterns correspond to wavelengths of 160, 80, 40, 20, and 10 m.

<table>
<thead>
<tr>
<th>WAVELENGTH</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>160m</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>80m</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>40m</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>20m</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>10m</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

Figure 35. Six-class classification matrix for the synthetic image in Figure 34. Classes 1-4 and 6 represent the 160, 80, 40, 20, and 10m rotated and un-rotated patterns. Class 5 represents the void space created during rotation.
6.6 Wavelength mixing

The final variable considered in this section is wavelength mixing. The presence of two or more simultaneous scales of roughness is a common phenomenon on the seafloor, as it is on other terrains. Long wavelength features such as abyssal hill constitute the deterministic topography; features shown on standard bathymetric maps. Short wavelength features constitute the surface roughness, usually on the order of centimeters to tens of meters in amplitude and meters to hundreds of meters in wavelength. With the limitation of 5 m pixels, we cannot produce synthetic side-scan images with which to study the effects of centimeter scale roughness, we can at least study the effects mixing moderate and long wavelengths.

Figure 36 shows a synthetic side-scan image created by combining the image shown in Figure 25 with itself reversed; that is, pattern 1 consists of a 20 m sine wave superimposed on a 480 m sine wave, pattern 2 a 40 m wave and a 460 m wave, etc. until the pattern reflects itself. In Figure 37 we have plotted values of our six texture features against the lower order wavelength. As expected, the plots are symmetrical about the wavelength equal to 250 m. What is interesting, however, is the nature of this symmetry. For the features CON, ASM, ENT, and ISO, there appears to be little or no contribution from the longer wavelengths. Comparison of the plots in Figure 37 with the plots for the same features in Figure 26 indicates little difference in either the nature of the curvature or the absolute value of the texture feature
Figure 36. Synthetic side-scan image created by combining 24 patterns ranging in wavelength from 20 to 480 meters with 24 patterns spanning 480 to 20 m.
Figure 37. Plots of the six textural features versus ascending wavelength for the image in Figure 36. Note symmetry about 240 meters.
between the wavelengths of 20 to 200 m, indicating the dominance of the contribution of the short wavelength features to texture, when the two scales of roughness are within an order of magnitude of each other.

Figures 38 and 39 show an interesting result of mixing a 2.5 km wavelength signal with the original pattern shown in Figure 25. On comparing the plots in Figure 26 of texture verses wavelength against similar plots in Figure 39 for the mixed patterns shown in Figure 38, one can see that while textures CON, ASM, ENT, and ISO are minimally effected by the presence of the long wavelength signal, textures ENR and CDR have changed from scatter plots to reasonably straight lines. As this texture analysis was carried out with the same texel size of 250 m sq., this increase in linear relationship between measures of texture and wavelength would seem to indicate there is some beneficial effect to be gained from low frequency heterodyning (mixing the 2.5 km signal with the more variable high frequency background variations). The effect of such low frequency heterodyning upon classification accuracy of real data remains unassessed.
Figure 38. Synthetic side-scan image generated by combining a 2.5 km wavelength signal with 24 patterns spanning the range of 20 to 480 meter wavelengths.
Figure 39. Plots of the six textural parameters versus wavelength. Note decrease in scatter for features 4 and 5 when compared to Figure 26.
7 APPLICATIONS

7.1

In this section we present examples of applications of the pre-processing, analysis, and classification techniques described in previous sections to SeaMARC II sonar imagery from a variety of locations. Textural methods based on the GLCM method are used for both segmentation and reconstruction of images. We also present a new semantic/textural classifier that simultaneously significantly increases the spatial resolution attainable with the GLCM texture method and minimizes the probability of creating mixed texels, without violating the sample density requirements.

The first data set under consideration is of a submarine lava flow located 200 km NW of Vancouver Island, near the Queen Charlotte transform fault and the Tuzo Wilson Knolls. The unprocessed SeaMARC II image, and its associated bathymetry, have been shown in Figures 19 and 20. The dark central feature is a lava flow, which is adjacent to a 100-meter mound in the lower right, that may be the vent source of the flow. An acoustic shadow, seen here as a patch of low intensity pixels, occurs on the steep back side of the volcanic mound. Many other smaller mounds occur in the lower portion of the image. The large lava flow apparently crosses the transform fault scarp, which is down-dropped to the upper right portion of the image.

Application of the GLCM textural analysis to the image first requires image subdivision into texels 250 meters on a side, each
containing 294 pixels. Each texel spans 6 pings and 49 pixels, so that there are 40 texels across the 10 km width of the image and N/6 texels along track, where N is the number of pings in the image. In this example the image consists of approximately 420 pings, and thus a total of 2800 texels result from the subdivision. The pixels in each texel were re-quantized to 16 intensity levels by dividing the original intensity values by 16 and rounding to the nearest integer. Statistical texture signatures consisting of the means and ranges for the features ENT, ASM, CON, COR, ISO, and AISM evaluated at look directions of 0, 45, 90, and 135 degrees and lag of length 1 were calculated for each of the texels. As no bottom samples were available to us, the 2800 six-dimensional texture vectors were classified in the un-supervised mode with our modified K-means clustering algorithm.

Clustering of these 2800 six-dimensional texture vectors into the six classes resulted in the thematic map shown in Figure 40. Although the general outline of the lava flow is delineated by the black texels, there is confusion where the ship's track crosses the lava flow. The original image (Figure 19) displays this region of confusion as a portion of the flow exhibiting anomalously low reflectivity, relative to the rest of the flow. As we assume the lava flow to be continuous beneath the ship's track, either instrumentation problems or scattering variations cause the flow to appear anomalously weak in reflective character in the vicinity of the track. The flow is probably rough compared to the wavelength of the ensonifying sound (12 cm), and hence is behaving in accordance with the scattering law for rough surfaces
Figure 40. Six-class texture classification of the image of the lava flow (Figure 19) prior to processing. Note the confusion in the classification of the flow, particularly near the track of the ship.
(Figure 2b), which predicts less backscatter from rough surfaces at near nadir angles than from smooth surfaces at similar angles. As the system AVG is designed to correct for backscatter variations similar to those from smooth bottoms (Figure 2a), the intensities in the near-nadir angles are over-compensated, yielding the observed lower intensities. Correction of this variation in acoustic character should, therefore, increase the probability of correct classification.

The outer half of the left side of the thematic map also shows variability in the classification of what we have interpreted as sediments. The original image (Figure 19) indicates that the areas mapped by the horizontal striped pattern coincide with a bottom of sedimentary character, while the stippled and diagonally patterned texels represent portions of the image that exhibit anomalously low reflectivity. Inspection of the pixels within these texels indicate that their average is close to zero, indicative of a very sparse presence of reflectors within an essentially non-scattering field. There are four possible causes for this change in acoustic character: (1) an actual change in lithology; (2) a change in the bottom slope such that the area of low reflectivity is sloping away from the towfish; (3) variations in the beam pattern; and (4) angle-variant scattering behavior. The first is possible, but as the bottom in this area is continually being blanketed by distal turbidites, it seems unlikely. Although the bathymetric map (Figure 20) is noisy in the region corresponding to this area of low backscattered intensity, inspection of bathymetry from an adjacent track indicates no bathymetric trend which would cause diminution of reflectivity. Thus, a combination of a faulty
beam pattern, resulting in diminished ensonification intensity, and a bottom which is sufficiently smooth as to reflect only as a forward scatterer past some critical angle of incidence best explain the zones of decreased reflectivity.

The processed image, shown in Figure 41, has been subjected to a refined version of the background subtraction routine discussed in the section on pre-processing. The image was first operated on with the Sobel edge extraction routine. The result of this is the edge map shown in Figure 21. This edge map was then used to delimit the region of operation of the background subtraction algorithm. This procedure prevented contamination of the filter for the region interpreted as a lava flow by the scattering properties exhibited by the surrounding sediments. The low average intensity of the lava flow within 1 km of the ship track is assumed to be due to the phenomenon of angle-dependent scattering, as stated previously. The average intensity as a function of angle for the area inside the edge map constraining the lava flow was calculated following the method described in the section on background subtraction. Figure 42 shows this function as a plot of average intensity verses angle of incidence. Unfortunately, this plot is not truly representative of the backscatter characteristics of the lava flow, in as much as the system beam pattern, TVG, and AVG are not yet accounted for. The plot is, however, a reasonable indicator of the departure of the gains and beam pattern from that which would result in a uniform appearance across-track for a uniform bottom of this type. We correct for this departure from ideal performance by inverting the profile with respect to the average intensity and applying the
Figure 41. Lava flow image as seen in Figure 19, subjected to the enhancement routines described in the text. Note specifically the diminution of anomalously low intensity pixels in the lava flow near the ship track, and the reconstruction of image character in the leftmost quarter of the image, as compared to Figure 19.
Figure 42. Average backscatter intensity as a function of angle for the lava flow shown in Figure 19.
parameters over the range of angles which the profile spans to the pixels in the flow area.

The low amplitude problem associated with the sediments requires a different approach. As most of the pixels in the aberrant area are of I.V. zero, no linear gain transformation will produce a reasonable recreation of the correct image character. Therefore, the bottom in the low intensity area is equated with the sedimentary bottom type mapped successfully elsewhere in the image, but that because of its extremely smooth nature and because of a possible beam pattern problem it yields no significant returns. This supposition is substantiated by the presence in the image of several dark outcrops in the anomalous area. These outcrops are interpreted as basalt, and are imaged because of their greater roughness. Furthermore, acoustic shadows, on the backside of topographic features, are classified similarly as the regions of low intensity. If our assumption about the nature of the bottom is correct, we can reasonably re-create the image in the low intensity area on a textural basis. We calculated a gray-level co-occurrence matrix for a representative area of the sedimented bottom of normal reflectivity. This matrix, which describes the frequency of co-occurrence of intensity values, was used as a rule for the reconstruction of the image in the anomalous classes.

Textural analysis of the corrected image (Figure 41) and subsequent classification, produce the thematic map shown in Figure 43. Without ground truth we cannot be certain whether Figure 40 or Figure 43 is the superior classification. Our geologic interpretation of the region,
Figure 43. Six-class texture classification of the enhanced image of the lava flow as shown in Figure 41. Though the absence of ground truth mitigates any conclusions, the continuity of the flow (represented by the darkest shade) across the track is consistent with our intuition of the flow's distribution.
however, is satisfied by the processed image, and the texture map in Figure 43 in which the lava flow is mapped as being continuous across the ship’s track.

7.2

The next application demonstrates the potential of textural analysis for discrimination of more subtle differences (Figure 44). The shapes and intensities in the image suggest that all of the dark outcrop is a single, continuous reflective unit, such as a lava flow, and the light portions of the image are sediments. However, the image is in reality a composite, consisting of an image of a dolomite outcrop from the upper slope basin of the Peru continental margin near Lima, spliced onto the image of the lava flow previously shown in Figure 41. The point of this unlikely juxtaposition is to demonstrate the difficulty in interpretation of lithologies solely upon the basis of intensity. Careful comparison of the two halves of Figure 44 reveals differences in the character of the two lithologies. The image of the dolomite seems less "crisp" than the basalt, perhaps because of fewer high frequency gray level transitions. However, "less crisp" is subjective, and insufficient for image classification purposes. Textural analysis quantifies and verifies this subjective interpretation.

Figure 45 shows a six-class thematic map based upon textural analysis of the dolomite and lava composite. Note that not only are the basalt and dolomite distinguished, but the hemipelagic sediments surrounding the dolomite are differentiated from the distal turbidites
Figure 44. Composite image consisting of a side-scan image of a dolomite outcrop from the Lima basin juxtaposed to half of the lava flow seen previously in Figure 19. See text for explanation.
Figure 45. Six-class textural classification of the composite image shown in Figure 44. Note the complete differentiation of both basalt from dolomite and hemipelagic sediments surrounding the dolomite from the distal turbidites that surround the basalt.
surrounding the basalt. The fact that all surfaces in the image containing the dolomite are distinguished from all surface is the lava flow image might lead one to suspect that the reasons for the separation are systemic rather than lithologic. However, both images are port-side data, collected with a 1 msec pulse and a 1kHz bandwidth, in water depths similar to within 100 meters. If the textural differences which allow this differentiation of lithologies represent actual intrinsic properties of bottom roughness specific to certain rock and sediment types, then recognition of similar lithologies in other images of the seafloor may be possible on the basis of textural analysis, with only limited recourse to bottom sampling.

7.3

To this point we have shown that textural analysis allows segmentation of images into classes which appear to represent viable geologic units. We would like to determine if these textural signatures are consistently representative of any intrinsic surface characteristics, such as roughness or wavelength. To this end we attempted a comparison of textural analysis of SeaMARC II imagery with estimates of bottom roughness and wavelength. The ground truth data was acquired as part of a program to determine a possible route between the island of Hawaii and Maui for an underwater power cable. The data consist of high precision (+6 cm) bathymetric profiles, with samples spaced approximately 1 meter apart. Of the bathymetric profiles collected, several are aligned perpendicular to the fish tracks of co-
registered SeaMARC II data, and hence were coincident with the high resolution (5 m) direction of the image data.

Attempts at correlation of texture statistics with bottom roughness statistics were characterized by a puzzling lack of success. A probable cause for this failure was encountered during an attempt to correlate texture statistics extracted from SeaMARC data with power spectral estimates from similar sections of SeaMARC data. The six pings of the SeaMARC data over which texture analyses had been conducted were averaged to produce an average profile for power spectral analysis. This average profile was subdivided into twenty segments, each consisting of 49 values, corresponding to the texel subdivision of the same data. Each of the twenty sections was de-trended and subjected to spectral analysis following the method of Jenkins and Watts [1968]. In the resulting power spectra there appeared a peak, of power equal to or greater than any other, which migrated from short to long wavelengths as the cross-track position of the sample shifted from near nadir to grazing incidence. The consistency and strength of this phenomenon throughout all of the data from this survey, as well as from the following survey indicated a system source.

A clue to the source of the noise was provided by consideration of the variation in frequency of the pixel sample generator. Because the speed at which the out-going signal sweeps the bottom varies from nadir to grazing incidence, the rate at which the signal is sampled must also vary. The sampling frequency used by the pixel sample generator is a function of nadir depth, and may be calculated for any cross-track
position. We calculated the approximate sampling frequency for the cross-track position at which each segment of the averaged series was located and determined the frequency of the noise peak for that segment by ratioing the nyquist of the power spectra to half the sampling frequency (i.e. the nyquist sampling frequency at that point and time). The resulting frequencies spanned a range of 30 to 80 Hz, centered about 60 Hz., possibly indicating a leakage of ship's power into the fish electronics at some point prior to the pixel sample generator.

Irrespective of the source, a noise signal of power equivalent to any other true signal, which spanned wavelengths from nyquist to 100 meters could not be removed from the side-scan data in any way which would leave a useful signal for further analysis.

7.4

Despite the absence of a relationship between textural signature and intrinsic surface character, texture-analysis thematic maps can be produced and the thematic units assigned to lithologies. Figure 46a shows a portion of a mosaic which covers a part of the continental slope offshore of Lima, Peru, referred to as the Lima Basin [Hussong et al., 1981]. The lighter portions of the image represent un-lithified, hemipelagic sediments. The darker areas represent lithified Neogene sediments, mostly micrites and dolomicrites of varying degrees of lithification and brecciation (Kulm, L.D., personal communication, 1986). The dolomite which was compared to the lava flow (Figure 44) is located at the northwest end of the oval sediment pond. The parallel
Figure 46a. SeaMARC II side-scan mosaic of Lima Basin. Black line shows position of seismic line CDP-1 with respect to the image.

Figure 46b. Three-class thematic map of mosaic shown in Figure 46a, generated via texture analysis as described in the text.
and sub-parallel lineations striking N45W are also outcrops of bedded micrites. Ground truth for the area comes from dredging (Kulm, L., personal communication, 1986). Unfortunately, as the length of seafloor covered by most dredges was long in relation to the variation of outcrop type visible in the side-scan image, and as the contents of most dredges were lithologically heterogeneous, most outcrops could not be correlated to specific samples. However, dredge MW85-11 sampled the dark outcrop referred to above in the comparison of the lava flow and dolomite. The dredge yielded only dolomicrites, of generally a strongly brecciated character. Due to the "mixed bag" nature of the other dredges, we were limited to two definitive sample types -- cored hemipelagic sediments, and the dredged dolomicrite.

Textural signatures were calculated as described in section 7.1 for the areas of the image coincident with the bottom sample locations. Mean texture vectors consisting of the six statistics CON, ASM, ENT, ENR, COR, and ISO, and their associated standard deviations, were calculated from the texels in the vicinity of the samples. The entire mosaic was then subdivided into 250 m² texels and subjected to textural analysis 7.1. The texels were classified in the supervised mode with the minimum-distance classifier (section 5.3). Those texels whose texture vectors were distinct from the mean vectors representing either dolomicrite or hemipelagic sediments by more than twice the standard deviation of either class were lumped in a third category -- anything else. The three categories of sediments, dolomicrites, and anything else were assigned the colors of green, magenta, and yellow.
respectively, and their texels colored accordingly. The result is shown in Figure 46b. The oval basin containing the hemipelagic sediments is clearly outlined, as is the outcrop of dolomicrite at its NW tip. Magenta texels indicated several other possible outcrops of dolomicrite, which might not have been either detected or discriminated from other dark outcrops on a visual basis.

Although the distribution of the magenta texels about the green oval gives the impression of a synclinal or anticlinal formation, multi-channel seismic data collected in this area argue against this interpretation (Figure 47, after Hussong and Wipperman, 1981). The inflection point on the depth section (arrowed) where the strata change from landward dipping to seaward dipping, corresponds approximately with the eastern edge of the oval sediment pond seen in the side-scan mosaic in Figure 46a. Consequently, the dark outcrops west of the pond which are mapped by magenta texels are the tops of landward dipping strata, stratigraphically lower than the dark outcrops similarly mapped to the east of the pond. The fact that texture analysis has mapped stratigraphically distinct outcrops as similar does not necessarily mandate that they be the same formation, but rather that they possess similar acoustic properties, in particular, roughness or induration.

7.5 Region Growing -- ReGATA

The texel, the basic unit upon which the analysis is based, is assumed to contain a subset of the image data which possesses only one homogeneous texture. The probability of spatial homogeneity of a
Figure 47. Portion of CDP-1, a 24-channel common depth point seismic line gathered by Seiscom Delta, Inc., as part of the Nazca Plate Project [Hussong and Wipperman, 1981].
randomly placed rectangle increases as the texel size decreases. For the gray level co-occurrence matrix to be a reasonable estimator of the image joint-probability distribution, the texel should contain at least $N_G^2$ pixels, where $N_G$ is the number of gray levels to which the image is quantized (section 4.9). The texel therefore can neither be shrunk below a minimum size without compromising the textural analysis nor increased drastically without encountering an unacceptable number of mixels. As a compromise between spatial resolution controlled by texel size, and resolution of textural features as controlled by the number of intervals to which the data are quantized, we have employed a texel measuring approximately 250 m on a side, containing 294 pixels quantized to 16 gray levels. As the gray-scale recorders used for displaying the image data are capable of producing only 16 gray levels, of which human observers are rarely able to discern more than 8, texture analysis at this level is at or slightly above what is discriminable to the eye.

In order to increase both the textural and spatial resolution, the author has designed an image segmentation routine which incorporates semantics and image data. Rather than divide the image into a priori boxes which may or may not contain homogeneous patterns, we seek, via data-controlled decisions, a subdivision of the image into closed regions of limited spatial heterogeneity. One feature vector will then be used to represent each closed region. By allowing regions to grow to their natural boundaries, the probability of producing "mixels" is strongly reduced. As the regions will in general be larger than the
previously used texels, classification time will be reduced concomitant with the reduction in number of feature vectors.

Image segmentation routines incorporate one of two strategies: the top-down or the bottom-up approach. Human vision is based upon the top-down approach. When we view a scene, we naturally segment it into object and background. Each object becomes the next background until the finest resolution desired is acquired. As I view the computer terminal at which I am writing this, my top-down vision system might parse it: CRT - SCREEN - PAGE - PARAGRAPH - SENTENCE - WORD - LETTER. There would be neither need nor ability to view the scene from a bottom-up perspective starting with the photo-elements which compose the letters and cursor.

On the other hand, the computer is better equipped to deal with details than Gestalt, hence the bottom-up approach is more appropriate [Brice and Fenema, 1970]. A bottom-up, region-growing routine begins with a cell definition. This cell, either a unit pixel or small group of pixels, defines the limiting resolution of the region to be grown. Each cell has associated with it some feature, calculated from the image data coincident with that cell. Region growing requires the comparison, based on some statistical test, of the feature for that cell with the features of immediately adjacent regions or cells. If the comparison test passes, the two cells or the region and the cell are annexed. If the comparison test fails, the cell is declared a new region, and the algorithm continues to the next cell. The statistic most often computed for comparison purposes is the average value of the measurements
associated with the pixels in the cells. For image data, the logic of the classification is that if two cells are neighbors and possess similar average value, they probably represent a uniform, connected surface in the object plane, and hence should be similarly classified.

Implementation of the bottom-up, region-growing routine is as follows. SeaMARC II data are stored in records, line-interleaved-by-pixel. Two records, containing two pings of 984 port and 984 starboard pixels, are "buffered in". The data for port and starboard are subdivided into cells containing two lines of 16 pixels each. The mean intensity and variance are calculated for each of these 32-pixel cells. Two tests are then conducted. The first determines if the cell is reasonably homogeneous, in order to separate the cells into the categories of "region" and "boundary". Those cells which do not pass this test of limited heterogeneity are assumed to span two regions, and hence are labelled as boundary cells. An option in the program allows these cells to be either flagged as non-region cells, or quartered and annexed to the most similar adjacent region. Kettig and Landgrebe [1976] propose a test which we utilize here, consisting of the ratio of the cell standard deviation to the cell mean. If this statistic falls above some heuristic threshold, the cell is labeled as a boundary. As sonar images are inherently noisy, this threshold should be fairly high, to avoid the classification of the entire image as boundaries.

The second test is the basis of annexation and region growing. The author's method is as follows. For purposes of illustration, the top of the image under consideration is labeled north. Cell classification
and region growing proceeds west to east, and lines are "buffered in" north to south. Each classified cell has an associated flag which indicates its status as either a boundary or region cell. If it is a boundary cell, there is no need to compare it to the cell under consideration. If it is a region cell, it has associated with it the mean and variance of the region of which it is a member. This region can consist of one or more cells. The cell under consideration is compared to the region cells immediately adjacent in the west, northwest, north, and northeast directions. Comparison in these four directions minimizes imposition of boundary direction preferences within the limitations of the cell tessellation. The comparison can be on the basis of either a parametric Student's t-test and F statistic, or the non-parametric minimally distant means, at the operator's discretion. As region growing requirements must be reasonably lax to prevent classifications composed entirely of boundary cells, it is imperative that the cell be annexed to the most similar region which passes the second test, and not just the first region to so do.

As a region grows, its mean and variance are re-computed, and the pixels within the annexed cell augment the four gray level co-occurrence matrices associated with the region. After the last cell from the two lines buffered has been classified, all regions are checked for growth in the last iteration. If they have not grown, they are considered closed, and their associated gray level co-occurrence matrices are submitted to a subroutine which extracts the standard six-dimensional texture feature vector. The texture vector and all the pixels in the
region from which it was extracted are then assigned a region number and "buffered out" to a storage file.

When the routine terminates at the end of the data file, all remaining open regions are closed by the routine and analysed for texture. The resulting list of region-numbered, feature vectors are submitted to an un-supervised cluster analysis. The region numbers map the regions to the feature vectors in the clusters, and all pixels in all regions with similarly clustered feature vectors are mapped similarly.

The benefits of this routine are three-fold: computational speed, increased spatial resolution, and increased textural fidelity. We stated above without proof that there would be fewer regions resultant from this method than texels from the arbitrary division of the image. The intuitive proof for this lies in the definition of texel as the largest tesselation unlikely to transect image region boundaries. If the regions are allowed to grow to their natural boundaries, which the texels are designed to avoid intersection of, then the grown regions will be larger and fewer in number than those resulting from an arbitrary texel subdivision of the image. Fewer regions mean fewer calls to the feature extraction subroutine, and fewer vectors to classify. Furthermore, the algorithm requires but one pass through the data to produce a segmentation with texture vectors, and requires only 4 lines of data in memory at any one time.
The cell dimension of 2 lines by 16 pixels results in a limiting resolution of approximately 80 meters. This cell size was chosen because it allows quartering of the cells in the event of a boundary classification. The number of pixels, 32, also straddles the small sampling theory/large sampling theory boundary, so that parametric and non-parametric tests are approximately equally valid.

Finally, growing regions of limited heterogeneity minimizes the inter-contamination of texture signatures by each other, or by boundaries, neither of which are likely to result in a representative signature. Gray level co-occurrence matrices from larger regions will contain more entries, and hence yield better estimates of image texture. By minimizing mixing, and increasing estimate fidelity, the clusters formed in the un-supervised classification step should be tighter, resulting in better discriminant boundaries in the classification n-space, and less confusion in the image plane.

FORTRAN code and documentation for our routine ReGATA - Region Growing And Texture Analysis - can be found in appendix G1. Clustering and classification algorithms designed around the format of the data produced by ReGATA are given in appendix G2. Lastly, a routine for transforming the clustered regions into side-scan format imagery is presented in appendix G3.
7.6 Applications of ReGATA

One of the other benefits of the region growing method is that as the regions that result are in general larger than the previously used texels, the image need not be re-quantized to such a small number of gray levels. Specifically, regions often grow to contain hundreds to thousands of cells, each containing 32 pixels. If we analyze all closed regions for texture, but only use those with more than 100 cells for determination of the cluster centroids, we can still classify the entire image, but with a four-fold increase in textural sensitivity.

In our first application of ReGATA, the original image (Figure 19) was re-quantized to 64 gray levels, and subdivided into cells containing 32 pixels, for which the average intensities and standard deviations were calculated. The non-parametric, minimally-distant means rule was used for region growing, with annexation occurring if the regions differed in their means by three levels or less. All closed regions containing five or more cells were analysed for texture and hence had associated with them the six-dimensional feature vector. In the cluster analysis which followed, only those regions containing 100 or more cells contributed to the determination of the class centroids. Those regions containing fewer than 100 cells were then placed into the class from which their feature vector was minimally distant.

Figure 48 shows the result of this region growing with three classes prescribed. The three resulting classes -- green, purple, and blue -- correspond in the image to regions containing the lava flow and
Figure 48. Three-class ReGATA map of lava flow image shown in Figure 19. Blank areas correspond to regions with fewer than five annexed cells. See text for details.
two classes of sediment respectively. The purple appears to correspond to the less reflective, possibly smoother sediment. The areas mapped as blue correspond to visually rougher sediments in the vicinity of the lava flow. In the original side-scan image, these sediments appear rougher, as if they were either receiving debris from the neighboring lava flow, or expressing the roughness of some thinly buried flow. The blank areas correspond to either boundary cells or to un-annexed regions containing less than five cells.

The chief advantages of the three-class thematic map shown in Figure 48 over the six-class map in Figure 43 are the increased spatial resolution of features such as the crenulations in the perimeter of the flow and the distribution and shape of sedimented fenestras in the flow, plus the resolution in the three-class map of two categories of sediment. Computer time required for analysis and classification is also less for the ReGATA map.

Figure 49 presents an interesting result of forcing the clustering routine to find four classes in the regions resulting from ReGATA analysis of the lava flow image. The distribution of the two sediment classes - the blue and purple - is identical to that for the three-class map. The lava flow, previously mapped as green, has been split into two classes, the green and brown. The distribution of these two classes is distinctly non-random; green dominates the port side and brown the starboard side of the image.
Figure 49. Four-class ReGATA map of lava flow image shown in Figure 19. Note strong separation of port and starboard halves of the lava flow.
This mapping could represent actual variation in the roughness or lithology of the flow, indicating perhaps a change from pahoehoe to aa. Considering, however, the strongly bi-modal spatial distribution of the two classes about the ship's track, the discrimination between the two classes may result from the frequency difference between the port and starboard transducers. The port transducers operate at 11 kHz and the starboard at 12.5 kHz, therefore, in the water they will produce signals of 13.6 and 12 cm wavelengths, respectively. Much as a sub-aerial terrain will appear different when viewed with k-band (0.86 cm) and L-band (3 cm) radars, it is possible that this 1.6 cm difference in wavelength between the port and starboard transducers is sufficient to cause a surface of uniform roughness to appear different to the port and starboard sides.

Table 3 quantifies the distinction between the four classes seen in Figure 49. The four features which displayed single-valued relationships with wavelength in section 6.2 - CDN, ASM, ENT, and ISO - are enumerated for each of the four classes. Comparison of the values for the four class feature vectors with the plots of texture feature verses wavelength in Figure 26 indicates an increase in relative wavelength from class 1 through 4. As classes 1 and 2 correspond to the lava flow and classes 3 and 4 to sediments, this mapping conforms with our expectation that at the scale of features visible to SeaMARC II the sediment signature would be dominated by long-wavelength variations and the signature from the lava flow by highly variable specular returns from corner reflectors. Class 1, mapped in brown in Figure 49, is characterized by a shorter wavelength response than class 2. If we
Table 3. Feature vectors for the four classes shown in the thematic map in Figure 49. See text for details.

<table>
<thead>
<tr>
<th>CLASS 1</th>
<th>CLASS 2</th>
<th>CLASS 3</th>
<th>CLASS 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>CON</td>
<td>ASM</td>
<td>ENT</td>
<td>ISO</td>
</tr>
<tr>
<td>2.3403</td>
<td>0.0265</td>
<td>0.0045</td>
<td>1.2000</td>
</tr>
<tr>
<td>2.1491</td>
<td>0.0356</td>
<td>0.0052</td>
<td>1.1016</td>
</tr>
<tr>
<td>1.8953</td>
<td>0.0555</td>
<td>0.0067</td>
<td>0.9691</td>
</tr>
<tr>
<td>1.5909</td>
<td>0.1179</td>
<td>0.0158</td>
<td>0.7976</td>
</tr>
</tbody>
</table>
assume that the flow is of uniform roughness and wavelength (at least with respect to its distribution about the track), then the port side, ensonified with the longer wavelength signal, should appear smoother and yield a longer wavelength signature than the starboard side, which it does!

As there is still the possibility that the distribution of classes 1 and 2 about the track is but fortuitous, due to anisotropic roughness of the flow or system differences not related to frequency, we can not make any definite conclusion about this finding. Should however, the flow prove to be uniform about the track, this would indicate the possibility of characterization of surface roughness by overlapping reversed tracks.

7.7 ReGATA Classification of Lima Basin

ReGATA was used to produce and interpret a 12-class thematic map of the Lima Basin mosaic (Figure 50). In the absence of sufficient bottom samples, a combination supervised/un-supervised classification scheme was followed wherein 12 class centroids were first established via clustering, and all regions grown by ReGATA were classified as one of the 12 classes from which their feature vectors were minimally distant.

The 12 class centroids were established by selecting 8 side-scan files from the original mosaic which appeared as a group to contain all surface types visible in the original mosaic. These 8 files were analysed for texture and clustered into 16 classes. Inspection by eye
Figure 50. Lima Basin side-scan mosaic, with salient features demarked.
of the results of this classification in comparison with the original 8 side-scan images allowed 4 redundant classes to be discarded or lumped. Re-classification of the 8 side-scan files into 12 classes in the clustering routine resulted in the 12 class centroid vectors used to span the variations of texture within the mosaic.

Once the 12 class centroid-vectors were established, the entire mosaic was subjected to ReGATA analysis. In order to avoid obfuscation of general trends by noisy detail, the pixel intensity values were requantized to 16 intervals, and the region annexation threshold was set at 2, so that cells were merged only if their average intensity after re-quantization differed by two or less. As a difference threshold of 2 implies a difference of up to 32 in the original 8-bit image data, any ability to differentiate all but the most obvious features would at first seem unlikely. However, the human visual system's response to gray-scale variations is logarithmic, flattening toward darker shades [Gregory, 1966]. Thus, setting the threshold at 2 mutes the contributions due to long-wavelength background variations and high frequency low amplitude noise, while expanding the detail in the darker portions of the mosaic wherein the eye is less perceptive. The results of the classification are shown in Figure 51.

7.8 Qualitative analysis of the classification

Several salient features are interpretable from comparison of the 12-color distribution of the classes mapped by the ReGATA routine for the Lima Basin with the original side-scan mosaic (Figures 50 and 51).
Figure 51. 12-class thematic map of the side-scan mosaic of Lima Basin, generated with ReGATA. See text for details.
The most obvious feature is the northwest trending oval, which represents the sediment pond. The blue and orange colors map the distribution of the un-consolidated hemipelagic sediments within and outside the pond. Despite the contrast in hue, the two classes represent basically the same material. The areas mapped in orange may represent either a thin layer underneath the sediments mapped in blue, or sediments of a rougher surficial nature, as they seem to correlate with sediment surfaces of a more reflective nature in the side-scan mosaic. Interdigitated boundaries between the orange and the blue, such as in the southeast corner of the map, may represent subtle variations in sediment roughness not visible in the original image. The abrupt, diagonal boundaries between blue and orange in the northwest portion of the map follow the structural grain of the region, and may also indicate of some difference in roughness imparted by the basement. The marked discontinuity between the mapping of orange and blue between the first and second tracks in the northeast corner may be indicative of some surficial sediment anisotropy, such that different textures are observed from different look directions.

The remainder of the colors we interpret to represent consolidated rock outcrops, or outcrops slightly covered with sediments. Interpretations of the finer details visible in the mosaic are made from the detailed figures which follow.

Figure 52a shows a blow-up of the central portion of the Lima Basin side-scan mosaic; the corresponding texture map is shown in Figure 52b. The feature of interest is a dark scarp, which begins at a black
Figure 52a. Detail of side-scan mosaic showing knick point and scarp.

Figure 52b. Detail of 12-class ReGATA map, corresponding to area shown in Figure 52a.
triangular feature in the northwest corner of Figure 52a and strikes southeast across the figure. The triangular feature is a knick point, resulting from transection of the scarp by the erosional gully which strikes southwest from it. Erosion of the scarp by the debris in the gully has created this triangular window into the sediments. Note how the knick point is represented in Figure 52b by three nesting triangles of color, specifically magenta, light green and dark green from its perimeter to its center. We interpret this change in textures to be indicative of changes in lithology or erosion-induced surface roughness, and base the stratigraphic interpretation which follows upon this erosional window.

We now turn our attention to the scarp, of which the knick point reveals a cross-section. In Figure 52b the scarp strikes southeast from the knick point, in a color sequence of yellow, magenta, light green with interspersed cyan, and dark green. The yellow regions represent either an erosional/stratal layer superadjacent to that which the magenta represents, or the magenta surface covered with variable amounts of sediment. The reproduction of the color sequence of magenta, light green, and dark green echos that seen in the knick point. Whether these three colors actually represent distinct strata, or differing responses to erosion, cannot be determined. Nonetheless, if we assume either a stratigraphic or erosional relationship, stratigraphic inferences can still be drawn from their respective areal distributions. A potential stratigraphic sequence from top to bottom at this point would be: dark blue-orange-yellow-magenta-light green-cyan-dark green-violet. Repetitions of portions of this color sequence can be found throughout
Figure 51, such as the magenta and green erosion gullies in a background of yellow to the northeast, or the heart-shaped magenta and green knick point in a background of blue near the center of the southern edge of the image. Small violet patches may be found exclusively in the dark green fields in Figure 52b and the entire thematic map, and are thought to represent a continuation of this erosional/stratal series.

A portion of the side-scan image containing the dredged dolomite outcrop, and the corresponding texture map are shown in Figures 53a and 53b. The outcrop is mapped by the magenta, light green, and yellow field in the northeast corner of Figure 53b. The bi- to tri-modal nature of the mapping of the outcrop is not inconsistent with our sampling, in as much as the dredge yielded both brecciated and non-brecciated clasts of dolomicrite.

Figure 53b also introduces two new major classes and a two minor ones. The major classes are represented by the pale-green and chocolate brown colors, predominantly distributed about the center of the figure. Comparison of the distribution of the two colors in reference to the side-scan image (Figure 53a) indicates that the pale-green correlates with the narrow, dark outcrop at the western edge of the sediment pond, while the brown correlates with the border between the dark outcrop and the sediment pond. We therefore interpret the areas mapped in pale green as outcrop, and those mapped in brown as the same surface as that mapped in green, but partially covered by sediment.
Figure 53a. Detail of side-scan mosaic showing the dolomite compared previously to the lava flow (Figure 44).

Figure 53b. Detail of 12-class ReGATA map, corresponding to area in Figure 53a.
The two minor colors, which make up the remainder of the 12 classes, are aquamarine and forest green, the best representations of which are seen as small patches in the pale-green pattern at the west edge of the sediment pond. Their areal extent is minor and correlates with dark patches in the original side-scan, but their lithologic nature is unknown.

A major submarine channel dissects the sediment pond (Figure 54a). The texture map shows the channel, outlined in orange and yellow, incised into the blue of the sediment pond, in accordance with our interpretation of the orange as either a subadjacent layer or material similar to the blue but roughened or thinned by erosion. Also visible in the map are more windows of the violet in the dark green, and the light green in the magenta.

Finally, an area of the side-scan image (Figure 55a) contains several dark, circular features previously interpreted as mud volcanoes [Hussong et al., 1985]. Although the roughly circular features seen in the side-scan appear similar, there is little similar mapping in the texture map (Figure 55b). Instead, the dark features are mapped as magenta and yellow, magenta, yellow and brown, and magenta, yellow, and green. The recurrent factor is the magenta, which appears to be mapping the outcrops which follow the northeasterly structural grain and the erosional south-southeasterly trend. The intersection of these two nearly-perpendicular lineaments may have produced, in conjunction with some spherical weathering, circular structures interpreted heretofore as flat-topped conical mud volcanoes. Even if this intersecting-lineament
Figure 54a. Detail of side-scan mosaic showing drainage channel.

Figure 54b. Detail of 12-class ReGATA corresponding to area shown in Figure 54b.
Figure 55a. Detail of side-scan mosaic showing features interpreted as mud volcanoes.

Figure 55b. Detail of 12-class ReGATA map corresponding to area shown in Figure 55a.
interpretation is incorrect, it is still unlikely that the effluent from any argilokinesis would have a surface roughness which would produce a texture similar to that of a strongly brecciated dolomicrite.

7.8 Quantitative Interpretation

Thus far our interpretation has been entirely qualitative. Can we determine anything about the intrinsic nature of the surfaces which the classes represent from analysis of their feature vectors? Figure 56 shows plots of the values of the first two texture features, CON and ASM, for the 12 classes. The range of values for the textural parameters are plotted along the y-axis. The scale along the x-axis is arbitrary, although by analogy with the theoretical studies presented in section 6, we assume it is in units of wavelength, with dimension increasing to the right. Until we have a better idea of the contributions of phenomena other than wavelength to the absolute value of a texture signature, it would be unwise to attempt to assign an absolute value to the wavelength of the surface which any class might represent. We can, however, make relative comparisons.

In these comparisons we refer back to Figure 26 which shows plots of textural parameter values verses wavelength for synthetically generated images. In this figure, the first feature, CON, is a decreasing function of wavelength, and the second feature, ASM, is increasing. For the texture features from the 12-class map, the three classes interpreted as sediments or sediments over outcrops -- yellow, orange, and slate blue -- fall at the long wavelength end of the series.
Figure 56. Plots of textural parameter versus class for the features CON and ASM, for the 12 class vectors used in the 12-class ReGATA map. Scale of the x-axis is arbitrary. See text for explanation.
Furthermore, surfaces represented by the orange color are interpreted as rougher than surfaces represented by the blue, if we allow some relation between roughness and wavelength.

Decreasing wavelengths (or increasing roughnesses) are predicted for the series magenta-cyan-light green-dark green-violet. This trend was a portion of the "stratigraphic" series we interpreted from considerations of knick points and other erosional relations. It seems unlikely that a random stratigraphic series, composed largely of micrites, would increase monotonically in roughness with depth. Rather it seems more likely that the predicted increase in roughness is more related to increasing effects of erosion. Furthermore, as a continuous series of the five colors is not seen in any one erosional feature, we propose that the series mentioned above may not be continuous, but rather composed of two lithologies, perhaps the magenta and the dark green, with their associated erosion-roughened surfaces mapped in light green and blue, and violet, respectively.

Finally, consideration of the textural values for the pale green and chocolate-brown classes sheds some light on their distribution in the thematic map. For the first texture feature, CON, the two classes are seen to be identical, falling into the intermediate wavelength prediction, between the the cluster of sediment classes and those interpreted as outcrops. For the second feature, however, the two are quite disparate, with the pale green falling to intermediate wavelengths as before but the chocolate falling into the long wavelength, sedimentary prediction. This bi-modal behavior on the part of the
texture vector for the chocolate class is consistent with our interpretation of the areas mapped in brown as representing outcrops partially covered with sediment, and hence less rough than the unsedimented outcrop.

7.9 Speculative Interpretation

Having developed the ReGATA technique and applied it to produce the 12-class texture map of the Lima Basin (Figure 51), the following questions arise: Do we believe the mappings produced by the ReGATA routine? Can we tie these mappings to the geology?

Consideration of the texture map (Figure 51) reveals strong spatial coherency. Textural units appear to be equally continuous along and across-track, indicating that variation of textural signature with change in angle of incidence is not significant. Our faith in the validity of the mapping is further bolstered by the obvious agreement between the majority of the tracks. As similar mappings are produced from opposite look-directions and different incidence angles, the ReGATA technique must be identifying in the image data some feature or combination of features of the seafloor intrinsic to the terrain surfaces.

Further consideration of both the side-scan and texture mosaics (Figures 50 and 51) indicates that many of the regions in the texture map coincide with discernable units in the side-scan mosaic. This would seem to contradict our earlier statement that texture was insensitive to
average intensity, as shown in section 6.2. However, correlation between units visible on the basis of intensity, and those mapped on the basis of texture, merely substantiates the work of Julesz [1975], which indicates that regions similar in \( n^{th} \) order statistics are similar in all \( n-1 \) and lower order statistics. However, the opposite is not true; similar first order statistics do not imply similar second-order statistics, as can be seen from comparison of Figures 50 and 51. Thus, many areas of no discernable intensity (first-order) differences are distinguished on the basis of texture (second-order statistics), e.g. variations in the sediment pond, the nesting triangles of texture in the knick-points, etc.

We feel that the consistency of the textural transitions is further evidence that the ReGATA routine is mapping actual bottom features. The consistent repetition of the texture series magenta-light green-dark green in various features such as erosion gullies, knick-points, and scarps substantiates our belief that we are mapping variations in bottom character, and not simply variations in the size and or slope of the feature. Comparison of the textural values for the three classes with that predicted from theoretical studies indicates a predicted decrease in wavelength (increase in roughness?) in the series magenta-light green-dark green. The magenta is consistently associated with the upper or outer edges of the erosional features, and the greens are associated with the lower or inner portions of the features. If we assume roughness increases toward the center of an erosional feature, due to
the accumulation of larger talus further down-slope, then we may postulate a first-order roughness prediction capability.

Finally, a comparison of the texture mapping of the outcrop at the west edge of the sediment pond (Figures 53a and b) with data from multi-channel seismic lines (B. Taylor, 1985, unpublished data) indicates that regions mapped as similar by texture appear to be similar in the seismic data as well. Two seismic lines cross the outcrop (labeled in Figure 53a) in a northeasterly direction. A third seismic line to the northeast provides a cross-tie between the two. The eastern edge of the outcrop, mapped in pale green in the texture map (Figure 53b), coincides with the projected surficial expression of a continuous seismic horizon mapped by the three seismic lines. Noting that the frequency and resolutions differ between the seismic and sonar systems, and that the seismic system is mapping contrasts in acoustic impedance while texture can only be measuring roughness, such a coincident mapping could not be fortuitous, and we believe that the texture patterns are mapping the outcrops.

From the above evidence we conclude that the texture map shown in Figure 51 is a first-order map of surficial character, as expressed through the interaction of 12kHz acoustic energy with the intrinsic or erosion-/tectonically-generated surface roughness.
Quantitative image processing techniques have been applied to a variety of SeaMARC II side-scan sonar seafloor images. Image contrast has been enhanced via histogram manipulations. Simultaneously, undesirable operator-induced gain changes have been removed. Across-track spurious variations in image intensity due to improper system beam pattern and irregular scattering properties of the bottom, and overprinting due to surface reflections, have been corrected by background subtraction. Track-parallel linear features resulting from along-track aliasing of point reflectors have been migrated back to more appropriate dimensions. Total geometric correction based upon explicit knowledge of co-registered simultaneous bathymetric data yielded images which are free of layover distortions, and yield true representations of reflector position and morphology. Median-value filtering has been employed to mitigate image noise in preparation for edge detection.

A theoretical development of a method for analysis and classification of image texture on the basis of estimates of image joint probability distributions is presented. The mappings produced by this method are shown to be independent of gain settings or changes, invariant under change in look-direction, and predictive of relative surface wavelength. This method is incorporated into a new routine for Region Growing and Texture Analysis -- REGATA.
Application of the above techniques to a variety of imagery resulted in both superior images for subjective and computer-aided interpretations. Application of the texture analysis routines demonstrated the ability to distinguish images of basalt and dolomite outcrops, which would not have been discernable on the basis of intensity alone. A possible sensitivity to sub-pixel roughness variations is indicated by textural discrimination of port and starboard images, acquired with a 1 cm difference in ensonifying wavelengths, of presumably uniform terrains. Correlation of limited ground truth with image texture parameters has allowed first-order mapping of lithology distributions.

In a 12-class classification of a half-degree square of the seafloor, a potential stratigraphic relationship was elucidated for eight classes, with erosional and structural features resolved to 80 meters. It is hoped that evaluation of the true lithologies of identified textures via ground truth data will allow future recognition of similar lithologies on the basis of remotely sensed textures, with the need for only limited sampling programs.

We regard these fundamental procedures for image processing and quantitative interpretation of SeaMARC II data to be the first steps in applying remote sensing techniques to seafloor swathmapping data. Although the use of marine acoustic data, which is strongly influenced by its low signal to noise ratio and susceptibility to ray path variations, is inherently more difficult than processing sub-aerial optical or radar images, we are confident that eventual increased
quantification of seafloor data will permit widespread application of various airborne and satellite remote sensing techniques to the imaging and mapping of the seafloor.
APPENDIX A1

PROGRAM HISTEQ, A HISTOGRAM EQUILIZATION ROUTINE

THIS PROGRAM READS THE PIXEL AMPLITUDES FOR BOTH PORT AND
STARBORD AND PUTS THEM INTO TWO SEPARATE HISTOGRAMS.
THESE HISTOGRAMS ARE THEN DIDDLED SO AS TO EFFECT ENTROPY
MAXIMIZATION. THE PIXEL AMPLITUDES ARE THEN ALTERED
ACCORDINGLY, AND IMAGED.

INTEGER PAMP(984), SAMP(984)
INTEGER DATA(994), NEW(994), NEWP(256), NEWS(256)
INTEGER PH(257), SH(256), NPH(256), NSH(256)
INTEGER IN, OUT, PARA, E, NHIST(256), IPAR, FILIN(6), FILOUT(6)
REAL X(257), Y(257), RH(257)
READ (10, FMT="(I5)") IPAR
READ (10, FMT="(I5)") IPAR
DATA DATAFIL/22/
DATA IN, OUT/20, 30/
10 READ (DATAFIL, 21, END=100) FILIN, FILOUT
21 FORMAT(12(A3))
CALL ASSIGN (IN, FILIN, IERR)
IF (IERR .NE. 0) GO TO 100
IF (IGENR(FILOUT, 3, 300, 5, 5000) .GT. 1) STOP GENSSF
CALL ASSIGN (OUT, FILOUT, IERR)
IF (IERR .NE. 0) GO TO 100

BUFFER IN(IN, DATA, B, 24, E, N)
CALL STATUS (IN)
BUFFER OUT (OUT, DATA, B, 24, E, N)
CALL STATUS (OUT)

INITIALIZE HISTOGRAM ARRAYS

FOR I=1, 256
  PH(I) = 0.0
  SH(I) = 0.0
  RH(I) = 0.0
  NHIST(I) = 0.0
END FOR

READ IN DATA

LOOP
  BUFFER IN(IN, DATA, B, 994, E, N)
  CALL STATUS (IN)
  EXIT LOOP IF (E .EQ. 3 .OR. E .EQ. 4)
  FOR I=1, 984
    PAMP(I) = (DATA(I+10) .AND. '377)
    SAMP(I) = (DATA(I+10)/256) .AND. '377
END FOR

CREATE HISTOGRAMS

FOR I=1, 984
  PH(PAMP(I)+1) = PH(PAMP(I)+1) + 1
SH(SAMP(I)+1)=SH(SAMP(I)+1)+1
END FOR
END LOOP
FOR I=1,256
PH(I)=PH(I)+SH(I)
END FOR
CALL EQUAL(PH,NPH,NEWP)
REWIND 20
BUFFER IN(IN,DATA,B,24,E,N)
CALL STATUS(IN)
BUFFER OUT(OUT,DATA,B,24,E,N)
CALL STATUS(OUT)

READ IN DATA

READ (40,-) NFIT,NOFFSET,NPINC,ITYPE,NUMB
K=0
LOOP
BUFFER IN(IN,DATA,B,994,E,N)
CALL STATUS(IN)
IF (E .EQ. 3 .OR. E .EQ. 4)
CLOSE 20
CLOSE 30
GO TO 10
END IF
FOR I=1,984
PAMP(I)=(DATA(I+10) .AND. '377)
SAMP(I)=(DATA(I+10)/256) .AND. '377
END FOR
MODIFY VALUES

FOR I=1,984
WRITE(6, ) PAMP(I),SAMP(I),NPAMP(I),NSAMP(I)
END FOR

FOR I=11,994
NHIST( NEWP(PAMP(I-10)+1))=NHIST(NEWP(PAMP(I-10)+1))+1
NEWP(I)=1
NEWS(I)=1
NEW(I)=NEWP(PAMP(I-10)+1) .AND. '377
NEW(I)=NEW(I) .OR. (NEWP(SAMP(I-10)+1) .AND. '377)*256
END FOR
K=K+1
BUFFER OUT(OUT,NEW,B,994,E,N)
CALL STATUS(OUT)
FORMAT(1H0,4(SX,I10))
END LOOP
FOR I=1,1256
Y(I)=NHIST(I)/FLOAT(K)
X(I)=FLOAT(I)
RH(I)=PH(I)/FLOAT(K)
END FOR
Y(257)=0.0
X(257)=0.0
YH(257)=0.0
IF (IPAR .EQ. 0 )
  CALL JBAXES(X,257,15.0,15HPIXEL INTENSITY,15,RH,257,6.0,16HNUM
  OF PIXELS,16)
  FOR I=1,257
    CALL DRAW LN(X(I),RH(I))
  END FOR
  CALL END PLT
ELSE
  CALL JBAXES(X,257,15.0,15HPIXEL INTENSITY,15,Y,257,6.0,16HNUM
  OF PIXELS,16)
  FOR I=1,257
    CALL DRAW LN(X(I),Y(I))
  END FOR
  CALL END PLT
END IF
STOP
END

C********************************************************************
C SUBROUTINE EQLIZR; A MAXIMUM ENTROPIZATION ROUTINE
C*******************************************************************

SUBROUTINE EQLIZR(H,NEWH,NEW)
INTEGER LEFT(256),RIGHT(256),H(256),NEW(256),Z,R,HAVE,HINT,HSUM
INTEGER NEWH(256),RANGE
HSUM=0
FOR Z=1,256
  NEW(Z)=0
  NEWH(Z)=0
  HSUM=HSUM+H(Z)
END FOR
HAVE = NINT(HSUM/256.)
WRITE(3,?) "AVERAGE OF THE HISTORAM=",HAVE
C HAVE=4

R=0.0
HINT=0.0
FOR Z=1,256
  K=0
  LEFT(Z)=R
  HINT=HINT+H(Z)
  NEW(Z)= INT(HINT/FLOAT(HAVE))
  WHILE (HINT .GT. HAVE)
    HINT=HINT-HAVE
    R=R+1
    K=K+1
  END WHILE
  RIGHT(Z)=R
  NEW(Z)=(LEFT(Z)+RIGHT(Z))/2.
END FOR
C FOR I=1,256
C NEWH(I)=0.0
C END FOR
C FOR I=1,984
C NEWH(NF(I)+1)=NEWH(NF(I)+1)+1
C END FOR

RETURN
END
APPENDIX A2

C THIS IS AVPRM, AN ATTEMPT AT CREATING AN ANGLE DEPENDENT DE-
C PRONGING PARAMETER FILE.
INTEGER DATA(994), PAMP(984), SAMP(984), PSUM( 90), SSUM( 90)
INTEGER E,K1,K2,DATAFIL,SSFIN,FILENAME(6),SPNAME(6)
INTEGER SDFFIL , IERR, PKT( 90),SKT( 90),FLAG,KEEP
INTEGER *6 TOTAL,PTOTAL,STOTAL,P1,P2,S1,S2
INTEGER *3 PTIT(13),STIT(13)
CHARACTER*39 PTITLE,STITLE
REAL ALT, SALT=GALT,S3,P3
EQUIVALENCE (PTITLE,PTIT(1))
EQUIVALENCE (STITLE,STIT(1))
DATA DATAFIL/22/
DATA SSFIN/24/
DATA SDFFIL/25/
C READ(20, FMT="(2I5)") K1,K2
C WRITE(3,) " K1=",K1," K2=",K2
KEEP = 0
FLAG=0
10 KT=0.
READ(DATAFIL,21,END =100) FILENAME, SPNAME
WRITE(3,21) FILENAME, SPNAME
21 FORMAT(12 (A3))
CALL ASSIGN (SSFIN,FILENAME,IERR)
WRITE(3,) " IERR=",IERR
IF (IERR .NE. 0) GO TO 100
OPEN 24
BUFFER IN(24,DATA,B,24,E,N)
CALL STATUS(24)
WRITE(3,) " E="
IF (E .EQ. 3 .OR. E .EQ. 4) GO TO 100
CALL ASSIGN (SDFFIL,SPNAME,IERR)
WRITE(3,) " IERR=",IERR
IF (IERR .NE. 0) GO TO 100
READ(25, FMT="(/ / / / / / / /")
11 CONTINUE
TOTAL = 0, 0
PTOTAL = 0
STOTAL = 0
P1=0
P2=0
S1=0
S2=0
FOR I=1, 90
. . APPAR(I)=1.
. . ASPAR(I)=1.
. . PKT(I)= 0
. . SKT(I)= 0
. . PSUM(I)=0
. . SSUM(I)=0
. . A(I)=FLOAT(I)
END FOR
IF(FLAG .EQ. 0) READ(20,-,END=100) K1,K2
WRITE(3) " K1=" ,K1 ," K2=" ,K2
LOOP
  IF(K1 .LE. KEEP) GO TO 15
  FLAG=0
  BUFFER IN (24,DATA,E,994,E,N)
  CALL STATUS(24)
  IF (E .EQ. 3 .OR. E .EQ. 4)
    IF (E .EQ. 3 .OR. E .EQ. 4)
      close 24
      close 25
      FLAG=1
      KEEP=0
      GO TO 10
    END IF
    READ(25, FMT="(35X,F6.1,21X,2F7.1)",END=10) ALT,GALT,FALT
    IF (FALT.NE.0) THEN
      ALT=FALT
    ELSE IF (GALT.NE.0) THEN
      ALT=GALT
    END IF
  END IF
C WRITE(3, ) ALT
  KT=KT+1.
  IF(KT .GE. Kl .AND. KT .LE. K2)
    FOR I=1,984
      PAMP(I)= (DATA(I+10) .AND. '377)
      SAMP(I)= (DATA(I+10)/256) .AND. '377
      PTOTAL =PTOTAL + PAMP(I)
      STOTAL=STOTAL + SAMP(I)
    END FOR
    K=1
    FOR I = 1,984
      L = 57.62 * ATAN((FLOAT(I)*5.)/ALT)
      WRITE(3,) " L=" ,L ," I=" ,I ," PAMP=" ,PAMP(I) ," PTK=" , PTK(I)
    END IF
    IF (L .GE. 20 .AND. L .LE. 59)
      S1=S1+SAMP(985-I)
      P1=P1+1
    END IF
    IF(L .GE. 30 .AND. L .LE. 49)
      S2=S2+SAMP(985-I)
      P2=P2+1
    END IF
    IF (L .LE. K)
      PSUM(K) = PSUM(K) + PAMP(I)
      SSUM(K) = SSUM(K) + SAMP(985-I)
      SKT(K) = SKT(K)+1
      PTK(K) = PTK(K)+1
      ELSE
      K=K+1
      PSUM(K) = PSUM(K) + PAMP(I)
      SSUM(K) = SSUM(K) + SAMP(985-I)
      SKT(K)=SKT(K)+1
      PTK(K)=PTK(K)+1
    END IF
  END FOR
END IF
EXIT LOOP IF(KT .EQ. K2)
END LOOP
CONTINUE
KEEP=K2
PTOTAL=FLOAT (PTOTAL) / (984.* (K2-K1+1))
STOTAL=FLOAT (STOTAL) / (984.* (K2-K1+1))
WRITE (3, ) " PTOTAL=" , PTOTAL , " STOTAL=",STOTAL
WRITE (21, FMT= "2I5") PTOTAL,STOTAL
C WRITE (21, FMT= "(2I5)") PTOTAL,STOTAL
FOR I=1, 90
IF (PSUM(I) .NE. 0 .AND. SSUM(I) .NE. 0)
APPAR(I)=PTOTAL*PKT(I)/FLOAT (PSUM(I))
ASPAR(I)=STOTAL*SKT(I)/FLOAT (SSUM(I))
END IF
WRITE (30, FMT= "(2F12.4)") APPAR(I), ASPAR(I)
END FOR
C P1=0.
C P2=0.
C S1=0.
C S2=0.
FOR I=20,59
P1=P1+APPAR(I)
S1=S1+ASPAR(I)
END FOR
FOR I=30,49
P2=P2+APPAR(I)
S2=S2+ASPAR(I)
END FOR
S3=(S1*P2/FLOAT(S2*P1))
WRITE (21, FMT= "(2F8.4)") P3 , S3
DO 110 I=1,90
ASPAR(I)=1./ASPAR(I)
APPAR(I)=1./APPAR(I)
END FOR
CALL JBAXES(A,90,12.5,5,SHANGLE,5,APPAR,90, 5.0,10H INTENSITY,10)
DO 110 I=1,90
   CALL JOIN PT(A(I),APPAR(I))
110   CALL MARK PT(A(I),APPAR(I),1)
   CALL BREAK
C DO 112 I=1,90
C CALL JOIN PT(A(I),APPAR(I))
c12 CALL MARK PT(A(I),APPAR(I),2)
c CALL BREAK
CALL SET KY (1HT,1HR,1,30)
WRITE (PTITLE, '("PORT INTENSITY, PAR=",F8.4")PTOTAL
CALL MARK KY (2,PTIT(I),39)
C WRITE(STITLE, '("STED INTENSITY, AVE=",I5 ")'STotal
C CALL MARK KY (1,STIT(I),39)
GO TO 11
100 CALL END PLT
STOP
END
APPENDIX A3

C ROLDAVT
C
C <12MAR86> ALTER TO READ NORMAL OR COMPACTED SIDE-SCAN USING
C Routines in Library CONSSF
C
C PROGRAM ROLDAV (COMBINE DAVID WITH ROLLER)
C USE ROLL ALONG TO GET PEAK OF LOCAL HIST PARAM
C THEN DAVID TO DEPROM DATA WITH AV/DEG FILE GENERATED IN AVPROD
C
C DEPROMS THEN REMOVES WATER BOUNCE AND SECOND MULTIPLE
C
C THIS IS DAVIDRT, AN ATTEMPT AT CREATING AN ANGLE DEPENDENT DE-
C PROMMING ROUTINE. TAKES INPUT PAR FILE FROM AVPROM
C MODIFIED 3/19/86 TO SELECT OPTIMAL DEPROM FILE ON THE BASIS OF
C AVERAGE INTENSITY IN THE PING... TRIV
C
C INTEGER DATA(994), PAMP(984), SAMP(984), NPAMP(984), NSAMP(984)
C INTEGER NEW(994), E, IN, OUT, K, SSFAREA(6), SPAREA(6), SFIN, SPIN
C INTEGER NEWAREA(6), SSFOUT, DATAIL, ZERO, SKEEP(100), SSUM
C INTEGER NUM, DIS, MIN, P(25), S(25), SA, SAVE, LEN
C REAL APFAR(90), ASPAR(90), PPAR(25,90), SPAR(25,90), KT, POWER
C REAL ALT, ALT2
C DIMENSION DELM(100), DEBL(100), DEBN(100), XNUM(100)
C
C RADIANS TO DEG TIMES 4 (57.29582 * 4)
C DATA R2D4/229.183312/
C DATA DATAIL/23/
C DATA SSFIN, SPIN, SSFOUT/24,25,26/
C DATA IPROM/0/
C
C K=INDEX IN TABLE OF VALUE TO USE FOR TABLE VALUES THAT ARE ZERO
C ZERO=VALUE TO ADD TO ALL PIXELS
C FUDGE=METERS TO ADD TO ADJUST DEPTH FOR CALCULATING WATER BOUNCE PEAK
C MINUS=#PIXELS BEFORE WATER BOUNCE PEAK TO START WATER BOUNCE AREA
C IPLUS=#PIXELS AFTER FIRST WATER BOUNCE PIXEL TO CONSIDER AS WATER BO
C ICUT=CUTOFF VALU FOR DETERMINING IF WATER BOUNCE (IF IVAL GT ICUT)
C FUDGE2=METERS TO ADD TO ALT WHEN CALCULATING 2ND MULT PEAK
C MULT/MINUS=#PIXELS BEFORE 2ND MULT PEAK TO START CHECKING
C MULT/PLUS=#PIXELS AFTER FIRST 2ND MULT PIXEL TO CONSIDER
C MCUT=CUTOFF VALUE FOR DETERMINING IF 2ND MULT (IF IVAL GT ICUT)
C
C READ (20, -) K,ZERO,FUDGE,MINUS,PLUS,ICUT,FUDGE2,MULT,MULTP,MCUT
C
C IPXN=#PINGS TO ACCUM
C IPXN=# SETS OF IPXN PINGS (UP TO 5)
C LMAG=MAXIMUM SHIFL ALLOWED FROM PARAM FILE PEAK (IN QUARTER DEG BINS
C FROM ONE PING TO THE NEXT
C LLMAG=INITIAL SHIFL ALLOWED FROM PARAM PEAK (IN QUARTER DEGREE BINS
C
C READ (20, -) IPXN, NPXN, LMAG, LLMAG, IDРОBUG, NUM, LEN
C WRITE (3, "(" IPXN,NPXN,LMAG,LLMAG",415') IPXN,NPXN,LMAG,LLMAG
C ISTART=IPXN*IPXN
C
C READ IN HIST FILE FOR DEPROMING (AND COMPUTE PEAK)
PEAK=0
WRITE (3, 1) " NUM=" NUM", LEN=" LEN
M=0
FOR I=1,NUM
  READ (30, FMT="(2I5)") P(I), S(I)
END FOR
FOR I=1,NUM
  FOR J=1,90
    . READ (22, FMT="(2F12.4)", END=10) PPAR(I,J), SPAR(I,J)
  END FOR
END FOR
FOR I=1,90
  APPAR(I)=PPAR(I,I)
END FOR
FOR I=1,360
  READ (22, FMT="(2F12.4)", END=10) APPAR(I), ASPAR(I)
  WRITE (3, 1) " APPAR=" APPAR(I), "ASPAR=" ASPAR(I)
  IF (ASPAR(I).GT. PEAK) THEN
    PEAK=ASPAR(I)
    MPEAK=I
  ENDIF
END FOR
10 CONTINUE
FOR I=1,360
  IF (APPAR(I).EQ. 0.) APPAR(I)=APPAR(K)
  IF (ASPAR(I).EQ. 0.) ASPAR(I)=ASPAR(K)
END FOR
LPEAK=MPEAK
PKLAST=PEAK
WRITE (3, 1) ' MPEAK, PEAK', MPEAK, PEAK

C GET FILE NAMES
READ (DATAFILE, 21, END=100) SSFAREA, SPDAREA, NEWAREA
11 FORMAT (18 (A3))
WRITE (3, 09) NEWAREA
09 FORMAT ( ' GENERATING NEW SIDE SCAN FILE ', 6A3)
CALL ASSIGN (SSFIN, SSFAREA, IERR)
IF (IERR .NE. 0) GO TO 100
CALL ASSIGN (SSFIN, SPDAREA, IERR)
IF (IERR .NE. 0) GO TO 100
IF (IGENR(NEWAREA, 3, 300, 5, 5000) .GT. 1) STOP GENSSSF
CALL ASSIGN (NEWAREA, 3, 5000, IERR)
IF (IERR .NE. 0) GO TO 100

C CHECK FOR SSF FORMAT TYPE (C OR D)
BUFFERIN (SSFIN, DATA, B, 24, E, N)
CALL STATUS (SSFIN)
BUFFERIN (SSFIN, DATA, B, 994, E, N)
CALL STATUS (SSFIN)
NUMWRD=N
REWIND (SSFIN)
READ AND WRITE HEADERS

BUFFER IN(SSFIN, DATA, B, 24, E, N)
CALL STATUS(SSFIN)

CHECK HEADER FLAGS (ENTER M FOR DEPRimed)
CALL SSFCHK('M', DATA)
BUFFER OUT(SSFOUT, DATA, B, 24, E, N)
CALL STATUS(SSFOUT)

ADVANCE PAST FIRST SIX RECORDS OF SPIFILE
READ(25, PRT="(/////)")

MAIN LOOP OVER ALL PINGS/FILE

LOOP
  BUFFER IN (24, DATA, B, 994, E, N)
  CALL STATUS(24)
  IF (E .EQ. 3 .OR. E .EQ. 4)
    WRITE(3, ) "NORMAL END OF INPUT FILE, E=" , E
    CLOSE SSFOUT
    CLOSE SFDIN
    CLOSE SSFIN
    GO TO 11
  END IF
  IF(NUMWRD.LT.994) CALL COND2C(DATA)
    FOR I=1,10
      NEW(I)=DATA(I)
    END FOR
    READ(25,'(34X,3F7.1,7X,2F7.1)',END=10) ALT, DEP, BATHY, GALT, F
    IF(FALT.NE.0) THEN
      ALT=FALT
    ELSE IF (GALT.NE.0) THEN
      ALT=GALT
    END IF
    IF(ALT.EQ.99999.9) ALT=BATHY-DEP

GET DATA OUT OF RAW ARRAY

  FOR I=1,984
    PAMP(I)=(DATA(I+10) .AND. '377) + ZERO
    SAMP(I)=((DATA(I+10)/256) .AND. '377) + ZERO
    SSUM=SSUM+SAMP(I)
  END FOR
  SSUM=NINT(FLOAT(SSUM)/984.)
  IF (M .LT. LEN)
    M=M+1
    SKEEP(M)=SSUM
    SAVE=SAVE+SSUM
  ELSE
    FOR I=1,LEN-1
      SKEEP(I)=SKEEP(I+1)
    END FOR
SKEEP (LEN) = SSUM
SAVE = SAVE - SKEEP (1) + SSUM
END IF
MIN = 255
SA = INT (SAVE / FLOAT (MIN0 (M, LEN)))
FOR I = 1, NUM
DIS = ABS (SA - S(I))
MIN = MIN0 (DIS, MIN)
END FOR
FOR I = 1, NUM
IF (MIN .EQ. ABS (SA - S(I)))
FOR J = 1, 90
ASPAR(J) = SPAR(I, J)
END FOR
EXIT FOR
END IF
END FOR
C GET ROLL ALONG HIST FILE (STD ONLY) AND COMPUTE PEAK
C
IPROM = IPROM + 1
CALL ROLLER (SAMP, IPXN, NP, IPROM, ALT, IPEAK, PKVAL)
IF (IDEBUG .EQ. 1)
* PRINT *, ' IPEAK, PKVAL ', IPEAK, PKVAL
C
C GET SHIFT IN TABLE (STD ONLY)
IF (IPROM .EQ. 1) THEN
IF (ABS (IPEAK - MPEAK) .GT. LMAX) THEN
IPEAK = MPEAK
PKVAL = PEAK
ENDIF
ELSE IF (ABS (IPEAK - LPEAK) .GT. LMAX ) THEN
RESET IF WITHIN RANGE OF NORMAL PEAK
EXIT IF IF (ABS (IPEAK - MPEAK) .LT. LMAX)
IPEAK = LPEAK
PKVAL = PKLAST
ENDIF
ISHFT = MPEAK - IPEAK
ONLY SHIFT PEAK LEFT (UNLESS DEEPER THAN ALT IN ORIGINAL TABLE)
IF (ISHFT .LT. 0) THEN
EXIT IF IF (ALT .GT. 3400.)
ISHFT = 0
IPEAK = MPEAK
PKVAL = PEAK
ENDIF
LPEAK = IPEAK
PKLAST = PKVAL
FACT = PKVAL / PEAK
FACT = 1.
C
IF (IDEBUG .EQ. 1)
* PRINT *, ' ID, ISHFT, IPEAK, PKVAL ', DATA (1), ISHFT, IPEAK, PKVAL
C
NOW DETERMINE BASED ON TABLE
FOR I = 1, 984
C CALCULATE THE ANGLE R, AT WHICH THE ITH PIXEL WILL BE FOUND
. . R = 57.269 * ATAN((I*5. + 00.)/ALT)
C GET INDEX IN TABLE (STBD)
. . LS = R + ISHFT
. . IF(LS.LT.1) LS=1
. . IF(LS.GT.90) LS=90
C GET FORT INDEX
. . L = R
. . IF(L.LT.1) L=1
. . IF(L.GT.90) L=90
C PRINT *,L,APPAR(L),I,NAMP(I)
. . NAMP(I) = MIN0 (255, INT(PAMP(I) * APPAR(L) ) )
. . NSAMP(985-I) = MIN0 (255, INT(SAMP(985-I) * ASPAR(LS) ) )
END FOR
C USING DEFORMED DATA,
C FOR PIXELS IN THE WATER BOUNCE AREA, ADJUST BY FIRST
C CHECKING THAT THERE IS WATER BOUNCE INDICATED IN DATA (VAL > ICUT)
C AND IF INDICATED, RECURRENT USING A VALUE FROM DATA OUTSIDE WATER
C BOUNCE AREA.
C GET APPROX ANGLE WHERE MIDPOINT OF WATER BOUNCE OCCURS FOR THIS PIN
. . ZALT = ALT + DEP + FUDGE
. . R2ND = ACOS (ALT/ZALT)
C GET PIXEL RANGE FOR WATER BOUNCE (NOTE PIXEL 1 IS AT 200 METERS)
. . I1 = (ZALT * SIN(R2ND) - 200.) / 5. - MINUS
. . I2 = I1 + IPPLUS
C REPLACE WITH ADJACENT PIXELS
C ONLY IF GREATER THAN CUTTOF
C FOR
. . FOR I=I1,I2
. . . IF (NAMP(I).GE.ICUT) THEN
. . . . FOR J=1,10
. . . . . ITEMP=NAMP(I2+10+IRAN(-5,5))
. . . . . EXIT FOR IF(ITEMP.GT.00)
. . . . ENDIF
. . . NPAMP(I)=ITEMP
. . ENDIF
. . END FOR
C STBD
. . FOR I=I1,I2
. . . IF (NSAMP(985-I).GE.ICUT)
. . . . NSAMP(985-I)=NSAMP(985-(I2+10+IRAN(-5,5)))
. . END FOR
C FOR PIXELS IN THE 2ND MULTIPLE AREA, ADJUST BY FIRST
C CHECKING THAT THERE IS 2ND MULTIPLE INDICATED IN DATA (VAL > ICUT)
C AND IF INDICATED, RECURRENT USING A VALUE FROM DATA OUTSIDE 2ND
C MULT AREA.
C GET APPROX PIXEL WHERE MIDPOINT OF 2ND MULTIPLE OCCURS FOR THIS PIN
. . ZALT = (ALT + FUDGE) * 2 + DEP
. . R2ND = ACOS (ALT/ZALT)
C GET PIXEL RANGE FOR 2ND MULT (NOTE PIXEL 1 IS AT 200 METERS)
  I1=(ZALT*SIN(R2ND)-200.)/5. - MULTM
  I2=II+MULTP
  IF(I2.GT.984) GOTO 222
C REPLACE WITH ADJACENT PIXELS
C ONLY IF GREATER THAN CUTOFF
C PORT
  FOR I=II,I2
    IF(NPAMP(I).GT.MCUT)
      NPAMP(I)=NPAMP(I-20-IRAN(-5,5))
    ENDFOR
C STBD
  FOR I=II,I2
    IF(NSAMP(985-I).GT.MCUT)
      NSAMP(985-I)=NSAMP(985-(I-20-IRAN(-5,5)))
    ENDFOR
C C SET UP NEW DEPRMED DATA
C 222  CONTINUE
  FOR I=II,994
    NEW(I)=NPAMP(I-10) .AND. '377
    NEW(I)=NEW(I) .OR. (NSAMP(I-10) .AND. '377)*256
  END FOR
  IF(NUMVID.LT.994) CALL CONC2D(NEW)
  BUFFER OUT(SSFOUT,NEW,B,NUMRD,E,N)
  CALL STATUS(SSFOUT)
  IF(E.EQ.30.R.E.EQ.4) STOPNODSC
END LOOP
C
C 100 CONTINUE
STOP
END
C
SUBROUTINE SSFCHK(ITYPE,INBUF)
C
SUBROUTINE TO CHECK FLAGS IN SSF HEADER
C CHARACTERS 55-63
C APPEND FLAG "ITYPE" TO EXISTING FLAGS
C ASSUMES INBUF IS 3-BYTE INTEGER ARRAY
C
INTEGER INBUF(1)
CHARACTER*9 CHARS
CHARACTER*1 ITYPE
WRITE(CHARS,31) (INBUF(IO),IO=19,21)
31 FORMAT(3A3)
IF(CHARS(1:1) .EQ. ' ') THEN
  CHAR$=ITYPE
ELSE
  DO 30 I=2,9
    IF (CHARS(I:I) .EQ. ' ') GO TO 29
  CONTINUE
  I=9
            

29 .  COMMON/PARAM/PMIN,PMAX,SMIN,SMAX
   INTEGER PSUM(984,5),SSUM(984,5),DATA(1),PPSUM(984),SSSUM(984)
   INTEGER IPROM(IPXN),IPXN,NPXN
   REAL PDEV(984),PDEV(984)
   REAL IDEV(984),PDEV(984)
   REAL IDEW,PPDEV
   INTEGER IPN(984)
   REAL AMEAN(5)
   DATA AMEAN/5*0/.
   DATA PMEAN,SMEAN/0,0/.
   DATA AVALT/0/.
   DATA INDLAST/0/.
   C RADIANS TO DEG TIMES 4 (57.29582 * 4)
   DATA R2D4/229.183312/.
   DATA R2D4/57.29582/.
   IND=MOD(IPROM/IPXN + 1,NPXN) + 1
   IPXN=MOD(IPROM,IPXN)
   IF (IPXN.EQ.0) THEN
      IPXN=IPXN
      IND=IND-1
      IF (IND.EQ.0) IND=NPXN
   ENDIF
   INDLAST=IND
   C WRITE (3,-) 'IND,IPX',IND,IPX
   C GET LATEST AV/PIX (PSUM,SSUM) AND MEAN/SIDE (PMEAN,SMEAN)
   FOR N=1,984
   C REMOVE LAST ONE
      SSSUM(N)=SSSUM(N)-SSUM(N,IND)
      SMEAN=SMEAN-SSUM
   C GET NEW
      SSSUM(N)=SSSUM(N)*(IPXN-1) + ISTBD(N)/IPX
   C ADD IN NEW
      SSSUM(N)=SSUM(N)+SSUM(N,IND)
      SMEAN=SMEAN+SSUM(N,IND)
   ENDFOR
   SRMEAN=SMEAN/984
   C GET AVALT
      AMEAN(IND)= (AMEAN(IND)*(IPXN-1) + ALT)/IPX
      AVALT=AMEAN(IND)
   C WRITE (3,-) 'AVALT',AVALT
   RETURN
END
C
C NOW GET TABLE BY ANGLE
FOR K=1, 90
   NUM(K)=0
   ASPAR(K)=0
ENDFOR
FOR I = 1, 984
   GET ANGLE ASSOCIATED WITH ITH PIXEL (TO NEAREST QUARTER DEGREE
   L = R2D4 * ATAN((I*5 + 00)/AVALT)
   IF (L.GT. 90) L= 90
   IF (SSSUM(985-I) .NE. 0) THEN
      TEMP = SRMEAN/SSSUM(985-I)
      ASPAR(L) = ASPAR(L) + TEMP
      NUM(L)=NUM(L)+1
      KMAX=MAX(KMAX,L)
   ENDIF
END FOR
END FOR
FOR K=1,KMAX
   IF (NUM(K).NE.0) THEN
      ASPAR(K)=ASPAR(K)/NUM(K)
   ENDIF
NOW FIND PEAK OF NEW TABLE
   IF (ASPAR(K).GT. PEAK) THEN
      IPEAK=K
      PEAK=ASPAR(K)
   ENDIF
ENDFOR
ENDFOR
C WRITE(3,-) ' IPEAK',IPEAK,' ASPAR(IPEAK)', ASPAR(IPEAK)
C WRITE(3,-) ' ASPAR', (ASPAR(I),I=100,120), (ASPAR(I),I=200,220)
C
RETURN
END
APPENDIX A4

PROGRAM ZFILT2 A FIRST ATTEMPT AT FILTERING BATHYMETRY ON THE BASIS OF INVERSE EUCLIDEAN DISTANCE. 2/24/86

REAL X, ZX, PY(5,70), SY(5,70), YPOS, XPOS, NZ, RX, WSUM, W
REAL PX(5,70), SX(5,70), PZ(5,70), SZ(5,70), YMIN, YMAX
REAL X0(6), Y0(6), Z0(6), LIM, SEP, DIFZ, MIN, XMIN
INTEGER KS(6), KP(6), NPTS, LA, P(6), FLAG
INTEGER XYZAREA(6), NEWAREA(6), XYZIN, XYZOUT, DATAFIL
CHARACTER*72 COPYRT1, COPYRT2
DATA COPYRT1//" SEAMARC II PROGRAM ZFILT2 "/
DATA COPYRT2//" COPYRIGHT 1985 THOMAS B. REED IV "/
DATA DATAFIL/23/
DATA ZXFIN, XYZOUT/24, 25/

WRITE (3, 8612) COPYRT1, COPYRT2

8612 FORMAT(/, 1X, A, /)

C GET FILE NAMES
11 WRITE (3, ) " GETTING FILE NAMES"
READ (DATAFIL, 19, END=998) XYZAREA, NEWAREA
WRITE (3, 19) XYZAREA, NEWAREA
19 FORMAT (24 (A3))
WRITE (3, 09) NEWAREA
09 FORMAT (" GENERATING NEW BATHYMETRY FILE ", 6A3)
CALL ASSIGN (XYZIN, XYZAREA, IERR)
IF (IERR .NE. 0) GO TO 998

C IF (IGENR(NEWAREA, 3, 300, 5, 5000) .GE. 1) STOP GENSSF
IF (FLAG .NE. 0) GO TO 1
CALL ASSIGN (XYZOUT, NEWAREA, IERR)
IF (IERR .NE. 0) GO TO 998

C MAIN LOOP OVER ALL PINGS/FILE

C READ (22, FMT="(F10.2)") LIM
1 READ (XYZIN, 1000) X, Y, ZX, K
WRITE (3, ) " FIRST LINE OF XYZ FILE"
1000 FORMAT (3F10.4, I10)
IF (FLAG .NE. 0)
  X0(6) = X
  Y0(6) = Y
  Z0(6) = ZX
  P(6) = K
  GO TO 2
END IF
X0(1) = X
Y0(1) = Y
Z0(1) = ZX
P(1) = K
1010 FORMAT (3F10.4)
5 KL = 0
FOR I=1, 5
  KP (I) = 0
  KS (I) = 0
  KL = KL + 1

FOR J=1,70
  PX(I,J)=0.
  SX(I,J)=0.
  PZ(I,J)=0.
  SZ(I,J)=0.
  PY(I,J)=0.
  SY(I,J)=0.
END FOR
END FOR

C WRITE (3,) " BEGIN LOOP"
FOR I=1,5
  READ (XYZIN,1000,END=5) X,Y,ZX,K
  IF (Y) 16,17,15
      KP(I)=KP(I)+1
  C WRITE (3,) " KP=",KP
    M=KP(I)
    PX(I,M )=X
    PY(I,M )=Y
    PZ(I,M )=ZX
    GO TO 10
  15  KS(I)=KS(I)+1
    M=KS (I)
    PX(I,M )=X
    SY(I,M)=ABS(Y)
    SZ(I,M)=ZX
    GO TO 10
  C WRITE (3,) " KS=",KS
  FOR I=1,2
      WRITE(25,1000) XO(I),YO(I),ZO(I),P(I)
      WRITE(25,1010) PX(I,J),PY(I,J),PZ(I,J)
      FOR J=1,70
          PX(I,J)=PX(I+1,J)
          SY(I,J)=SY(I+1,J)
          PZ(I,J)=PZ(I+1,J)
          SX(I,J)=SX(I+1,J)
          SY(I,J)=SY(I+1,J)
          SZ(I,J)=SZ(I+1,J)
      END FOR
      P(I)=P(I+1)
      WRITE(25,1000) X0(I),YO(I),ZO(I),P(I)
      WRITE(25,1010) PX(I,J),PY(I,J),PZ(I,J)
      FOR J=1,70
          PX(I,J)=PX(I+1,J)
          SY(I,J)=SY(I+1,J)
          PZ(I,J)=PZ(I+1,J)
          SX(I,J)=SX(I+1,J)
          SY(I,J)=SY(I+1,J)
          SZ(I,J)=SZ(I+1,J)
      END FOR
      P(I)=P(I+1)
END FOR
END FOR
\[
\begin{align*}
\text{XO(I)} &= \text{XO}(I+1) \\
\text{YO(I)} &= \text{YO}(I+1) \\
\text{Z0(I)} &= \text{Z0}(I+1) \\
\text{KP(I)} &= \text{KP}(I+1) \\
\text{KS(I)} &= \text{KS}(I+1) \\
\text{END FOR} \\
\text{X0}(5) &= \text{X0}(6) \\
\text{YO}(5) &= \text{YO}(6) \\
\text{Z0}(5) &= \text{Z0}(6) \\
\text{P}(5) &= \text{P}(6) \\
\text{KP}(5) &= 0 \\
\text{KS}(5) &= 0 \\
\text{READ(XYGIN,1000,END=999)} &\quad \text{X,Y,ZX,K} \\
\text{IF(Y) 26,27,25} \\
\text{KP}(5) &= \text{KP}(5)+1 \\
\text{M} &= \text{KP}(5) \\
\text{WRITE(3,) " KP=" ,KP} \\
\text{PX}(5,M) &= X \\
\text{PY}(5,M) &= Y \\
\text{PZ}(5,M) &= ZX \\
\text{GO TO 20} \\
\text{KS}(5) &= \text{KS}(5)+1 \\
\text{M} &= \text{KS}(5) \\
\text{WRITE(3,) " KS=" ,KS} \\
\text{SX}(5,M) &= X \\
\text{SY}(5,M) &= \text{ABS}(Y) \\
\text{SZ}(5,M) &= ZX \\
\text{GO TO 20} \\
\text{Z0}(6) &= ZX \\
\text{X0}(6) &= X \\
\text{YO}(6) &= Y \\
\text{P}(6) &= K \\
\text{IF(FLAG .EQ. 1)} \\
\text{KF} &= \text{KF}+1 \\
\text{IF(KF .GT. 2)} \\
\text{CLOSE XYZOUT} \\
\text{CALL ASSIGN(XYZOUT,NEWAREA,IERR)} \\
\text{IF(IERR .NE. 0) GO TO 998} \\
\text{FLAG} &= 0 \\
\text{KF} &= 0 \\
\text{END IF} \\
\text{END IF} \\
\text{DETERMINE IF NADIR POINT IS REASONABLE} \\
\text{C} \\
\text{IF(Z0(3) .LT. MIN1(SZ(3,1),PZ(3,1))} \\
\text{OR. Z0(3) .GT. MAX1(SZ(3,1),PZ(3,1)))} \\
\text{SEP} &= \text{PY}(3,1)+\text{SY}(3,1) \\
\text{DIFZ} &= \text{ABS}(\text{PZ}(3,1)-\text{SZ}(3,1)) \\
\text{MIN} &= \text{MAX1}(\text{SZ}(3,1),\text{PZ}(3,1)) \\
\text{IF(MIN .EQ. SZ(3,1))} \\
\text{XMIN} &= (\text{SY}(3,1)) \\
\text{ELSE} \\
\text{XMIN} &= \text{PY}(3,1)
\end{align*}
\]
START PROCESSING IN SEPARATE LOOPS FOR S&P

99 WRITE(25,1000) X0(3),Y0(3),Z0(3),P(3)

YMIN=PY(3,1)
YMAX=PY(3,KP(3))
NPTS=NINT((YMAX-YMIN)/100.)
IF (NPTS .EQ. 0) WRITE(3,) " NPTS=" ,NPTS, " COUNT=" ,COUNT
YPOS=YMIN
FOR I=1,NPTS
  WSUM=0.
  RZ=0.
  XPOS=X0(3)
  YPOS=YPOS+100.
  KT=0
  ATXY=0.
  FOR J=1,5
    FOR K=1,KP(3)
      W=SQRT((XPOS-PX(J,K))**2 + (YPOS-PY(J,K))**2)
      IF (W .LE. LIM)
        IF (W .NE. 0)
          WSUM=WSUM+1./W
        ELSE
          ATXY=ATXY+PZ(J,K)
        END IF
      END IF
    END FOR
  END FOR
  IF (WSUM .NE. 0)
    Nl=(RZ/WSUM + ATXY)/(1.+FLOAT(KT))
    WRITE(25,1010) XPOS,YPOS,Nl
  END IF
END FOR

YMIN=SY(3,1)
YMAX=SY(3,KS(3))
WRITE(3,) " YMAX=" ,YMAX," KS=" ,KS(3)
NPTS=NINT((YMAX-YMIN)/100.)
WRITE(3,) " NPTS=" ,NPTS," COUNT=" ,COUNT
YPOS=YMIN
FOR I=1,NPTS
  WSUM=0.
  RZ=0.
  XPOS=X0(3)
  YPOS=YPOS+100.
  ATXY=0.
  KT=0
  FOR J=1,5
FOR K=1,KS(3)
    W=SQRT((XPOS-SX(J,K))**2 + (YPOS-SY(J,K))**2)
    IF (W .LE. LIM)
        IF(W .NE. 0)
            WSUM=WSUM+1./W
            RZ=RZ+SZ(J,K)/W
        ELSE
            ATXY=ATXY+SZ(J,K)
            KT=KT+1
        END IF
    END IF
END FOR
IF (WSUM .NE. 0)
   N'l=(RZ/WSUM + ATXY)/(1.+FLOAT(KT))
   WRITE(25,1010) XRJS,-1*YFOS,N'l
END IF
END FOR

COUNT = COUNT + 1
END LOOP

999 CONTINUE
FLAG=1
KF=0
GO TO 11

998 FOR I=3,5
    WRITE(25,1000) X0(I),Y0(I),Z0(I),P(I)
    FOR J=1,KP(I)
        WRITE(25,1010) PX(I,J),PY(I,J),PZ(I,J)
    END FOR
    FOR J=1,KS(I)
        WRITE(25,1010) SX(I,J),-1*SY(I,J),SZ(I,J)
    END FOR
END FOR
STOP
END
APPENDIX A5 "1000SM2*DELAY2"

PROGRAM DELAY, A FIRST ATTEMPT AT REMOVING LAYOVER FROM SIDE-
SCAN IMAGES. TRIV. 1,28,86.
REAL X,Z,XZ,ZO,ZSAVE,XSAVE,ZZERO,FDEPTH,Z0SAVE
REAL PX(70),SX(70),PZ(70),SZ(70),MIN,MINX,SEP,DIFZ
INTEGER DATA(994),PNEW(984),SNEW(984),PPD(984),SOLD(984)
INTEGER XSTART,TSTART,FLAG1,FLAG2,IXZ,ITZ,K1,K2,KS,KP
INTEGER NEW(994),E,IN,CUT,K,SSFAREA(6),ZXFAREA(6),SSFIN,ZXFIN
INTEGER NEWAREA(6),CSSOUT,DATAFIL,PMOUNT,PAKEEP,PIKEEP
INTEGER SSTYPE,SSSTAR,FLAG3,SPAREA(6),SSDIN,FLAG4
CHARACTER*72 COPYRT1,COPYRT2
DATA COPYRT1/"SEAMARC II PROGRAM DELAY "/
DATA COPYRT2/"COPYRIGHT 1985 THOMAS B. REED IV "/
DATA DATAFIL/23/
DATA SSFIN,ZXFIN,SSDIN,SSFOUT/24,25,26,27/
C
C WRITE (3, 8612) COPYRT1,COPYRT2
8612 FORMAT(/,1X,A,/) C GET FILE NAMES
11 WRITE (3,) "GETTING FILE NAMES"
READ (DATAFIL,19,END=100) SSFAREA,ZXFAREA,SPAREA,NEWAREA
WRITE (3,19) SSFAREA,ZXFAREA,SPAREA,NEWAREA
19 FORMAT(24(A3))
WRITE (3,09) NEWAREA
09 FORMAT('Generating new side scan file ',A3)
CALL ASSIGN(SSFIN,SSFAREA,IERR)
IF (IERR .NE. 0) GO TO 100
CALL ASSIGN(ZXFIN,ZXFAREA,IERR)
IF (IERR .NE. 0) GO TO 100
CALL ASSIGN(SSDIN,SPAREA,IERR)
IF (IERR .NE. 0) GO TO 100
IF (IGENR(NEWAREA,3,300,5,5000),GT. 1) STOP GENSSF
CALL ASSIGN(SSFOUT,NEWAREA,IERR)
IF (IERR .NE. 0) GO TO 100
C
C READ AND WRITE HEADERS
C
BUFFER IN(SSFIN,DATA,B,24,E,N)
CALL STATUS(SSFIN)
WRITE (3,) "SSF HEADER BUFFERED IN"
C
CHECK HEADER FLAGS (ENTER N FOR DELAYOVER)
CALL SSFCHK('N',DATA)
BUFFER OUT(SSFOUT,DATA,B,24,E,N)
CALL STATUS(SSFOUT)
WRITE (3,) "HEADER BUFFERED OUT"
C
C SKIP OVER SSD FILE HEADER
C
READ(SSDIN,fmt="(/*)")
WRITE (3,) "SSD HEADER BUFFERED IN"
BUFFER IN (24,DATA,B,994,E,N)
CALL STATUS(24)
IF (E .EQ. 3 .OR. E .EQ. 4)
  WRITE (3,) "NORMAL END OF INPUT FILE , E=",E
CLOSE SSFOUT
GO TO 11
END IF
READ(SHDFIN, FMT="(42X,F6.1)") FDEPTH
WRITE(3, )" FISH DEPTH="FDEPTH
FOR I=1,994
NEW(I)=DATA(I)
END FOR

C
C MAIN LOOP OVER ALL PINGS/FILE
C
READ(ZXFIN,1010) X,Z0,POCOUNT
WRITE(3, )" FIRST LINE OF XYZ FILE"
1000 FORMAT(10X,2F10.4)
1010 FORMAT(10X,2F10.4,I10)
K1=1
GO TO 99
5 LOOP
   . KP=0
   . KS=0
   . K1=K1+1
   . FOR I=1,70
      . PX(I)=0.
      . SX(I)=0.
      . PZ(I)=0.
      . SZ(I)=0.
   . END FOR
   C WRITE(3, )" BEGIN LOOP"
   . BUFFER IN (24,DATA,B,994,E,N)
   . CALL STATUS(24)
   . IF (E .EQ. 3 .OR. E .EQ. 4)
   . WRITE(3, )" NORMAL END OF INPUT FILE , E="E
   . CLOSE SSFOUT
   . GO TO 11
   . END IF
   . READ(SHDFIN, FMT="(42X,F6.1)") FDEPTH
   . WRITE(3, )" FISH DEPTH="FDEPTH
C IF(POCOUNT .NE. K1)
   FOR I=1,994
   NEW(I)=DATA(I)
   END FOR
   GO TO 99
   END IF
C
C REMOVE FISH DEPTH FROM NADIR DEPTH
C
   Z0=Z0-FDEPTH
C
C CONVERT FROM D TO C IF NECESSARY
C IF(N,LT.994) CALL COND2C(DATA)
   FOR I=1,10
      NEW(I)=DATA(I)
   END FOR
C GET DATA OUT OF RAW ARRAY
C PAMP(1-984)=PIXEL 1-984
C SAMP(1-984)=PIXEL 984-1
C
C FOR I=1,984
.  FOLD(I)= (DATA(I+10) .AND. '377)
.  SOLD(I)=((DATA(I+10)/256) .AND. '377)
. END FOR
C
C NOW DELAY DATA BASED ON THEORY OF PUSH-ME-PULL-YOU...
C
C INITIALIZE FLAGS AND COUNTERS
.  FLAG1=0
.  FLAG3=0
.  FLAG4=0
.  SXSTART=0
.  STSTART=0
.  XSTART=0
.  TSTART=0
.  K2=0
10  READ (ZXFIN,1000,END=5) X,ZX
.  ZX=ZX-FDEPTH
.  K2=K2+1
.  IF(X) 16,17,15
15  KP=KP+1
C WRITE (3,) " KP=" ,KP
.  PX(KP)=X
.  PZ(KP)=ZX
.  GO TO 10
16  KS=KS+1
C WRITE (3,) " KS=" ,KS
.  SX(KS)=X
.  SZ(KS)=ZX
.  GO TO 10
17  ZOSAVE=ZX+FDEPTH
C
C DETERMINE IF NADIR POINT IS REASONABLE
C
.  IF(Z0 .LT. MIN1(SZ(1),PZ(1)) .OR. Z0 .GT. MAX1(SZ(1),PZ(1)))
.    SEP=PX(1)-SX(1)
.    DIFZ=ABS(PZ(1)-SZ(1))
.    MIN=MAX1(SZ(1),PZ(1))
.    IF(MIN .EQ. SZ(1))
.      XMIN=ABS(SX(1))
.    ELSE
.      XMIN=PX(1)
.    END IF
C WRITE(3,) PZ(1),SZ(1),Z0,DIFZ,XMIN,SEP
.    Z0=MIN - DIFZ*XMIN/SEP
. END IF
C
C START PROCESSING IN SEPARATE LOOPS FOR S&P
FOR J=1, KP
    IF(ZO-PZ(J) .LT. 0) FLAG2=-1
    IF(ZO-PZ(J) .EQ. 0) FLAG2=0
    IF(ZO-PZ(J) .GT. 0) FLAG2=1
    X=PX(J)
    ZX=PZ(J)
    FLAG2=-1
    CONTINUE
    IF(FLAG1*FLAG2 .LE. -1) GO TO 50

CALCULATE COUNTER POSITIONS FOR TRUE DISTANCE(IX) AND TIME

DISTANCE (IT) IE., SLANT RANGE ASSUMING FLAT BOTTOM.

IX=NINT((X-200)/5.)-XSTART
TS=ZX*ZX + X*X
D=TS-ZO*ZO
    IF(D .LT. 0)
        WRITE(3, ) " WILD HAIR POINT AT", " PING ", KL," POINT#"
        WRITE(3, ) " NADIR DEPTH =", ZO, " X=" ,X, " ZX=" ,ZX
        FLAG4=1
        GO TO 22
    END IF
    IT=NINT((SQRT(D)-200)/5.)-TSTART
    IF(FLAG4 .NE. 0)
        N=INT(IX*PX(1)/X)
        FOR I=N+1, IX
            PNEW(N+I+XSTART)=NINT(FOLD(I-1) *1.)
            END FOR
            FOR I=1, N
                PNEW(N-I+1+XSTART)=NINT(FOLD(I) *1.)
                END FOR
                TSTART=TSTART+IX
        ELSE
            FOR I=1, IX
                M=MAX0(TSTART+NINT(I*IT/FLOAT(IX)),1)
                PNEW(I+XSTART)=FOLD(M)
                END FOR
                    TSTART=XSTART+IX
                    TSTART=TSTART+IT
                    PXKEEP=XSTART
                    PXKEEP=TSTART
                    FLAG1=FLAG2
                    ZSAVE=ZX
                    XSAVE=X
                    FLAG4=0
                    GO TO 22
                END IF

SPECIAL PUSH-PULL, FOR ZERO CROSSINGS
                CONTINUE
XZERO = (X - XSAVE) * (ABS(ZSAVE - ZO) / ABS(ZX - ZSAVE))
IXZ = NINT(XZERO / 5.)
ITZ = NINT((XSAVE + XZERO - 200) / 5.) - TSTART
FOR I = 1, IXZ
  M = MAX0(TSTART + NINT(ITZ*I/FLOAT(IXZ)), 1)
  FNEW(XSTART + I) = RFLD(M)
END FOR
XSTART = XSTART + IXZ
TSTART = TSTART + ITZ
FLAG1 = 0
END FOR

STARTING STARBOARD PROCESSING SECTION

FLAG1 = 0
FLAG3 = 0
FLAG4 = 0
FOR J = 1, KS
  X = ABS(SX(J))
  ZX = SZ(J)
  IF(ZO - ZX .LT. 0) FLAG2 = -1
  IF(ZO - ZX .EQ. 0) FLAG2 = 0
  IF(ZO - ZX .GT. 0) FLAG2 = 1
END FOR

COMMENCE PUSH-FULL

IF(FLAG3 * FLAG2 .EQ. -1) GO TO 51

CALCULATE COUNTER POSITIONS FOR TRUE DISTANCE (IX) AND TIME
DISTANCE (IT) IE., SLANT RANGE ASSUMING FLAT BOTTOM.

IX = NINT((X - 200) / 5.) - SXSTAR
IF((IX .LE. 0) GO TO 23
TS = ZX*ZX + X*X
D = TS - ZO*ZO
IF(D .LT. 0)
  WRITE(3, ) " WILD HAIR POINT AT", " PING ", K1, " POINT"#
  WRITE(3, ) " NADIR DEPTH =", ZO, " X=" X, " ZX=" ZX
  FLAG4 = 1
  GO TO 23
END IF
IT = NINT((SQRT(D) - 200) / 5.) - SXSTAR
IF(FLAG4 .NE. 0)
  N = INT(IX*ABS(SX(1))/X)
  FOR I = N + 1, IX
    M = 985 - SXSTAR
    R = (IX - .25*I) / FLOAT(IX)
    SNEW(M - I) = NINT(SOLD(M - I + N) * 1.)
  END FOR
  FOR I = 1, N
    M = 985 - SXSTAR
    R = (IX - .25*I) / FLOAT(IX)
    SNEW(M - N + I - 1) = NINT(SOLD(M - I) * 1.)
E1'D FOR STSTAR=STSTAR+IX
  ELSE
    FOR I=1,IX
      M=MINO(STSTAR+NINT(I*IT/FLOAT(IX)),984)
      IF (M .LT. 1)
        M=1
      END IF
      SNEW(985-I-SXSTAR)=SOLD(985-M)
    END FOR
    SXSTAR=SXSTAR+IX
    STSTAR=STSTAR+IT
    FLAG3=FLAG2
    ZSAVE=ZX
    XSAVE=X
    FI.JC4=0
    GO TO 23
C
C SPECIAL PUSH-PULL, FOR ZERO CROSSINGS
C
51  CONTINUE
  XZERO=(X-XSAVE) * (ABS(ZSAVE-ZO)/ABS(ZX-ZSAVE))
  IDZ=NINT((XZERO/5.).)
  ITZ=NINT((XSAVE+XZERO-200)/5.)-STSTAR
  FOR I=1,IDZ
    M=MINO(STSTAR+NINT(ITZ*I/FLOAT(IDZ)),984)
    IF (M .LT. 1)
      M=1
    END IF
    SNEW(985-IDZ-I)=SOLD(985-M)
  END FOR
  SXSTAR=SXSTAR+IDZ
  STSTAR=STSTAR+ITZ
  FLAG3=0
23  END FOR
C
C 999  CONTINUE
C COMPLETE THE REST OF PORT LINE
  FOR I=1,984-PXKEEP
    PNEW(I+PXKEEP)=FOLD(PKEEP+I*(984-PKEEP)/(984-PXKEEP))
  END FOR
C COMPLETE THE REST OF STBD LINE
  L=984-SXSTAR
  FOR I=1,L
    M=MINO(STSTAR+I*(984-STSTAR)/(L),984)
    IF (M .LT. 1)
      M=1
    END IF
    SNEW(985-I-SXSTAR)=SOLD(985-M)
  END FOR
  FOR I=11,994
    NEW(I)=PNEW(I-10) .AND. '377
NEW(I) = NEW(I) .OR. (SNEW(I-10) .AND. '377) * 256
END FOR
POCOUNT = POCOUNT + 1
Z0 = Z0SAVE
BUFFER OUT (SSFOUT, NEW, B, 994, E, N)
CALL STATUS (SSFOUT)
END LOOP
STOP
END
FILE: "1000SM2*DELAY2"
SUBROUTINE SSFCHK (IYPE, INBUF)
C
C SUBROUTINE TO CHECK FLAGS IN SSF HEADER
C CHARACTERS 55-63
C APPEND FLAG "IYPE" TO EXISTING FLAGS
C ASSUMES INBUF IS 3-BYTE INTEGER ARRAY
C
INTEGER INBUF(24)
CHARACTER*9 CHAR
CHARACTER*1 IYPE
WRITE (CHAR, 31) (INBUF(IO), IO=19, 21)
FORMAT (3A3)
IF (CHARS (1:1) .EQ. ' ') THEN
    CHAR = IYPE
ELSE
    DO 30 I = 2, 9
        IF (CHARS (I:I) .EQ. ' ') GO TO 29
    CONTINUE
    I = 9
29    CHAR = CHAR (1:I-1) // IYPE
    END IF
READ (CHAR, 31) (INBUF(IO), IO=19, 21)
RETURN
END
APPENDIX A5
C**********************************************************************
cC 'THIS IS MIGRATE
cC**********************************************************************
INTEGER DATA(994),IPAMP(984),ISAMP(984),IN,E,DATAFIL,FILNAME(3)
INTEGER SSFILE,NDATA(994),OUT,SPDFILE
REAL D
COMMON NPAMP(984,9),NSAMP(984,9)
SPECIAL COMMON
DATA DATAFIL,SPDFILE,OUT/20,22,26/
DATA SSFILE/24/
10 READ (DATAFIL,21,END=100) FILNAME
21 FORMAT(3(A3))
CALL ASSIGN(SSFILE,FILNAME,IERR)
IF (IERR .NE. 0) STOPAS24
BUFFER IN(24,DATA,B,24,E,N)
CALL STATUS (24)
IF (E .EQ. 3 .OR. E .EQ. 4) GO TO 100
K=1
FOR J=1,7
  BUFFER IN(24,DATA,B,994,E,N)
  CALL STATUS (24)
  IF (E .EQ. 3 .OR. E .EQ. 4)
    .CT.iOSE SSFILE
  END IF
  IF (J .LE. 3)
    BUFFER OUT(OUT,DATA,B,994,E,N)
    CALL STATUS(OUT)
  END IF
FOR I=1,984
  NPAMP(I,J) = (DATA(I+10) .AND. '377)
  NSAMP(985-I,J) = (DATA(I+10)/256) .AND. '377
END FOR
END FOR
READ(22,25)
25 FORMAT(/////////)
LOOP
  IF (K .GT. 1)
    FOR J=1,6
      FOR I= 1,984
        NPAMP(I,J) = NPAMP(I,J+1)
        NSAMP(I,J) = NSAMP(I,J+1)
      END FOR
    END FOR
    BUFFER IN(24,DATA,B,994,E,N)
    CALL STATUS (24)
    IF (E .EQ. 3 .OR. E .EQ. 4)
      CLOSE SSFILE
      GO TO 10
    END IF
  END IF
• FOR I=1,984
  • NPAMP(I,7)=(DATA(I+10) .AND. '377)
  • NSAMP(985-I,7)=(DATA(I+10)/256) .AND. '377
  • END FOR
• END IF
• READ(22, FMT="(35X, 1F8.3)") D
• CALL FILTER(NPAMP,D,IPAMP)
• CALL FILTER(NSAMP,D,ISAMP)
• FOR I=11,994
  • NDATA(I)=(IPAMP(I-10) .AND. '377)
  • NDATA(I)=NDATA(I) .OR. (ISAMP(985-I+10) .AND. '377)*256
  • END FOR
• BUFFER OUT(OUT, NDATA, B, 994, E, N)
• CALL STATUS(OUT)
• K=K+1
END LOOP
100 CONTINUE
FOR J=5,7
  • FOR I=11,994
    • NDATA(I)=(NPAMP(I-10,J) .AND. '377)
    • NDATA(I)=NDATA(I) .OR. (NSAMP(985-I+10,J) .AND. '377)*256
    • END FOR
  • BUFFER OUT(OUT, NDATA, B, 994, E, N)
  • CALL STATUS(OUT)
END FOR
STOP
END

C**********************************************************************
C
SUBROUTINE FILTER(AMP,D,NAMP)
C
C**********************************************************************
INTEGER AMP(984,9),NAMP(984),SUM(984),DIST,P,NP,P1,P2,MAX
REAL D
NP=100
P1=50
P2=100
FOR I=1,NP
  • NAMP(I)=AMP(I,4)
END FOR
FOR I=978,984
  • NAMP(I)=AMP(I,4)
END FOR
MAX=0
FOR I=NP+1,977
  • SUM(I)=0
  • IF(AMP(I,3) .LE. P1 .OR. AMP(I,3) .GE. P2)
    • FOR J=1,5
      • SUM(I)=2*(AMP(I,J)+AMP(I+1,J)+AMP(I+2,J)+AMP(I+3,J))-(AMP(I+5,J)-AMP(I-1,J)-AMP(I-2,J)+SUM(I)-AM)
    • END FOR
    • MAX= MAX0(MAX,ABS(SUM(I)))
  • ELSE
    • SUM(I)=0
  • END IF
NAMP(I) = AMP(I,3)
END IF
END FOR
FOR I = NP+1, 977
  IF (SUM(I) .GE. .10*MAX)
    NAMP(I) = MIN0 (AMP(I-4,3), AMP(I+4,3))
  ELSE
    NAMP(I) = AMP(I,3)
  END IF
C WRITE (3,) AMP(I,3), NAMP(I), SUM(I), MAX
END FOR
RETURN
END
APPENDIX B1

PROGRAM MULTFIL A FILTER WHICH PRODUCES LOW, HIGH, AND HIGH-BOOST VERSIONS OF THE ORIGINAL IMAGE AT THE CENTER OF THAT WINDOW.

INTEGER NPAMP(3,984),NSAMP(3,984),DATA(994),NEW(994)
INTEGER PAMP(5,984),SAMP(5,984),IN,OUT,E,N,LA,LN,F
INTEGER SSFNAME(6),NEWSSF1(6),NEWSSF2(6),NEWSSF3(6)
INTEGER DATAFIL,NEWFIL,NEWFIL2,NEWFIL3,SSFIL
DATA DATAFIL,SSFIL,NEWFIL,NEWFIL2,NEWFIL3/22,24,25,26,27/
READ (22, FMT="(I5,F5.2) ") LA,FK
READ (DATAFIL,21,END=99) SSFNAME,NEWSSF1,NEWSSF2,NEWSSF3
21 FORMAT(24(A3))
CALL ASSIGN(SSFIL,SSFNAME,IERR)
IF (IERR .NE. 0) STOPAS24
BUFFER IN(24,DATA,B,24,E,N)
CALL STATUS (24)
IF (IGENR(NEWSSF1,5,100,2,4000) .GT. 1) STOP GENSSF
CALL ASSIGN(NEWFIL,NEWSSF1,IERR)
IF (IERR .NE. 0) STOPAS25
BUFFER OUT(25,DATA,B,24,E,N)
CALL STATUS (25)
IF (IGENR(NEWSSF2,5,100,2,4000) .GT. 1) STOP GENSSF
CALL ASSIGN(NEWFIL2,NEWSSF2,IERR)
IF (IERR .NE. 0) STOPAS26
BUFFER OUT(26,DATA,B,24,E,N)
CALL STATUS (26)
IF (IGENR(NEWSSF3,5,100,2,4000) .GT. 1) STOP GENSSF
CALL ASSIGN(NEWFIL3,NEWSSF3,IERR)
IF (IERR .NE. 0) STOPAS27
BUFFER OUT(27,DATA,B,24,E,N)
CALL STATUS (27)
K=1
C IF (FLAG .NE. 0) GO TO 15
C
FOR J=1,LA
  BUFFER IN(SSFIL,DATA,B,994,E,N)
  CALL STATUS (SSFIL)
  FOR I=1,984
    PAMP(J,I)=(DATA(I+10) .AND. '377)
    SAMP(J,I)=(DATA(I+10)/256) .AND. '377
  END FOR
END FOR
LN=LA*7
M=1+(LA-1)/2
LOOP
  K=K+1
  WRITE(3, ) " K=",K
  FOR J=1,3
    FOR I=1,984
NPAMP(J,I) = PAMP(M,I)
NSAMP(J,I) = SAMP(M,I)

END FOR
END FOR
IPSUM = 0
ISSUM = 0
FOR J = 1, LA
  FOR I = 1, LN
    IPSUM = IPSUM + PAMP(J,I)
    ISSUM = ISSUM + SAMP(J,I)
  END FOR
END FOR
F = FLOAT(LA*LN)
FOR I = LN - (LN-1)/2, 984 - (LN-1)/2
  NPAMP(1,I) = NINT(IPSUM/F)
  NSAMP(1,I) = NINT(ISSUM/F)
  NPAMP(2,I) = ABS(PAMP(M,I) - NINT(IPSUM/F))
  NSAMP(2,I) = ABS(SAMP(M,I) - NINT(ISSUM/F))
  NPAMP(3,I) = ABS(PAMP(M,I) - NINT(FK*IPSUM/F))
  NSAMP(3,I) = ABS(SAMP(M,I) - NINT(FK*I SSUM/F))
  IP = I + (LN-1)/2
  IM = I - (LN-1)/2
  FOR J = 1, LA
    IPSUM = IPSUM - PAMP(J, IM) + PAMP(J, IP)
    ISSUM = ISSUM - SAMP(J, IM) + SAMP(J, IP)
  END FOR
END FOR
WRITE(3, fmt = "(5I8)") PAMP(M,I), NPAMP(1,I), NPAMP(2,I), NPAMP(3,I) , IPSUM
C
END FOR
FOR J = 1, LA - 1
  FOR I = 1, 984
    PAMP(J,I) = PAMP(J+1,I)
    SAMP(J,I) = SAMP(J+1,I)
  END FOR
END FOR
BUFFER IN (SSFIL, DATA, B, 994, E, N)
CALL STATUS (SSFIL)
EXIT LOOP IF (E .EQ. 3 .OR. E .EQ. 4)
FOR I = 1, 984
  PAMP(LA,I) = (DATA(I+10) .AND. '377)
  SAMP(LA,I) = (DATA(I+10)/256) .AND. '377
END FOR
OUTPUT NEW DATA FILES
C
C
FOR I = 11, 994
  NEW(I) = NPAMP(1,I-10) .AND. '377
  NEW(I) = NEW(I) .OR. (NSAMP(1,I-10) .AND. '377)*256
END FOR
BUFFER OUT (NEWFIL1, NEW, B, 994, E, N)
CALL STATUS (NEWFIL1)
C
FOR I = 11, 994
  NEW(I) = NPAMP(2,I-10) .AND. '377
C
FOR I=11,994
  NEW(I)=NSAMP(3,I-10) .AND. '377
  NEW(I)=NEW(I) .OR. (NSAMP(3,I-10) .AND. '377)*256
END FOR
BUFFER OUT(NEWFIL2,NEW,B,994,E,N)
CALL STATUS(NEWFIL2)
END LOOP

C
FOR I=11,994
  NEW(I)=NSAMP(2,I-10) .AND. '377
  NEW(I)=NEW(I) .OR. (NSAMP(2,I-10) .AND. '377)*256
END FOR
BUFFER OUT(NEWFIL2,NEW,B,994,E,N)
CALL STATUS(NEWFIL2)
END LOOP

99  ENDFILE NEWFIL1
ENDFILE NEWFIL2
ENDFILE NEWFIL3
1000 FORMAT(1H0,I10,5X,I10)
STOP
END
APPENDIX B2

C**********************************************************************************
C PROGRAM HIMEFILT, A FILTER WHICH SUBSTITUTES THE MEDIAN VALUE OF
C WINDOW FOR THE VALUE OF THE PIXEL AT THE CENTER OF THAT WINDOW.
C**********************************************************************************
C
INTEGER NPAMP(984),NSAMP(984),DATA(994),NEW(994),PAMP(984)
INTEGER SAMP(984),IN,OUT,E,N,LA,PHIS(256),SHIS(256),LN,P
DATA IN/10/,OUT/20/
READ(22, FMT="(I5)") LA
BUFFER IN (IN,DATA,B,24,E,N)
CALL STATUS (IN)
BUFFER OUT(OUT,DATA,B,24,E,N)
CALL STATUS (OUT)
NL=(LA-1)/2
NR=(LA-1)/2
LOOP
  BUFFER IN(IN,DATA,B,994,E,N)
  CALL STATUS (IN)
  EXIT LOOP IF (E .LT. 3 .OR. E .EQ. 4)
  FOR I=1,984
    PAMP(I)=1+(DATA(I+10) .AND. '377)
    SAMP(I)=1+(DATA(I+10)/256) .AND. '377
  END FOR
  FOR I=1,256
    PHIS(I)=0
    SHIS(I)=0
  END FOR
  FOR I=1,LA
    PHIS(PAMP(I))=PHIS(PAMP(I))+1
    SHIS(SAMP(I))=SHIS(SAMP(I))+1
  END FOR
  FOR I=NL+1,984-NR
    IPH=NL+1
    ISH=NL+1
    J=1
    WHILE(IPH .GT. 0)
      J=J+1
    END WHILE
    NPAMP(I)=J-1
    J=1
    WHILE(ISH .GT. 0)
      J=J+1
    END WHILE
    NSAMP(I)=J-1
    PHIS(PAMP(I-NL))=PHIS(PAMP(I-NL))-1
    SHIS(SAMP(I-NL))=SHIS(SAMP(I-NL))-1
    PHIS(PAMP(I+NR+1))=PHIS(PAMP(I+NR+1))+1
    SHIS(SAMP(I+NR+1))=SHIS(SAMP(I+NR+1))+1
END FOR
  FOR I=984-NR+1,984
    NPAMP(I)=PAMP(I)-1
    NSAMP(I)=SAMP(I)-1
  END FOR
  FOR I=1,NL
    NPAMP(I)=PAMP(I)-1
    NSAMP(I)=SAMP(I)-1
  END FOR
  FOR I=11,994
    NEW(I)=NPAMP(I-10) .AND. '377
    NEW(I)=NEW(I) .OR. (NSAMP(I-10) .AND. '377)*256
  END FOR
  BUFFER OUT(OUT,NEW,B,994,E,N)
  CALL STATUS(OUT)
END LOOP
ENDFILE OUT
1000 FORMAT(1HO,I10,5X,I10)
STOP
END
C*****************************************************************************
C SUBROUTINE SORT
C*****************************************************************************
SUBROUTINE SORT (AMP, M1, LIMIT)
INTEGER AMP(3), M1, LIMIT, EXCHAN, SAVE, LSAVE
LSAVE=(LIMIT-1)/2 +1
LOOP
  LIMIT=LIMIT-1
  EXCHAN=0
  DO 100 L=1,LIMIT
    IF (AMP(L) .LE. AMP(L+1)) GO TO 100
    EXCHAN=1
    SAVE=AMP(L+1)
    AMP(L+1)=AMP(L)
    AMP(L)=SAVE
  CONTINUE
  IF (EXCHAN .EQ. 0) GO TO 200
END LOOP
200  M1=AMP(LSAVE)
RETURN
END
APPENDIX B3


INTEGER PAMP(984,3), SAMP(984,3), NEW(994), E, N, K
INTEGER NPAMP(984,3), NSAMP(984,3)
INTEGER DATA(994), IN, OUT, IPAR
REAL P

DATA IN/20/, OUT/30/
READ (22, ) IPAR, P
BUFFER IN (IN, DATA, B, 24, E, N)
CALL STATUS (IN)
BUFFER OUT (OUT, DATA, B, 24, E, N)
CALL STATUS (OUT)

READ IN IMAGE

K=0
FOR J=1,3
   K=K+1
   BUFFER IN (IN, DATA, B, 994, E, N)
   CALL STATUS (IN)
   FOR I=1,984
      PAMP(I,J) = (DATA(I+10) .AND. '377)
      SAMP(I,J) = (DATA(I+10)/256) .AND. '377
   END FOR
END FOR
FOR I=11,994
   NEW(I) = PAMP(I-10,1) .AND. '377
   NEW(I) = NEW(I) .OR. (SAMP(I-10,1) .AND. '377)*256
END FOR
BUFFER OUT (OUT, NEW, B, 994, E, N)
CALL STATUS (OUT)

BEGIN READING AND PROCESSING THE DATA

LOOP
   IF (K .EQ. 3) THEN
      CALL SOBEL (PAMP, NPAMP, P, IPAR)
      CALL SOBEL (SAMP, NSAMP, P, IPAR)
      FOR I=11,994
         NEW(I) = NPAMP(I-10,2) .AND. '377
         NEW(I) = NEW(I) .OR. (NSAMP(I-10,2) .AND. '377)*256
      END FOR
      BUFFER OUT (OUT, NEW, B, 994, E, N)
      CALL STATUS (OUT)
      K=K+1
   ELSE
      FOR I=1,984
         PAMP(I,1) = PAMP(I,2)
         SAMP(I,1) = SAMP(I,2)
   END IF
END LOOP
PAMP(I,2)=PAMP(I,3)
SAMP(I,2)=SAMP(I,3)
END FOR
BUFFER IN(IN,DATA,B,994,E,N)
CALL STATUS(IN)
EXIT LOOP IF (E .EQ. 3 .OR. E .EQ. 4)
FOR I=1,984
  PAMP(I,3)=(DATA(I+10) .AND. '377)
  SAMP(I,3)=(DATA(I+10)/256) .AND. '377
END FOR
CALL SOBEL(PAMP,NEAMP,P,IPAR)
CALL SOBEL(SAMP,NSAMP,P,IPAR)
FOR I=11,994
  NEW(I)=NPAMP(I-10,2) .AND. '377
  NEW(I)=NEW(I) .OR. (NSAMP(I-10,2) .AND. '377)*256
END FOR
BUFFER OUT(OUT,NEW,B,994,E,N)
CALL STATUS(OUT)
END IF
END LOOP
STOP
END

C**********************************************************************
cSUBROUTINE SOBEL
c**********************************************************************
SUBROUTINE SOBEL(O,NEW,P,IPAR)
INTEGER O(984,3),NEW(984,3),L(123),N,X,Y,A(123,3),IPAR
REAL P,SUM
FOR I=1,123
  L(I)=0
END FOR
J=2
FOR M=1,3
  FOR I=1,123
    A(I,M)=0
    FOR K=1,8
      A(I,M)=A(I,M)+O((I-1)*8+K,M)
    END FOR
    A(I,M)=NINT(A(I,M)/8.)
    IF(A(I,M) .LT. IPAR) A(I,M)=0
  END FOR
END FOR
FOR I=2,122
  X=(A(I-1,1)+2*A(I,1)+A(I+1,1))-(A(I-1,1)+2*A(I,1)+A(I+1,1))
  Y=(A(I-1,1)+2*A(I,1)+A(I+1,1))-(A(I-1,1)+2*A(I,1)+A(I+1,1))
  SUM= FLOAT(X**2 + Y**2)
  L(I)=NINT(SQRT(SUM))
END FOR
MAX=0
FOR I=1,123
  MAX= MAX0(MAX,L(I))
END FOR
FOR I=1,984
  NEW(I,2)=0
END FOR
FOR I=2,122
  IF(L(I) .GE. P*MAX .AND. A(I,2) .GE. IPAR)
    FOR M=1,8
      NEW((I-1)*8+M,2)=255
    END FOR
  END IF
END FOR
C  CALL REMAP(L,NEW(I,2),984)
RETURN
END

FILE:  "1000SM2*SOBEEDGE"
C**********************************************************************
C REMAPPThG OF DATA INID '!HE RAN3E 1-256 BY LINEAR STRECH
C**********************************************************************
SUBROUTINE REMAP(DEPHT,NEWDEP,M)
  REAL DEPHT(984),DMAX,DMIN,RANGE
  INTEGER NEWDEP(984),M
  DMIN=10000
  DMAX=0.0
  FOR I=1,M
    DMAX=MAX(DMAX,DEPHT(I))
    DMIN=MIN(DMIN,DEPHT(I))
  END FOR
  RANGE=DMAX-DMIN
  FOR I=1,M
    NEWDEP(I)=INT((DEPHT(I)-DMIN)/RANGE*256.)
  END FOR
RETURN
END
APPENDIX C1

C PCA1 PRINCIPAL COMPONENT ANALYSIS 3/16/86 TRIV.
C
C REQUIRES IMSL SCIENTIFIC LIBRARY
INTEGER NBR(6),IX,M,IA,IER
REAL DMAX(10),DMIN(10)
REAL X(24,6),TEMP(6),XM(6),VCV(21),EVAL(6)
REAL EVEC(6,6),COMP(6,6),VAR(6),CL(6),CU(6)
COMMON X
N=24
NBR(1)=6
NBR(2)=N
NBR(3)=N
NBR(4)=1
NBR(5)=1
NBR(6)=0
IX=N
M=6
IA=6
CL(1)=FLOAT(N)
FOR I=1,N
   READ (20,FMT=*(10F8.4)) (X(I,J),J=1,6)
END FOR
FOR J=1,6
   DMAX(J)=X(1,J)
   DMIN(J)=DMAX(J)
   FOR I=2,N
      DMAX(J)=AMAX1(DMAX(J),X(I,J))
      DMIN(J)=AMIN1(DMIN(J),X(I,J))
   END FOR
END FOR
CALL TRANS(M,L,X)
FOR J=1,6
   FOR I=1,N
      X(I,J)=(X(I,J)-DMIN(J))/(DMAX(J)-DMIN(J))
   END FOR
   WRITE (3,FMT=*(10F8.4)) (X(I,J),I=1,6)
END FOR
CALL BECOVM(X,IX,NBR,TEMP,XM,VCV,IER)
WRITE (30,FMT=*(10F8.4)) "VARIANCE-COVARIANCE MATRIX"
FOR I=1,6
   WRITE (30,FMT=*(10F8.4)) (VCV(I*(I-1)/2.+J),J=1,I)
END FOR
WRITE (30,FMT=*(10F8.4)) "MEAN VECTOR"
WRITE (30,FMT=*(10F10.4)) (XM(I),I=1,6)
CALL OPRINC(VCV,M,IA,EVAL,EVEC,COMP,VAR,CL,CU,IER)
WRITE (30,FMT=*(10F8.4)) "10 EIGENVALUES ARE:"
WRITE (30,FMT=*(10F8.4)) (EVAL(I),I=1,6)
WRITE (30,FMT=*(10F8.4)) "10 EIGENVECTORS ARE"
FOR I=1,6
   WRITE (30,FMT=*(10F8.4)) (EVEC(I,J),J=1,6)
END FOR
APPENDIX D1

C**********************************************************************
C 'IBIS IS GLQ.1, A TEXTURE PROGRAM BASED ON GREY LEVEL CO OCCURRENCE
C MATRICES.
C**********************************************************************

INTEGER PAMP(984,12), SAMP(984,12), DATA(994), PNEW(98,12)
INTEGER SNEW(98,12), L, K, IN, E, A, B, C, D, R, IPAR, DATAFIL, FILENAME(6)
INTEGER SSFILE, JKEEP, FLAG, TEXNAME(6), TEXTFIL, TEST, S, FACTOR
REAL STATP(10), STATS(10)

COMMON TEST
DATA DATAFIL/20/
DATA SSFILE, TEXTFIL/24,30/
READ (22,1) IPAR, R, TEST, FACTOR, K, L, M
WRITE (50,1) IPAR, R, TEST, FACTOR, K, L, M

1 FORMAT (7I5)
   JKEEP = 1
   IF (TEST .EQ. 1)
      S = R*FACTOR
   ELSE
      S = R
   END IF
   WRITE (3, 1) " S=" , S
   FLAG = 0
   A = 2*K*(M-R)
   B = 2*M*(K-S)
   C = 2*(M-R)*(K-S)
   D = C

10 READ (DATAFIL, 21, END = 100) FILENAME, TEXNAME

21 FORMAT (12 (A3))
   CALL ASSIGN (SSFILE, FILENAME, IERR)
   IF (IERR .NE. 0) STOP S24
   BUFFER IN(24, DATA, B, 24, E, N)
   CALL STATUS (24)
   IF (IGENR (TEXNAME, 5, 50, 2, 1000) .GT. 1) STOP GENTEX
   CALL ASSIGN (TEXTFIL, TEXNAME, IERR)
   IF (IERR .NE. 0) STOP S25
   IF (FLAG .NE. 0) GO TO 14
   LOOP
   FOR J = 1, M
     BUFFER IN(24, DATA, B, 994, E, N)
     CALL STATUS (24)
     IF (E .EQ. 3 .OR. E .EQ. 4)
       CLOSE TEXTFIL
     FLAG = 1
     JKEEP = J
     GO TO 10
   END IF
   FOR I = 1, 984
     PAMP(I, J) = (DATA(I+10) .AND. '377)
     SAMP(I, J) = (DATA(I+10)/256) .AND. '377
   END FOR
   FLAG = 0
FOR N=1,L
   FOR I=1,K
      FOR J=1,M
         IF (IPAR) 15,16,15
            PNEW(I,J) = PAMP(I+(N-1)*K,J)
            SNEW(I,J) = SAMP(I+(N-1)*K,J)
            GO TO 17
         END IF
         IF (IPAR) 15,16,15
            PNEW(I,J) = INT(PAMP(I+(N-1)*K,J)/16.)+1
            SNEW(I,J) = INT(SAMP(I+(N-1)*K,J)/16.)+1
         END IF
      END FOR
   END FOR
   CALL TEXTURE(PNEW,M,K,STATP,A,B,C,D,R,S,IPAR)
   CALL TEXTURE(SNEW,M,K,STATS,A,B,C,D,R,S,IPAR)
   IF (K.GT.2) TEST=0
   WRITE (30, FMT='="(10(F8.4))") (STATP(I), I=1,10)
   WRITE (30, FMT='="(10(F8.4))") (STATS(J), J=1,10)
   END FOR
END LOOP
STOP

C**********************************************************************
C SUBROUTINE TEXTURE -- A GLCM GENERATOR, MODELLED AFTER HARALICK
AND SHANMUGAN, 1982, IEEE PAMI
C**********************************************************************

SUBROUTINE TEXTURE(M,Y,X,STAT,A,B,C,D,R,S,IPAR)
   INTEGER CMH(16,16), CMV(16,16), CMNE(16,16), CMNW(16,16)
   REAL STAT(10), DATA(4,5), MAX, MIN
   INTEGER M(98,12), X,Y,N,A,B,C,D,E,R, MJ,S,TEST,IPAR,OLD(49,6)
   COMMON TEST
   FOR I=1,10
      STAT(I)=0.
   END FOR
   IF (IPAR .EQ. 1)
      CALL MEAN(M,X,Y)
   END IF
   WRITE(3, ) " S=" , S
   FOR I=1,96
      WRITE(60, FMT='="(12(I6))") (M(I,J), J=1,12)
   END FOR
   FOR R=1,1
      E = 1
      A=2*X* (Y-R)
      B=2*Y* (X-S)
      C=2* (X-S)* (Y-R)
      D=C
      FOR I=1,16
         CMH(I,J)=0
         CMV(I,J)=0
         CMNE(I,J)=0
      END FOR
   END FOR
C**********************************************************************
C SUBROUTINE TEXTURE -- A GLCM GENERATOR, MODELLED AFTER HARALICK
AND SHANMUGAN, 1982, IEEE PAMI
C**********************************************************************

SUBROUTINE TEXTURE(M,Y,X,STAT,A,B,C,D,R,S,IPAR)
   INTEGER CMH(16,16), CMV(16,16), CMNE(16,16), CMNW(16,16)
   REAL STAT(10), DATA(4,5), MAX, MIN
   INTEGER M(98,12), X,Y,N,A,B,C,D,E,R, MJ,S,TEST,IPAR,OLD(49,6)
   COMMON TEST
   FOR I=1,10
      STAT(I)=0.
   END FOR
   IF (IPAR .EQ. 1)
      CALL MEAN(M,X,Y)
   END IF
   WRITE(3, ) " S=" , S
   FOR I=1,96
      WRITE(60, FMT='="(12(I6))") (M(I,J), J=1,12)
   END FOR
   FOR R=1,1
      E = 1
      A=2*X* (Y-R)
      B=2*Y* (X-S)
      C=2* (X-S)* (Y-R)
      D=C
      FOR I=1,16
         CMH(I,J)=0
         CMV(I,J)=0
      END FOR
   END FOR
CMW(I,J)=0
END FOR
END FOR
T=5
FOR I=1,X
FOR J=1,Y
MJ=M(I,J)
IF (J .GT. R) CMH(MIJ,M(I,J-R))=CMH(MIJ,M(I,J-R))+1
IF (J .LE. Y-R) CMH(MIJ,M(I,J+R))=CMH(MIJ,M(I,J+R))+1
IF (I .GT. R) CMV(MIJ,M(I-R,J))=CMV(MIJ,M(I-R,J))+1
IF (I .LE. X-R) CMV(MIJ,M(I+R,J))=CMV(MIJ,M(I+R,J))+1
IF (I .GT. S .AND. J .LE. Y-R) CMNE(MIJ,M(I-S,J+R))=CMNE(MIJ,M(I-S,J+R))+1
IF (I .LE. X-S .AND. J .GT. R) CMNE(MIJ,M(I+S,J-R))=CMNE(MIJ,M(I+S,J-R))+1
CALL FEATURE(CMH,DATA,A,1)
CALL FEATURE(CMV,DATA,B,2)
CALL FEATURE(CMNE,DATA,C,3)
CALL FEATURE(CMW,DATA,D,4)
FOR I=1,5
FOR J=1,4
STAT(I)=STAT(I)+DATA(J,I)
END FOR
END FOR
IF (stat((r-1)*5+1) .NE. 0) stat((r-1)*5+1)=log(stat((r-1)*5+1)
END FOR
FOR I=1,5
STAT(I)=STAT(I)/4.
MAX = DATA(1,I)
MIN = MAX
FOR J=2,4
MAX=AMAX1(MAX,DATA(J,I))
MIN=AMIN1(MIN,DATA(J,I))
END FOR
STAT(5+I) = (MAX-MIN)/2.
END FOR
STAT(6)=0.
FOR J=1,16
FOR I=J,16
STAT(6)=STAT(6)+ABS(CMH(I,J)/FLOAT(A)-CMV(I,J)/FLOAT(B)
END FOR
END FOR
END
INTEGER FUNCTION MM(I,J,K)
MM=MMT(I*.293 + .707*J )
RETURN
END

C**********************************************************************
C   SUBROUTINE FEATURE -- A STATISTICAL FEATURE EXTRACTOR
C**********************************************************************
SUBROUTINE FEATURE(CM,DATA,R,N)
REAL DATA(4,5),IDATA,VAR,MUA,MUB,MUA2,MUB2,SIGMAA,SIGMAB,COR,SUM
COMMON TEST
C IF (TEST .EQ. 1).
C FOR I=1,16
C WRITE (50, FMT="(16(I4))") (O(I,I) ,J=1,16)
C END FOR
C END IF
SUM =0.
NDATA = 0.
MUA = 0.
MUB = 0.
MUA2 = 0.
MUB2 = 0.
MUAB = 0.
FOR I=1,5
   DATA(N,I)=O.O
END FOR
SIGMAA=O.
SIGMAB=O.
VAR=O.
COR=O.
FOR I=1,16
   FOR J=1,16
      IF (CM(I,J) .NE. 0)
         NDATA = CM(I,J)/FLOAT(R)
         DATA(N,1)=((I-J)*((I-J))*NDATA*2.+ DATA(N,1)
         MUA = MUA + I*NDATA*2.
         MUA2 = MUA2 + (I*I)*NDATA*2.
         MUB = MUB + J*NDATA*2.
         MUB2 = MUB2 + (J*J)*NDATA*2.
         MUAB = MUAB + I*J*NDATA*2.
         DATA(N,2)=DATA(N,2)+DATA(N,2)/FLOAT(1+(I-J)*((I-J))
         IF (NDATA .NE. 0.)
            DATA(N,3)=DATA(N,3)-2.*NDATA*LOG(NDATA+0.01)
            END IF
            DATA(N,4)=DATA(N,4)+2.*NDATA*NDATA
            END IF
END IF
SIGMAA = SQRT(ABS(MUA2-(MUA*MUA)/256.))/255.
SIGMAB = SQRT(ABS(MUB2-(MUB*MUB)/256.))/255.
VAR = SIGMAA*S Igmab
COR=(MUAB -((MUA)*(MUB)/256.))/255.
IF (VAR .NE. 0.)
  DATA(N,5)=2*COR/VAR
END IF
RETURN
END

***********************************************************************

SUBROUTINE MEAN(OLD,X,Y)

***********************************************************************

INTEGER OLD(98,12),SUM,AVE,OL,X,Y
SUM=0
AVE=0
FOR I=1,X
  FOR J=1,Y
    SUM=SUM+OLD(I,J)+1
  END FOR
END FOR
AVE=NINT((SUM)/FLOAT(X*Y))
FOR I=1,X
  FOR J=1,Y
    OL= 8+NINT((OLD(I,J)+1-AVE)*.0625)
    IF(OL .GT. 16) OL = 16
    IF(OL .LT. 1 ) OL= 1
    OLD(I,J)=OL
  END FOR
END FOR
RETURN
END
APPENDIX EI

C*********************************************************************
C
C THIS IS PROGRAM CLUSTER, A VERSION OF THE K-MEANS ALGORITHM, FOR
C DETERMINING THE MAXIMUM LIKELIHOOD CLUSTERS OF DATA ON THE BASIS
C OF MINIMIZATION OF EUCLIDIAN DISTANCES OF THE DATA IN A CLUSTER
C FROM THE CLUSTER MEAN(S).
C
C IF THAT MAKES ANY SENSE...
C*********************************************************************

REAL S(25,25), E(25), SMIN, SMAX, DMIN(25), DMAX(25), RAT(25), SD(25,25)
REAL SUM(9000)
INTEGER M, L, N1, N2, IDR, IP, Q(25), P(9000), PAR, PAR2
COMMON X(9000, 10)
SPECIAL COMMON CM
SPECIAL COMMON CM
READ (8, FMT='("9(I5)")') M, L, N1, N2, IDR, IP, PAR, IT, PAR2
WRITE (10, 3) M, L, N1, N2, IDR, IP
3 FORMAT (1H1, ' M=', I6, ' L=', I2, ' N1=', I2, ' N2=', I2, ' IDR=', I1, 
       ' IP=', I1)
WRITE (10, 4)
4 FORMAT (1H1)
FOR J=1, M
   READ (60, FMT='("6F8.4")') (X(J,K), K=1, L)
END FOR
IF (IT .EQ. 0)
   FOR J=1, L
      DMAX(J)=-1000.
      DMIN(J)= 1000.
      FOR I=1, M
         DMAX(J)=AMAX1(DMAX(J), X(I,J))
         DMIN(J)=AMIN1(DMIN(J), X(I,J))
      END FOR
   END FOR
   FOR J=1, L
      FOR I=1, M
         X(I,J) = (X(I,J)-DMIN(J))/(DMAX(J)-DMIN(J))
      END FOR
      WRITE (3, ) " RANGE=" , DMAX(J)-DMIN(J) , " J=" , J
   END FOR
ELSE IF (IT .EQ. 1)
   CALL TRANS (M, L)
ELSE IF (IT .EQ. 2)
   FOR I=1, M
      SUM(I)=0.0
   END FOR
   FOR I=1, M
      FOR J=1, L
         SUM(I)=SUM(I)+X(I,J)
      END FOR
   END FOR
SMIN = SUM(I)
SMAX = SUM(I)
FOR I = 2, M
  SMIN = AMIN1(SMIN, SUM(I))
  SMAX = AMAX1(SMAX, SUM(I))
END FOR
WRITE(3, ) SMIN, SMAX
C FOR I = 1, M
C P(I) = INT(N* (SUM(I) - SMIN) / (SMAX - SMIN)) + 1
C IF (P(I) .GT. N) P(I) = N
C END FOR
C WRITE (10, FMT='(10X,60H)') ( VALUE OF THE STAT, AT EACH TEXEL, )
C *FOR EACH STAT )
C FOR J = 1, M
C WRITE (10, FMT='(1H ,10(F8.4))") (X(J,I),I=1,L)
C END FOR
FOR N = N1, N2
  WRITE (10, 4)
  IF (IP .NE. 0) GO TO 9
  K = 0
  FOR I = 1, M
    K = K + 1
    IF (K .GT. N) K = K - N
    P(I) = K
  END FOR
  CONTINUE
  IF (IP .NE. 0)
    FOR I = 1, M
      P(I) = INT(FLOAT(N) * (SUM(I) - SMIN) / (SMAX - SMIN)) + 1
      IF (P(I) .GT. N) P(I) = N
    END FOR
    END IF
    CALL KMEANS(P, SD, M, L, N, S, E, D, Q, IDR, PAR)
    WRITE (10, 4)
    IF (PAR2 .EQ. 20)
      WRITE (10, FMT='(20I4)") (P(I),I=1,M)
    ELSE
      WRITE (10, FMT='(10I4)") (P(I),I=1,M)
    END IF
    WRITE (10, 4)
    FOR J = 1, N
      FOR K = 1, L
        S(J,K) = S(J,K) * (DMAX(K) - DMIN(K)) + DMIN(K)
        SD(J,K) = SD(J,K) * S(J,K)
      END FOR
      END FOR
    FOR J = 1, N
      WRITE (10, FMT='(10F8.4)") (S(J,K),K=1,L)
      WRITE (10, FMT='(10F8.4)") (SD(J,K),K=1,L)
      WRITE (10, 4)
      END FOR
    WRITE (10, 4)
    WRITE (10, 4)
    WRITE (10, FMT='(1H ,10I8)") (Q(J), J=1,N)
    WRITE (10, 4)
WRITE(10, 
FMT="(1H ,10F8.3)") (E(J),J=1,N),D
DO 99 I=1,N
  IF (Q(I) .NE. 0)
    RAT(I)=E(I)/FLOAT(Q(I))
  END IF
99 CONTINUE
WRITE(10,9)
WRITE (10, FM!"=" (1H ,10F8.5) ") (RAT(I)
I
I=1,N)
C WRITE(10,11)
C 11 FORMAT (lX, 131 (1H-))
END FOR
STOP
END
C**********************************************************************
C SUBROUTINE KMEANS (P,STDV,M,L,N,S,E,D,Q,IDR,PAR)
C**********************************************************************
C THE INITIAL ASSINGMENT OF THE VECTORS X(M,L) TO N CLUSTERS IS
C GIVEN BY THE ARRAY P, WHERE P(I) IS THE CLUSTER NUMBER OF THE
C I-TH VECTOR; THEREFORE, EACH P(I) MUST BE SUCH THAT 1.LE.P(I)
C .LE.N AND FOR EACH J : 1,2,...,N AT LEAST ONE I WITH P(I) = J
C MUST EXIST.
C LET D DENOTE THE SUM OF THE E(J), WHERE E(J) IS THE SUM OF THE
C SQUARES OF THE DISTANCES BETWEEN THE MEMBERS OF THE J-TH CLUSTER
C AND THEIR CENTROIDS.
C THE SUBROUTINE MINIMIZES D AS FAR AS IS POSSIBLE BY REPEATED
C EXCHANGES IF CLUSTER MEMBERS. THE P(I) ARE CORRESPONDINGLY
C MODIFIED WITHOUT, CHANGING THE NUMBER OF CLUSTERS.
C THE SUBROUTINE RETURNS THE VALUES AT THE FINAL CONFIGURATION FOR
C THE CENTROIDS (S(N,L), THE SUMS E(N), AND D.
C IF IDR = 1 THE CURRENT VALUES OF D AND OF THE VECTOR P ARE
C PRINTED AT EACH ITERATION.
C DIMENSION X(M,L), P(M), S(N,L),E(N), Q(N)
REAL MAX,FR
INTEGER Q,R,U,V,W,IP,IDR,M,L,P(9000),FLAG,PAR,NFLAG,KEEPFLAG
COMMON X(9000,10)
SPECIAL COMMON
FLAG=PAR
WHILE (FLAG .GE. PAR)
C WRITE(3,) " INSIDE WHILE"
  KEEPFLAG=FLAG
20 FOR J=1,N
  Q(J)=0
  E(J)=0.
  FOR K=1,L
    SSQR(J,K)=0.
  END FOR
  STDEV(J,K)=0.
\[ S(J,K) = 0. \]
\[ \text{END FOR} \]
\[ \text{END FOR} \]
\[ \text{FOR } I=1,M \]
\[ R=P(I) \]
\[ \text{IF}(R \text{ .LT. 1 .OR. R .GT. N}) \]
\[ \text{WRITE}(3,\) " RETURNING TO MAIN PROGRAM, OR=", R \]
\[ \text{RETURN} \]
\[ \text{END IF} \]
\[ Q(R)=Q(R)+1 \]
\[ \text{FOR } K=1,L \]
\[ S(R,K)=S(R,K)+X(I,K) \]
\[ \text{END FOR} \]
\[ \text{END FOR} \]
\[ \text{FOR } I=1,N \]
\[ \text{WRITE}(3,) " R=",Q(I)," I="I \]
\[ \text{END FOR} \]
\[ \text{FOR } J=1,N \]
\[ R=Q(J) \]
\[ \text{IF}(R \text{ .EQ. 0}) \]
\[ \text{IMAX}=-1000 \]
\[ \text{FOR } I=1,N \]
\[ \text{IMAX}=\max(\text{IMAX},Q(I)) \]
\[ \text{END FOR} \]
\[ \text{FOR } I=1,N \]
\[ \text{IF}(\text{IMAX} \text{ .EQ. Q(I)}) IBIG=I \]
\[ \text{END FOR} \]
\[ Q(J)=Q(IBIG)-Q(IBIG)/2 \]
\[ Q(IBIG)=Q(IBIG)/2 \]
\[ KT=0 \]
\[ \text{FOR } I=1,M \]
\[ \text{IF}(P(I) \text{ .EQ. IBIG}) \]
\[ P(I)=J \]
\[ \text{FOR } A=1,L \]
\[ S(J,A)=S(J,A)+X(I,A) \]
\[ S(IBIG,A)=S(IBIG,A)-X(I,A) \]
\[ \text{END FOR} \]
\[ KT=KT+1 \]
\[ \text{END FOR} \]
\[ \text{EXIT FOR IF(KT .EQ. Q(J))} \]
\[ \text{END FOR} \]
\[ \text{END IF} \]
\[ \text{FOR } I=1,N \]
\[ \text{WRITE}(3,) " R=",Q(I)," I="I \]
\[ \text{END FOR} \]
\[ \text{FOR } J=1,N \]
\[ R=Q(J) \]
\[ F = 1./\text{FLOAT}(R) \]
\[ \text{FOR } K=1,L \]
\[ S(J,K)=S(J,K)*F \]
\[ \text{END FOR} \]
\[ \text{END FOR} \]
C FOR J=1,N
C IF (MAX EQ. VAR(J)) JMAX=J
C END FOR
C FOR K=1,L
C S(JMAX,K)=S(JMAX,K)-STDV(JMAX,K)*0.5
C N=N+1
C NFLAG=0
C S(N,K)=S(JMAX,K)+STDV(JMAX,K)*0.5
C END FOR
C END IF
C CALL MINDIST (S,STDV,M,N,L,P,FLAG)
C WRITE (3, ) "FLAG="FLAG"
C EXIT WHILE IF (FLAG EQ. KEEPFLAG)
C END WHILE
C FOR I=1,N
C Q(I)=0
C END FOR
C FOR I=1,M
C R=P(I)
C Q(R)=Q(R)+1
C F=0.
C FOR K=1,L
C T=S(R,K)-X(I,K)
C STDV(R,K)=STDV(R,K)+T*T
C F=F+T*T
C END FOR
C E(R)=E(R)+F
C END FOR
C D=0.
C FOR I=1,N
C IF (Q(I) EQ. 0)
C WRITE (3, ) "HAH! THE FUCKER'S HERE!, I="I
C Q(I)=1
C END IF
C FR=FLOAT (Q(I))
C FOR J=1,L
C STDV(I,J)=SQRT (STDV(I,J)/FR)
C END FOR
C D=D+E(I)
C END FOR
C IF (IDR EQ. 1) WRITE (10,16) I,D,(P(U),U=1,M)
16 FORMAT (1H ,I4,F12.2,10I3/(17X,10I3))
C RETURN
C*******************************************************************************
SUBROUTINE TRANS(M,L)
*******************************************************************************
COMMON X(9000,10)
SPECIAL COMMON
F=FLOAT (M)
DO 3 K=1,L
. T=0.
. U=0.
. DO 1 I=1,M
\[ V = X(I,K) \]
\[ T = T + V \]
\[ U = U + V^2 \]

1. \text{CONTINUE}

\text{WRITE}(3, \) " F="F

\[ Q = T / F \]

\text{WRITE}(3, \) " U="U," T="T

\[ S = \sqrt{((F-1.) / (U-T*Q))} \]

\text{WRITE}(3, \) " S="S

2. \text{DO } I = 1, M

\[ X(I,K) = S * (X(I,K) - Q) \]

2. \text{CONTINUE}

3. \text{CONTINUE}

\text{RETURN}

*****************************************************************
C PROGRAM MINDIST, A MINIMUM DISTANCE CLASSIFIER.
C THIS PROGRAM TAKES AS INPUTS, THE CENTROIDS OF OPTIMALY DETERMIN
C CLUSTERS, AND TEXT VECTORS OF LENGTH TEN. THE VECTOR IS PLACED
C THAT CLUSTER FOR WHICH THE MEAN SQUARED ERROR IS MINIMUM.
C TRIV 4/22/84
C*****************************************************************

SUBROUTINE MINDIST(CENT,STDV,NROWS,NCLUS,L,S,FLAG)
INTEGRER NROWS,NCLUS,P(9000),S(9000),L,FLAG,R
REAL CENT(25,25),STDV(25,25),VECTR(10)
REAL *12 MIN,SS(25)
COMMON X(9000,10)
SPECIAL COMMON
FLAG=0
FOR M=1,NROWS
  R=S(M)
  FOR I=1,L
    VECTR(I)=X(M,I)
  END FOR
  MIN=10.**6.
  FOR I=1,NCLUS
    SS(I)=0.
    IF(I .NE. R)
      FOR J=1,L
        SS(I)=SS(I)+ (CENT(I,J)-VECTR(J))**2.
      END FOR
      MIN=AMIN1(MIN,SS(I))
    END IF
  END FOR
  IF(MIN .EQ. SS(I)) P(M) =I
END FOR
IF (P(M) .EQ. S(M))
  FLAG=FLAG
ELSE
  S(M)=P(M)
  FLAG=FLAG+1
END IF
C   WRITE(30, FMT="(10I7)") (P(I), I=1,10)
END FOR
C   IF(FLAG .GE. 1) FLAG=1
RETURN
STOP
END
APPENDIX E2

PROGRAM MINDIST, A MINIMUM DISTANCE CLASSIFIER.
THIS PROGRAM TAKES AS INPUTS, THE CENTROIDS OF OPTIMALLY DETERMIN
CLUSTERS, AND TEXEL VECTORS OF LENGTH TEN. THE VECTOR IS PLACED
THAT CLUSTER FOR WHICH THE MEAN SQUARED ERROR IS MINIMUM.
TRIV 4/22/84

INTEGER NSIG,NCLUS,P(100000),DATAFIL,TEXFILE,TEXNAME(6),IERR
DOUBLE PRECISION MIN,SS(12),CENT(12,10),VECTR(10),X(10)
REAL SIGMA(12,10),TSIG(12)
COMMON P
DATA DATAFIL,TEXFILE/24,25/
READ(10, FMT='(2I5)') NCLUS,NSIG
WRITE(30, 31) NCLUS,NSIG
31 FORMAT ( 1H 
      # OF CLUSTERS=",5, # OF STDV=",5)
FOR I=1,NCLUS
  READ (20, FMT=('(10 (FB.5))") (CENT(I,J), J=1,10)
  READ (20, FMT=('(10 (FB.5))") (SIGMA(I,J),J=1,10)
END FOR
FOR J=1,10
  WRITE(30, FMT=('(10(F15.5))") (CENT(I,J), I=1,NCLUS)
END FOR
READ(10, FMT=('(10F8.3")") (X(I), I=1,10)
WRITE(3,) (X(I),I=1,10)
FOR I=1,NCLUS
  TSIG(I)=0.0
  FOR J=1,10
    SIGMA(I,J)=SIGMA(I,J)*CENT(I,J)*X(J)*NSIG
    TSIG(I)=TSIG(I)+SIGMA(I,J)
  END FOR
END FOR
WRITE(3,) (TSIG(I),I=1,NCLUS)
K=0
10 READ (DATAFIL,21,END=99) TEXNAME
21 FORMAT (6 (A3))
CALL ASSIGN (TEXFILE,TEXNAME,IERR)
IF (IERR .NE. 0) GO TO 99
FOR M=1,NROWS
  READ (25, FMT=('(10(F8.4))") (X(M,L),L=1,10)
END FOR
C FOR I=1,NCLUS
C FOR J=1,10
C X(NROWS+I,J)=CENT(I,J)
C END FOR
C END FOR
C CALL TRANS(NROWS+NCLUS,10)
C FOR I=1,NCLUS
C FOR J=1,10
C CENT(I,J)=X(NROWS+I,J)
C END FOR
C END FOR
C FOR M=1,NROWS
LOOP
  
  READ(25, FMT="(10F8.4)", END =10) (VECTR(I),I=1,10)
  
  K=K+1
  
  MIN=10.**6
  
  FOR I=1,NCLUS
  
  SS(I)=0.
  
  FOR J=1,10
  
  SS(I)=SS(I)+((CENT(I,J)-VECTR(J))*X(J))**2
  
  END FOR!

  END FOR!

  MIN=DMIN(MIN,SS(I))

  END FOR!

  WRITE(3,) " MIN="MIN

  FOR I=1, NCLUS
  
  IF(MIN .EQ. SS(I) .AND. MIN .LE. TSIG(I))
  
  P(K)=I
  
  ELSE
  
  P(K)=NCLUS+1
  
  END IF

  END FOR!

  IF(FLAG .EQ. 1)
  
  WRITE(30, FMT="(20I4)") (P(I), I=1,20)
  
  P("

  FLAG=0
  
  ELSE
  
  WRITE(40, FMT="(20I4)") (P(I), I=1,20)
  
  FLAG=1

  END IF

END LOOP
99 CONTINUE

C**********************************************************************
SUBROUTINE TRANS(M,L)
C**********************************************************************
DIMENSION X(50000,10)
COMMON X
F=FLOAT(M)
DO 3 K=1,L
  
  T=0.
  
  U=0.
  
  DO 1 I=1,M
  
  V=X(I,K)
  
  T=T+V
  
  U=U+V*V

1 CONTINUE
. WRITE(3, ) " F=" , F  
. Q=T/F  
. WRITE(3, ) " U=" , U , " T=" , T  
. S=SQRT((F-1.)/(U-T*Q))  
. WRITE(3, ) " S=" , S  
. DO 2 I=1,M  
. X(I,K)=S*(X(I,K)-Q)  
2   CONTINUE  
3   CONTINUE  
RETURN  
END
APPENDIX G1

C**********************************************************************
C   THIS IS REGATA. REGION GROWING AND TEXTURE ANALYSIS.
C**********************************************************************

INTEGER PAMP(984,2),SAMP(984,2),DATA(994),PT(0:63,0:3),K(2048,4)
INTEGER E,R,IPAR,DATAFIL,FILNAME(6)
INTEGER *6 SSUM,PSUM,SSUMSQ,PSUMSQ,S,P
INTEGER SSFILE,JKEEP,FLAG,TEXNAME(6),TEXFIL,TEST,FACTOR,ICMCT
INTEGER ST(0:63,0:3),ICS(61,2)
REAL PMU(61,2),PVAR(61,2),SMU(61,2),SVAR(61,2),CMU(9048),CKT(9048)
REAL STAT(10),DUDA(4,5),DIF(4),MIN
INTEGER CMH,CWH,CWNE,CWMW,IC(61,2),CSTAT(0:9048),KTCM,IBF(128)
LOGICAL SIMTEST
COMMON /SHIT/ CMH,CKT
SPECIAL COMMON SHIT
COMMON /CM/ CMH(128,16,16),CWH(128,16,16),CWNE(128,16,16),CWMW(128,16,16),
*CMNE(128,16,16),CSTAT
SPECIAL COMMON CM
COMMON /PARS/ TT,FS
DATA DATAFIL,SSFILE,TEXFIL/22,24,30/
READ (20,1) IPAR,R,TL,T,T,FS,NUM
WRITE (3,1) IPAR,R,TL,T,T,FS,NUM
1 FORMAT(2I5,3F5.2,I5)
FLAG=0
IFLAG=1
CSTAT (0)=1
   FOR J=0,3
       FOR I=0,63
           PT(I,J)=0
           ST(I,J)=0
       END FOR
   END FOR
C TL=0.25
10 READ (DATAFIL,21,END=99) FILNAME,TEXNAME
   WRITE (3,21) FILNAME,TEXNAME
21 FORMAT(12(A3))
   CALL ASSIGN(SSFILE,FILNAME,IERR)
   WRITE (3, ) " ierr=" ,IERR
   IF (IERR .NE. 0) STOP AS24
   BUFFER IN(24,DATA,B,24,E,N)
   CALL STATUS (24)
   IF (IGENR(TEXNAME,5,50,2,1000) .GT. 1) STOP GENTEX
   CALL ASSIGN(TEXFIL,TEXNAME,IERR)
   IF (IERR .NE. 0) STOP AS25
   IF (FLAG .NE. 0) GO TO 15
   LOOP
       FOR J=1,2
           BUFFER IN(24,DATA,B,994,E,N)
           CALL STATUS (24)
           IF (E .EQ. 3 .OR. E .EQ. 4)
           CLOSE TEXFIL
CLOSE 24
FLAG=1
JKEEP=J
GO TO 10
END IF
FOR I=1,984
PAMP(I,J) = (DATA(I+10) .AND. '377)
PAMP(I,J)=1+INT((PAMP(I,J))*.0625)
SAMP(I,J) = (DATA(I+10)/256) .AND. '377
SAMP(I,J)=1+INT((SAMP(I,J))*.0625)
END FOR
FLAG=0
END FOR
MIN=1000
MAX=1
CALCULATE STATS FOR EACH 2 BY 16 CELL
FOR I=1,61
PSUM=0.0
PSUMSQ=0.0
SSUM=0.0
SSUMSQ=0.0
FOR J=1,2
FOR K=1,16
S= SAMP(K+16*(I-1),J)+1
P= PAMP(K+16*(I-1),J)+1
SSUM=SSUM+S
SSUMSQ=SSUMSQ+S*S
PSUM=PSUM+P
PSUMSQ=PSUMSQ+P*P
END FOR
END FOR
CALC MEAN AND VARIANCE
SMU(I,2)=SSUM/32.
SVAR(I,2)=(SSUMSQ-(SSUM*SSUM)/32.)/31.
FMU(I,2)=PSUM/32.
FVAR(I,2)=(PSUMSQ-(PSUM*PSUM)/32.)/31.
HOMOGENEITY TEST
IF(FMU(I,2) .EQ. 0) WRITE(3, ) " PMU=",0
IF(SQRT(Abs(FVAR(I,2))/FMU(I,2)) .GT. TI)
boundary cell
PT(I,2)=0
ICP(I,2)=0
GO TO 81
END IF
IF(FVAR(I,2) .EQ. 0)
PT(I,2)=0
GO TO 81
END IF
NOT A BOUNDARY CELL; CLASSIFY AS EITHER A NEW CLASS OR SIM TO NEIGHBOR
MIN=1000.
FOR J=1,4
  DIF(J)=10000.
END FOR
IF (PT(I-1,1) .NE. 0)
  DIF(1)=ABS(CMU(PT(I-1,1))-PMU(I,2))
  MIN=AMIN1(MIN,DIF(1))
END IF
IF (PT(I,1) .NE. 0)
  DIF(2)=ABS(CMU(PT(I,1))-PMU(I,2))
  MIN=AMIN1(MIN,DIF(2))
END IF
IF (PT(I+1,1) .NE. 0)
  DIF(3)=ABS(CMU(PT(I+1,1))-PMU(I,2))
  MIN=AMIN1(MIN,DIF(3))
END IF
IF (PT(I-1,2) .NE. 0)
  DIF(4)=ABS(CMU(PT(I-1,2))-PMU(I,2))
  MIN=AMIN1(MIN,DIF(4))
END IF
IF (MIN .LE. TT)
  FOR L=1,4
    IF (MIN .EQ. DIF(L))
      IKEEP=L
    EXIT FOR
    END IF
  END FOR
  J=INT((IKEEP-1)/3)+1
  K=IKEEP-(J)*3+1
  PT(I,2)=PT(I+K,J)
  ICP(I,2)=ICP(I+K,J)
  CMU(PT(I,2))=(CMU(PT(I,2))*CKT(PT(I,2))+PMU(I,2))
  CKT(PT(I,2))=CKT(PT(I,2))+1.
  GO TO 80
ELSE
  NCLASS=NCLASS+1
  PT(I,2)=NCLASS
C WRITE(3,') " NOT SIM TO ANY, I="I," NCLASS="NCLASS
C WRITE(3,') " INCOMING NCLASS,="NCLASS
  IF (KTCM .NE. 0)
C WRITE(3,') " USING BURNED CM, KTCM="KTCM,"BURN FILE="IBF(KTCM)
    ICP(I,2)=IBF(KTCM)
    KTCM=KTCM-1
  ELSE
    NCM=NCM+1
    ICP(I,2)=NCM
  END IF
  CMU(PT(I,2))=(CMU(PT(I,2))*CKT(PT(I,2))+PMU(I,2))
  CKT(PT(I,2))=CKT(PT(I,2))+1.
END IF
80 CONTINUE
C R=7
L=16*(I-1)

ICMCT=ICP(I,2)

CALL MEAN(PAMP,PMU(I,2),PVAR(I,2),L)

FOR K=1,2
  FOR J1=R+1,16
    J=J1+L
    MIJ=PAMP(J,K)
    WRITE(3, ) ICMCT,MIJ,PAMP(J-R,K)
    CMH(ICMCT,MIJ,PAMP(J-R,K))=CMH(ICMCT,MIJ,PAMP(J-R,K))
    CSMCI(ICMCT,PAMP(J-R,K),MIJ)=CSMCI(ICMCT,PAMP(J-R,K),MIJ)
  END FOR
  KT(ICMCT,1)=KT(ICMCT,1)+36
  FOR J1=1,16
    J=J1+L
    MIJ=PAMP(J,1)
    CMV(ICMCT,MIJ,PAMP(J,2))=CMV(ICMCT,MIJ,PAMP(J,2))+1
    CMV(ICMCT,PAMP(J,2),MIJ)=CMV(ICMCT,PAMP(J,2),MIJ)+1
  END FOR
  KT(ICMCT,2)=KT(ICMCT,2)+32
  FOR J1=1,16-R
    J=J1+L
    MIJ=PAMP(J,1)
    CMNE(ICMCT,MIJ,PAMP(J+R,1))=CMNE(ICMCT,MIJ,PAMP(J+R,1))
    CMNE(ICMCT,PAMP(J+R,1),MIJ)=CMNE(ICMCT,PAMP(J+R,1),MIJ)
  END FOR
  KT(ICMCT,3)=KT(ICMCT,3)+18
  KT(ICMCT,4)=KT(ICMCT,4)+18

START STBD PROCESSING

HOMOGENEITY TEST

IF(SMU(I,2) .EQ. 0) WRITE(3, ) "SMU="','
IF(SQRT(ABS(SVAR(I,2))) /SMU(I,2) .GT. Tl)

boundary cell
  ST(I,2)=0
 ICS(I,2)=0
  GO TO 90
  END IF
  IF(SVAR(I,2) .EQ. 0)
    ST(i,2)=0
    GO TO 90
  END IF

NOT A BOUNDARY CELL; CLASSIFY AS EITHER A NEW CLASS OR SIM TO NEIGHBOR

MIN=1000.
  FOR J=1,4
    DIF(J)=10000.
  END FOR
IF (ST(I-1,1) .NE. 0)
  DIF(1)=ABS(CMU(ST(I-1,1))-SMU(I,2))
  MIN=AMIN(MIN,DIF(1))
END IF

IF (ST(I,1) .NE. 0)
  DIF(2)=ABS(CMU(ST(I,1))-SMU(I,2))
  MIN=AMIN(MIN,DIF(2))
END IF

IF (ST(I+1,1) .NE. 0)
  DIF(3)=ABS(CMU(ST(I+1,1))-SMU(I,2))
  MIN=AMIN(MIN,DIF(3))
END IF

IF (ST(I-1,2) .NE. 0)
  DIF(4)=ABS(CMU(ST(I-1,2))-SMU(I,2))
  MIN=AMIN(MIN,DIF(4))
END IF

IF (MIN .LE. TT)
  FOR L=1,4
    IF (MIN .EQ. DIF(L))
      IKEEP=L
      EXIT FOR
    END IF
  END FOR
  J=INT((IKEEP-1)/3)+1
  K=IKEEP-(J)*3+1
  ST(I,2)=ST(I+K,J)
  ICS(I,2)=ICS(I+K,J)
  CMU(ST(I,2))=(CMU(ST(I,2))*CKT(ST(I,2))+SMU(I,2))
                /(CKT(ST(I,2))+1.)
  CSTAT(ST(I,2))=1
  CKT(ST(I,2))=CKT(ST(I,2))+1.
  TO 82
ELSE
  NCLASS=NCLASS+1
  ST(I,2)=NCLASS
  WRITE(3) " NOT SIM TO ANY S, I=" ,I," NCLASS=" ,NCLASS
  WRITE(3) " INCOMPLETED NCLASS,=" ,NCLASS
  IF (KTCM .NE. 0)
    WRITE(3) " USING BURNED CM, KTCM=" ,KTCM," BURN FILE=" ,IBF(KTCM)
    ICS(I,2)=IBF(KTCM)
    KTCM=KTCM-1
  ELSE
    NCM=NCM+1
  END IF
  CMU(ST(I,2))=(CMU(ST(I,2))*CKT(ST(I,2))+SMU(I,2))
                /(CKT(ST(I,2))+1.)
  CTKT(ST(I,2))=CKT(ST(I,2))+1.
END IF

82  CONTINUE
R=7
L=16*(I-1)
ICMCT=ICS(I,2)
CALL MEAN(SAMP,SMU(I,2),SVAR(I,2),L)
FOR k=1,2
  FOR J1=1+1,16
    J=J1+L
    MIJ=SAMP(J,K)
    CMH(ICMCT,MIJ,SAMP(J-R,K))=CMH(ICMCT,MIJ,SAMP(J-R,K))
    CMH(ICMCT,SAMP(J-R,K),MIJ)=CMH(ICMCT,SAMP(J-R,K),MIJ)
  END FOR
END FOR
KT(ICMCT,1)=KT(ICMCT,1)+36
FOR J1=1,16
  J=J1+L
  MIJ=SAMP(J,1)
  CMV(ICMCT,MIJ,SAMP(J,2))=CMV(ICMCT,MIJ,SAMP(J,2))+1
  CMV(ICMCT,SAMP(J,2),MIJ)=CMV(ICMCT,SAMP(J,2),MIJ)+1
END FOR
KT(ICMCT,2)=KT(ICMCT,2)+32
FOR J1=1,16-R
  J=J1+L
  MIJ=SAMP(J,1)
  CMNW(ICMCT,MIJ,SAMP(J+R,2))=CMNW(ICMCT,MIJ,SAMP(J+R,2)
  CMNW(ICMCT,SAMP(J+R,2),MIJ)=CMNW(ICMCT,SAMP(J+R,2),MIJ
  MIJ=SAMP(J,2)
  CMNE(ICMCT,MIJ,SAMP(J+R,1))=CMNE(ICMCT,MIJ,SAMP(J+R,1)
  CMNE(ICMCT,SAMP(J+R,1),MIJ)=CMNE(ICMCT,SAMP(J+R,1),MIJ
END FOR
KT(ICMCT,3)=KT(ICMCT,3)+18
KT(ICMCT,4)=KT(ICMCT,4)+18
END FOR

C TWO SINGS SPROCESSE; NOW OUTSUT AND RE-NMAE COUNTERS AND STATS
C
C PORT SECTION
C
FOR I=1,61
  IF (CSTAT(PT(I,1)) .EQ. 0 .AND. IFLAG .NE. 1)
    IF(CKT(PT(I,1)) .GE. NUM)
      IM=ICP(I,1)
      WRITE(3, ) IM FILE CLOSED PT="PT(I,1), "IM="IM
      CALL FEATURE(CMH,DUDA,KT(IM,1),1,IM)
      CALL FEATURE(CMV,DUDA,KT(IM,2),2,IM)
      CALL FEATURE(CMNW,DUDA,KT(IM,3),3,IM)
      CALL FEATURE(CMNE,DUDA,KT(IM,4),4,IM)
    FOR J=1,10
      STAT(J)=0.
    END FOR
    FOR K=1,5
      FOR J=1,4
        STAT(K) = STAT(K)+DUDA(J,K)
      END FOR
    END FOR
    FOR K=1,5
      STAT(K)=STAT(K)/4.
    END FOR
  C RMAX = DUDA(1,K)
  C RMIN =RMAX
FOR J = 2, 4
RMAX = AMAX1 (RMAX, DUDA(J,K))
RMIN = AMIN1 (RMIN, DUDA(J,K))
END FOR
STAT(S+K) = (RMAX - RMIN)
END FOR
FOR J = 1, 16
FOR K = 1, 16
STAT(J) = STAT(J) + ABS(QH(IM,J,K)/FLOAT(KT(IM,1)) - CMV(IM,J,K)/FLOAT(KT(IM,2)))
END FOR
END FOR
WRITE (35, FMT= "(10F8.4)") (STAT(K), K = 1, 10)
WRITE (40, FMT= "(10, 2F12.2)") PT(IM, 1), CMV(PT(IM, 1)), KT
FOR J = 1, 16
FOR K = 1, 16
CMH(IM, J, K) = 0
CMV(IM, J, K) = 0
CMNE(IM, J, K) = 0
CMNW(IM, J, K) = 0
END FOR
END FOR
END FOR
END FOR
IF (CSTAT(ST(I, 1)) .GT. 1) .AND. IFLAG .NE. 1)
IM = ICS(I, 1)
FOR J = 1, 16
FOR K = 1, 16
CMH(IM, J, K) = 0
CMV(IM, J, K) = 0
CMNE(IM, J, K) = 0
CMNW(IM, J, K) = 0
END FOR
END FOR
END FOR
IF (CSTAT(ST(I, 1)) .EQ. 0 .AND. IFLAG .NE. 1)
IM = ICS(I, 1)
WRITE (3, ) " BURNED A FILE, ", IM, " KT= ", KT
ELSE
IM = ICP(I, 1)
FOR J = 1, 16
FOR K = 1, 16
CMH(IM, J, K) = 0
CMV(IM, J, K) = 0
CMNE(IM, J, K) = 0
CMNW(IM, J, K) = 0
END FOR
END FOR
END FOR
WRITE (3, ) " BURNED A FILE, ", IM, " KT= ", KT
END IF
WRITE (3, ) " BURNED A FILE, ", IM, " KT= ", KT
END IF
STBD OUTPUT SECTION
IF (CSTAT(ST(I, 1)) .EQ. 0 .AND. IFLAG .NE. 1)
IF (CKT(ST(I, 1)) .GT. NUM)
IM = ICS(I, 1)
WRITE(3, ) " FILE CLOSED CT="ST(I,1)," IM=" IM

CALL FEATURE(CMH,DUDA,KT(IM,1),1,IM)
CALL FEATURE(CMV,DUDA,KT(IM,2),2,IM)
CALL FEATURE(CMNW,DUDA,KT(IM,3),3,IM)
CALL FEATURE(CMNE,DUDA,KT(IM,4),4,IM)
FOR J=1,10
STAT(J)=0.
END FOR
FOR K=1,5
FOR J=1,4
STAT(K)=STAT(K)+DUDA(J,K)
END FOR
END FOR
FOR K=1,5
STAT(K)=STAT(K)/4.
RMAX = DUDA(1,K)
RMIN =RMAX
FOR J=2,4
RMAX=?MAX1(RMAX,DUDA(J,K))
RMIN=M?MIN1(RMIN,DUDA(J,K))
END FOR
STAT(5+K) = (RMAX-RMIN)
FOR J=1,16
FOR K=1,16
CMH(IM,J,K)=0
CMV(IM,J,K)=0
CMNE(IM,J,K)=0
SUM(IM,J,K)=0
END FOR
END FOR
FOR J=1,4
KT(IM,J)=0
END FOR
CSTAT(ST(I,1))=-1
KTCM=KTCM+1
IBF(KTCM)=IM
WRITE(3, ) " BURNED A FILE," IM, " KTCM=" KTCM
ELSE
IM=ICS(I,1)
FOR J=1,16
FOR K=1,16
CMH(IM,J,K)=0
CMV(IM,J,K)=0
CMNE(IM,J,K)=0
CMNW(IM,J,K)=0
ENDIF
\begin{verbatim}
    . . . .\ END FOR  
    . . . .\ END FOR  
    . . . .\ FOR J=1,4  
    . . . .\ KT(IM,J)=0  
    . . . .\ END FOR  
    . . . .\ CSTAT(ST(I,1))=-1  
    . . . .\ KTCPM=KTCPM+1  
    . . . .\ IBF(KTCPM)=IM  
    C WRITE(3, )" BURNED A FILE",IM," KTCPM="KTCPM  
    . . . .\ END IF  
    . . . .\ END IF  
    . . . .\ END FOR  
    . . . .\ IFLAG=2  
    . . . .\ FOR I=1,61  
    . . . .\ PT(I,1)=PT(I,2)  
    . . . .\ ICP(I,1)=ICP(I,2)  
    . . . .\ PMU(I,1)=PMU(I,2)  
    . . . .\ PVAR(I,1)=PVAR(I,2)  
    . . . .\ ST(I,1)=ST(I,2)  
    . . . .\ ICS(I,1)=ICS(I,2)  
    . . . .\ SMU(I,1)=SMU(I,2)  
    . . . .\ SVAR(I,1)=SVAR(I,2)  
    . . . .\ END FOR  
    . . . .\ FOR I=1,NCLASS  
    . . . .\ IF(CSTAT(I) .NE. 0) CSTAT(I)=0  
    . . . .\ END FOR  
    . . . .\ WRITE(30,FMT="(31I4)") (PT(I,1), I=1,61)  
    C WRITE(9,FMT="(61I2)") (ICP(I,1), I=1,61)  
    C WRITE(30,FMT="(31I4)") (ST(I,1), I=1,61)  
    C WRITE(9,FMT="(61I2)") (ICS(I,1), I=1,61)  
    \end{verbatim}
C
RMIN=AMINI (RMIN,DUDA(J,K))
C END FOR
C STAT(5+K) = (RMAX-RMIN)
   . . END FOR
   . . FOR J=1,16
   . . FOR K=1,16
   . . . STAT(6)=STAT(6)+ABS(CMH(IM,J,K)/FLOAT(KT(IM,1)))
   . . . -CMV(IM,J,K)/FLOAT(KT(IM,2)))
   . . . END FOR
   . . END FOR
   . WRITE(35,FMT="(10F8.4)") (STAT(K) ,K=1,10)
   . WRITE(40,FMT="(110,2F12.2)") PT(I,1),CMJ(PT(I,1)),CKT(PT
   . . CSTAT(PT(I,1))=-1
   . END IF
   . END IF

C
C STEP CLEANUP
C IF (CSTAT(ST(I,1)) .EQ. 0 )
   . IF (CKT(ST(I,1)) .GT. NUM)
      . IM=ICS(I,1)
      . CALL FEATURE(CMH,DUDA,KT(IM,1),1,IM)
      . CALL FEATURE(CMV,DUDA,KT(IM,2),2,IM)
      . CALL FEATURE(CMW,DUDA,KT(IM,3),3,IM)
      . CALL FEATURE(CMNE,DUDA,KT(IM,4),4,IM)
      . FOR J=1,10
      . . STAT(J)=0.
      . END FOR
      . FOR K=1,5
      . . FOR J=1,4
      . . . STAT(K) = STAT(K)+DUDA(J,K)
      . . . END FOR
      . . END FOR
      . FOR K=1,5
      . . STAT(K)=STAT(K)/4.*
C RMAX=DUDA(1,K)
C RMIN=RMAX
C FOR J=2,4
   . RMAX=AMAX1 (RMAX,DUDA(J,K))
   . RMIN=AMINI (RMIN,DUDA(J,K))
C END FOR
C STAT(5+K) = (RMAX-RMIN)
   . . END FOR
   . . FOR J=1,16
   . . FOR K=1,16
   . . . STAT(6)=STAT(6)+ABS(CMH(IM,J,K)/FLOAT(KT(IM,1)))
   . . . -CMV(IM,J,K)/FLOAT(KT(IM,2)))
   . . . END FOR
   . . END FOR
   . WRITE(35,FMT="(10F8.4)") (STAT(K) ,K=1,10)
   . WRITE(40,FMT="(110,2F12.2)") ST(I,1),CMJ(ST(I,1)),CKT(ST
   . . CSTAT(ST(I,1))=-1
   . END IF
   . END IF
C END OF CLEAN-UP
C END FOR
FOR I=1,NCLASS
  WRITE(3, FMT=*(I10,2F12.2)) I,CMI(I),CKT(I)
END FOR
STOP
END

LOGICAL FUNCTION SIMTEST(MU1,MU2,VAR1,VAR2)
  REAL MU1, MU2,VAR1,VAR2,VARMAX,VARMIN,F,SP,T
  COMMON /PARS/ TT, FS
C TEST EQUALITY OF VARIANCES
  IF(FLAG .LT. 1)
    WRITE(3, ) TT,FS
  END IF
  FLAG=FLAG+1
  VARMAX=AMAX1(VAR1,VAR2)
  VARMIN=AMIN1(VAR1,VAR2)
  F=VARMAX/VARMIN
  IF(F .LT. FS )
    SIMTEST = .TRUE.
  ELSE
    SIMTEST = .FALSE.
    RETURN
  END IF
C TEST OF EQUALITY OF MEANS
  SP=(VAR1 + VAR2)*0.5
  T=(MU1-MU2)*4./SQRT(SP)
  IF(ABS(T) .GT. TT ) SIMTEST = .FALSE.
  RETURN
END

SUBROUTINE FEATURE -- A STATISTICAL FEATURE EXTRACTOR

SUBROUTINE FEATURE(CM,DATA,R,N,IX)
  REAL DATA(4,5),NDATA,VAR,MUA,MUB,MUA2,MUB2,SIGMAA,SIGMAB,COR,SUM
  INTEGER CM(128,16,16),R,N,TEST,S,IX
  COMMON TEST
  SUM =0.
  NDATA = 0.
  MUA = 0.
  MUB = 0.
  MUA2 = 0.
  MUB2 = 0.
  MUAB = 0.
  FOR I=1,5
    DATA(N,I)=0.0
  END FOR
  SIGMAA=0.
  SIGMAB=0.
  VAR=0.
mR = 0.
FOR I = 1, 16
  FOR J = 1, 16
    IF (CM(IX, I, J) = 0) GO TO 10
    NDATA = CM(IX, I, J) / FLOAT(R)
    DATA(N, 1) = ((I - J) * (I - J)) * NDATA + DATA(N, 1)
    MJA = MJA + I * NDATA
    MJA2 = MJA2 + (I * I) * NDATA
    MUB = MUB + J * NDATA
    MUB2 = MUB2 + (J * J) * NDATA
    MUAB = MUAB + I * J * NDATA
    DATA(N, 2) = DATA(N, 2) + NDATA / ((1 + (I - J) * (I - J))
    IF (NDATA /= 0.)
      DATA(N, 3) = DATA(N, 3) - NDATA * LOG(NDATA + 1.)
    END IF
    DATA(N, 4) = DATA(N, 4) + NDATA * NDATA
  END FOR
END FOR
SIGMAA = SQRT(ABS(MUA2 - (MJA * MJA) / 256.)) / 256.
SIGMAB = SQRT(ABS(MUB2 - (MUB * MUB) / 256.)) / 256.
VAR = SIGMAA * SIGMAB
COR = (MUAB - ((MUA) * (MUB)) / 256.) / 256.
IF (VAR /= 0.)
  DATA(N, 5) = COR / VAR
END IF
RETURN
END

**********************************************************************
SUBROUTINE TEXTURE -- A GLCM GENERATOR, MODELED AFTER HARRALICK
AND SHANMUGAN, 1982, IEEE PAMI
**********************************************************************
SUBROUTINE TEXTURE (M, Y, X, STAT, A, B, C, D, R, S)
  INTEGER CMH(16, 16), CMV(16, 16), CMNE(16, 16), CMNW(16, 16)
  REAL STAT(10), DATA(4, 5), MAX, MIN
  INTEGER M(49, 6), X, Y, N, A, B, C, D, E, R, MLJ, S, TEST
  COMMON TEST
  WRITE (3, ) " S=" , S
  FOR I = 1, 96
    WRITE (60, FMT = "(12(I6))") (M(I, J), J = 1, 12)
  END FOR
  E = 1
  FOR I = 1, 16
    FOR J = 1, 16
      CMH(I, J) = 0
      CMV(I, J) = 0
      CMNE(I, J) = 0
      CMNW(I, J) = 0
    END FOR
  END FOR
  T = 5
  FOR I = 1, X
    FOR J = 1, Y
      MLJ = M(I, J)
IF (J .GT. R) CMH(MIJ,M(I,J-R)) = CMH(MIJ,M(I,J-R)) + 1
IF (J .LE. Y-R) CMV(MIJ,M(I,J+R)) = CMV(MIJ,M(I,J+R)) + 1
IF (I .GT. S) CMV(MIJ,M(I-S,J)) = CMV(MIJ,M(I-S,J)) + 1
IF (I .LE. X-S) CMi(MIJ,M(I+S,J)) = CMi(MIJ,M(I+S,J)) + 1

IF (I .GT. S .AND. J .LE. Y-R)
   CMNE(MIJ,M(I-T,J+R)) = CMNE(MIJ,M(I-T,J+R)) + 1
   CMNE(MIJ,MM(MIJ,M(I-S,J+R),S)) = CMNE(MIJ,MM(MIJ,M(I-S,J+R),S)) + 1
   CMNW(MIJ,M(I+T,J-R)) = CMNW(MIJ,M(I+T,J-R)) + 1
   CMNW(MIJ,MM(MIJ,M(I+T,J-R),S)) = CMNW(MIJ,MM(MIJ,M(I+T,J-R),S)) + 1
END FOR
END FOR

CALL FEATUR(CMH,DATA,A,1)
CALL FEATUR(CMV,DATA,B,2)
CALL FEATUR(CMNE,DATA,C,3)
CALL FEATUR(CMNW,DATA,D,4)
FOR I=1,10
   STAT(I) = 0.
END FOR
FOR I=1,5
   FOR J=1,4
      STAT(I) = STAT(I) + DATA(J,I)
   END FOR
END FOR
FOR I=1,5
   STAT(I) = STAT(I)/4.
   MAX = DATA(1,I)
   MIN = MAX
   FOR J=2,4
      MAX = AMAX1(MAX,DATA(J,I))
      MIN = AMIN1(MIN,DATA(J,I))
   END FOR
   STAT(5+I) = (MAX-MIN)/2.
END FOR
RETURN

INTEGER FUNCTION MM(I,J,K)
MM = NINT(I*.293 + .707*J)
RETURN

C********************************************************************************
SUBROUTINE FEATURE -- A STATISTICAL FEATURE EXTRACTOR
********************************************************************************

SUBROUTINE FEATURE(CM,DATA,R,N)
REAL DATA(4,5),NDATA,VAR,MUA,MUB,MUA2,MUB2,SIGMAA,SIGMAB,COR,SM
INTEGER CM(16,16),R,N,TEST,S
COMMON TEST
C
IF (TEST .EQ. 1)
FOR I=1,16
   WRITE (50, FMT='"(16(I4))"') (CM(I,J), J=1,16)
END FOR
C
END IF
SUM = 0.
NDATA = 0.
MJA = 0.
MUB = 0.
MUA2 = 0.
MUB2 = 0.
MUAB = 0.
FOR I=1,5
   DATA(N,I)=0.0
END FOR
SIGMAA=0.
SIGMAB=0.
VAR=0.
COR=0.
FOR I=1,16
   FOR J=1,16
      NDATA = CM(I,J)/FLOAT(R)
      DATA(N,1)=((I-J)*(I-J))*NDATA+DATA(N,1)
      MJA = MJA + I*NDATA
      MJA2 = MJA2 + (I*I)*NDATA
      MUB = MUB + J*NDATA
      MUB2 = MUB2 + (J*J)*NDATA
      MUAB = MUAB + I*J*NDATA
      DATA(N,2)=DATA(N,2)+NDATA/(1+(I-J)*(I-J))
      IF (NDATA .NE. 0.)
         DATA(N,3)=DATA(N,3)-NDATA*LOG(NDATA+1.)
      END IF
      DATA(N,4)=DATA(N,4)+NDATA*NDATA
   END FOR
   SIGMAA = SQRT(ABS(MJA*MJA)/256.)/255.
   SIGMAB = SQRT(ABS(MUB2-(MUB*MUB))/256.)/255.
   VAR = SIGMAA*SIGMAB
   COR= (MUAB - ((MJA)*(MUB))/256.))/255.
   IF (VAR .NE. 0.)
      DATA(N,5)= COR/VAR
   END IF
RETURN
END

C
SUBROUTINE MEAN(OL, MU, VAR, L)
C
C
INTEGER OL(984,2), SUM, AVE, OL
REAL MU, VAR
C
S=1.
C
IF (VAR .NE. 0.) S=SQRT(1./VAR)
FOR I=1,16
  FOR J=1,2
    OL = 8+0.0625*(OLD(I+L,J)-MU)
    WRITE(3, ) OLD(I+L,J),OL,L
    IF(OL .GT. 16) OL = 16
    IF(OL .LT. 1 ) OL= 1
    OLD(I+L,J)=OL
  END FOR
END FOR
RETURN
END
APENDIX G2

C*********************************************************************
C THIS IS PROGRAM CLUSTERE A VERSION OF THE K-MEANS ALGORITHM, FOR
C DETERMINING THE MAXIMUM LIKELIHOOD CLUSTERS OF DATA ON THE BASIS
C OF MINIMIZATION OF EUCLIDIAN DISTANCES OF THE DATA IN A CLUSTER
C FROM THE CLUSTER MEAN(S).
C
C IF THAT MAKES ANY SENSE...
C
C MODIFIED 14-9-86 FOR ECHO CLASSES
C
C*********************************************************************

REAL S(25,10),DMIN(10),DMAX(10)
REAL E(25),RAT(25)
REAL RCT(9900),CMJ(9900),SUM(9900),X(9900,10),SD(25,10)
INTEGER P(9900),M,L,NL,N2,IDR,IP,Q(25),IT,IX(9900),PAR
COMMON /CM/ RCT,CMJ,SUM
COMMON X
SPECIAL COMMON
SPECIAL COMMON CM
READ(8, FMT="(8(I5))") M,L,NL,N2,IDR,IP,IT,PAR
WRITE (10,3) M,L,NL,N2,IDR,IP,IT,PAR
3 FORMAT (1H1, 'M=',M,'I6, 'L=',L,'I2, 'NL=',NL,'I2, 'N2=',N2,'I2, 'IDR=',IDR,'I1,
* 'IP=',IP,'I1, 'IT=',IT,'I1, 'PAR=',PAR,'I1)
WRITE (10,4)
4 FORMAT (1H1)
FOR J=1,M
  .  read (50, fmt="(10F8.4)") (X(J,k) ,k=1,L)
  END FOR
IF (IT .EQ. 1)
  .  FOR J=1,L
  .   DMAX(J) = X(1,J)
  .   DMIN(J) = DMAX(J)
  .   FOR I=2,M
  .     DMAX(J) = AMAX1(DMAX(J),X(I,J))
  .     DMIN(J) = AMIN1(DMIN(J),X(I,J))
  .   END FOR
  .  END FOR
  .  FOR J=1,L
  .   FOR I=1,M
  .     X(I,J) = (X(I,J) - DMIN(J)) / (DMAX(J) - DMIN(J))
  .   END FOR
  .  WRITE (3,) " RANGE="DMAX(J)-DMIN(J)
  . END FOR
ELSE
  .  CALL TRANS(M,L)
END IF
FOR I=1,M
  .  SUM(I)=0.0
END FOR
FOR I=1,M
FOR J=1,L
  SUM(I)=SUM(I)+X(I,J)
END FOR
FOR I=2,M
  SMIN=AMIN1(SMIN, SUM(I))
  SMAX=AMAX1(SMAX, SUM(I))
END FOR
SMIN=SUM(1)
SMAX=SUM(1)
FOR I=2,M
  SMIN=AMIN1(SMIN, SUM(I))
  SMAX=AMAX1(SMAX, SUM(I))
END FOR
WRITE (3, 5) SMIN, SMAX
FOR I=1,M
  P(I)=INT(N*(SUM(I)-SMIN)/(SMAX-SMIN))+1
  IF(P(I) .GT. NL) P(I)=NL
END FOR
WRITE (10, FMT='(10X,60H)' 11 ) ( VALUE OF THE STAT, AT EACH TEXEL,
* FOR EACH STAT )
FOR J=1,L
  WRITE(10, FMT='(1H ,10(F8.4))' 11 ) (X(I,J),I=1,M)
END FOR
FOR N=NL,N2
  IF(IP .NE. 0) GO TO 9
  K=0
  FOR I=1,M
    K=K+1
    IF(K .GT. N) K=K-N
    P(I)=K
  END FOR
  CONTINUE
  IF(IP .NE. 0)
    FOR I=1,M
      P(I)=INT(FLOAT(N)*(SUM(I)-SMIN)/(SMAX-SMIN))+1
      IF(P(I) .GT. N) P(I)=N
      IF(P(I) .LE. 0) P(I)=1
    END FOR
    WRITE (10, FMT="(2014)" 11 ) (P(I) ,I=1,M)
    CALL KMEANS(P,SD,M,L,N,S,E,D,Q,IDR,PAR)
    WRITE (10, FMT="(2I10,2F12.2)" ) (IX(I),P(I),CMU(I),RCT(I),I=1
    WRITE (10, FMT="(1010,2F12.2)" ) (IX(I),P(I),CMU(I),RCT(I),I=1
    FOR K=1,L
      RANGE=MAX(K)-DMIN(K)
      WRITE (10, FMT="(10F8.4)" ) ((DMIN(K)+RANGE*S(J,K)) ,J=1,N)
    END FOR
    WRITE (10, FMT="(10F8.4)" ) ((DMIN(K)+RANGE*S(J,K)) ,J=1,N)
    END FOR
    WRITE (10, FMT="(1H ,10I8)" ) (Q(J),J=1,N)
    WRITE (10, FMT="(1H ,10F8.3)" ) (E(J),J=1,N),D
    DO 99 I=1,N
      IF (Q(I) .NE. 0)
        RAT(I)=E(I)/FLOAT(Q(I))
      END IF

99 CONTINUE
CONTINUE
     WRITE (10, 4)
     WRITE (10, FMT='(1H,10F8.4)') (RAT(I), I=1,N)
     WRITE (10, 11)
     FORMAT(1X,131(1H-))
     END FOR
     STOP
     END

SUBROUTINE KMEANS (P,SD,M,L,N,S,E,D,Q,IDR,PAR)

THE INITIAL ASSIGNMENT OF THE VECTORS X(M,L) TO N CLUSTERS IS
GIVEN BY THE ARRAY P, WHERE P(I) IS THE CLUSTER NUMBER OF THE
I-TH VECTOR; THEREFORE, EACH P(I) MUST BE SUCH THAT 1.LE.P(I)
.LE.N AND FOR EACH J : 1,2,...,N AT LEAST ONE I WITH P(I) = J
MUST EXIST. IN THE EVENT OF A LOSS OF CLUSTER, THE LARGEST
CLUSTER IS SPLIT, DIVIDED UP, AND THE PROCESS CONTINUES.
TRIV 11/1/86

LET D DENOTE THE SUM OF THE E(J), WHERE E(J) IS THE SUM OF THE
SQUARES OF THE DISTANCES BETWEEN THE MEMBERS OF THE J-TH CLUSTER
AND THEIR CENTROIDS.

THE SUBROUTINE MINIMIZES D AS FAR AS IS POSSIBLE BY REPEATED
EXCHANGES IF CLUSTER MEMBERS. THE P(I) ARE CORRESPONDINGLY
MODIFIED WITHOUT, CHANGING THE NUMBER OF CLUSTERS.

THE SUBROUTINE RETURNS THE VALUES AT THE FINAL CONFIGURATION FOR
THE CENTROIDS (S(N,L), THE SUMS E(N), AND D.

IF IDR = 1 THE CURRENT VALUES OF D AND OF THE VECTOR P ARE
PRINTED AT EACH ITERATION.

DIMENSION X(M,L), P(M), S(N,L), E(N), Q(N)
DIMENSION S(25,10),E(25),Q(25),STDV(25,10),SSQR(25,10),VAR(25)
REAL MAX,FR
INTEGER Q,R,U,V,W,IP,IDR,M,L,P(9900),FLAG,PAR,NFLAG,KEEPFLAG
COMMON X(9900,10)
     SPECIAL COMMON
     FLAG=PAR
     WHILE (FLAG .GE. PAR)
     WRITE(3,')" INSIDE WHILE"
     KEEPFLAG=FLAG
     ZERO OUT SUM SQUARE OF SQUARES, QUANTITY, AND ERROR KEEPERS
     FOR J=1,N
     Q(J)=0
     E(J)=0.
     FOR K=1,L
     SSQR(J,K)=0.
     STDV(J,K)=0.
S(J,K)=0.
END FOR
END FOR

GO THROUGH ALL ITEMS, INCMNT'G COUNTERS AND SSQR FOR ALL CLASSES

FOR I=1,M
  R=P(I)
  IF(R .LT. 1 .OR. R .GT. N)
    WRITE(3,) " RETURNING TO MAIN PROGRAM, OR=" , R
    RETURN
  END IF
  Q(R)=Q(R)+1
  FOR K=1,L
    S(R,K)=S(R,K)+X(I,K)
    SSQR(R,K)=SSQR(R,K)+X(I,K)**2.
  END FOR
END FOR

FOR I=1,N
WRITE(3,) " R=" ,Q(I)," I=" ,I
END FOR

TEST TO SEE IF ANY CLASS IS EMPTY: IF SO, SPLIT BUGGEST AND CONT

FOR J=1,N
  R=Q(J)
  IF(R .EQ. 0)
    IMAX=-1000
    FOR I=1,N
      IMAX= MAXO(IMAX,Q(I))
    END FOR
    FOR I=1,N
      IF(IMAX .EQ. Q(I)) IBIG=I
    END FOR
    Q(J)=Q(IBIG)-Q(IBIG)/2
    Q(IBIG)=Q(IBIG)/2
    KT=0
    FOR I=1,M
      IF(P(I) .EQ. IBIG)
        P(I)=J
        FOR A=1,L
          S(J,A)=S(J,A)+X(I,A)
          S(IBIG,A)=S(IBIG,A)-X(I,A)
        END FOR
        KT=KT+1
      END IF
    END FOR
    EXIT FOR IF(KT .EQ. Q(J))
  END IF
END FOR

FOR I=1,N
WRITE(3,) " R=" ,Q(I)," I=" ,I
END FOR
END FOR
CALCULATE NEW CENTROID VALUES S(J,K) FOR EACH CLASS J=1,N

FOR J=1,N
  R=Q(J)
  F=1./FLOAT(R)
  FOR K=1,L
    S(J,K)=S(J,K)*F
  END FOR
END FOR

FOR J=1,N
  IF (MAX VAR(J)) JMAX=J
END FOR

FOR K=1,L
  S(JMAX,K)=S(JMAX,K)-STDV(JMAX,K)*0.5
END FOR
N=N+1
NFLAG=0
S(N,K)=S(JMAX,K)+STDV(JMAX,K)*0.5
END FOR
END IF

CALL MINDIST TO RE-SORT THE VECTORS, EXIT IFF # OF SHIFTS LE FLAG

CALL MINDIST(S,STDV,M,N,L,P,FLAG)
WRITE (3, ) " FLAG=",FLAG
EXIT WHILE IF (FLAG .NE. KEEPFLAG)
END WHILE
FOR I=1,N
  Q(I)=0
END FOR

FOR I=1,M
  R=P(I)
  Q(R)=Q(R)+1
  F=0.
  FOR K=1,L
    T=S(R,K) -X(I,K)
    STDV(R,K)=STDV(R,K)+T*T
    F=F+T*T
  END FOR
  E(R)=E(R)+F
END FOR
D=0.
FOR I=1,N
  IF(Q(I) .EQ. 0)
    WRITE(3, ) " HAH! THE FUCKER'S HERE!, I=",I
  Q(I)=1
END IF
FR=FLOAT(Q(I))
FOR J=1,L
  STDV(I,J)=SQRT( STDV(I,J)/FR )
END FOR
D=D+E(I)
**PROGRAM MINDIST, A MINIMUM DISTANCE CLASSIFIER.**

This program takes as inputs, the centroids of optimally determined clusters, and texel vectors of length ten. The vector is placed that cluster for which the mean squared error is minimum.

TRIV 4/22/84

**SUBROUTINE MINDIST(cent, stdv, nrows, nclus, l, s, flag)**

INTEGER nrows, nclus, p(9900), s(9900), l, flag
REAL cent(25), stdv(25), vecr(10)
REAL *12 min, ss(25)
COMMON x(9900,10)
SPECIAL COMMON
FLAG=0
FOR M=1,nrows
  R=S(M)
  FOR I=1,l
    vecr(I)=x(M,I)
  END FOR
  MIN=10.**6.
  FOR I=1,nclus
    ss(I)=0.
    IF (I .NE. R)
      FOR J=1,L
        ss(I)=ss(I)+(cent(I,J)-vecr(J))**2.
      END FOR ! J
      MIN=AMIN1(MIN,ss(I))
    END IF
  END FOR
  IF(MIN .EQ. SS(I)) P(M)=I
  IF(P(M) .EQ. S(M)) FLAG=FLAG
  ELSE
    S(M)=P(M)
    FLAG=FLAG+1
  END IF
WRITE(30,fmt="(10I7)") (P(I), I=1,10)
END FOR
IF(FLAG .GE. 1) FLAG=1
RETURN
STOP
END

**SUBROUTINE TRANS(M,L)**

COMMON X(9900,10)
SPECIAL COMMON
F=FLOAT(M)
DO 3 K=1,L
  T=0.
  U=0.
  DO 1 I=1,M
    V=X(I,K)
    T=T+V
    U=U+V*V
1  CONTINUE
C     WRITE(3,) " F=",F
     Q=T/F
C     WRITE(3,) " U=",U," T=",T
     S=SQR((F-1.)/(U-T*Q))
C     WRITE(3,) " S=",S
     DO 2 I=1,M
     X(I,K)=S*(X(I,K)-Q)
2  CONTINUE
3  CONTINUE
RETURN
END
APENDIX G3

C****************************************************************************
C PROGRAM TO GENERATE color side scan file from echo map.
C GENERATES A n VALUE LINEAR RAMP
C TEXTFILE, A SHORT ROUTINE TO TRANSFORM CLUSTER CLASSES TO SIDE-
C SCAN COMPATIBLE FORMAT. TRIV 2/26/84
C*****************************************************************************

INTEGER DATA(994),PX(61),SX(61),E,N,NPTS,PAMP(984),SAMP(984)
INTEGER IYR,IMIN,ISEC,NCLASS(9000)
INTEGER SSFNAME(6),TEXNAME(6),NEWSSF(6),NEWFIL,SSFIL,TEXFIL,IERR
REAL SEC,PAR, CT(9000),RMEAN(9000)
COMMON CT,RMEAN,NCLASS
SPECIAL COMMON
DATA NPTS/61/
DATA DATAFIL,SSFIL,TEXFIL,NEWFIL/22,24,25,26/
read(20, fmt=" ( IS ) 11 ) nc
C READ(21, FMT=" (IS) ") IYR
LOOP
  READ(20,FMT="(2I10,2F12.2) 11 , END=2) I, NCLASS(I),RMEAN(I),CT(I
C if (ct(i) .ge. it) ntc=ntc+1
END LOOP
2 CONTINUE
10 READ (DATAFIL,21,END=99) SSFNAME,TEXNAME,NEWSSF
21 FORMAT(18(A3))
CALL ASSIGN(SSFIL,SSFNAME,IERR)
IF (IERR .NE. 0) STOPAS24
BUFFER IN(24,DATA,B,24,E,N)
CALL STATUS (24)
IF (IGENER(NEWSSF ,5,50,2,3000) .GT. 1) STOP GENSSF
CALL ASSIGN(NEWFIL,NEWSSF,IERR)
IF (IERR .NE. 0) STOPAS26
BUFFER OUT(26,DATA,B,24,E,N)
CALL STATUS (26)
CALL ASSIGN(TEXFIL,TEXNAME,IERR)
IF (IERR .NE. 0) STOPAS24
K=1
IF (FLAG .NE. 0) GO TO 15
C loop
  FOR K=1,2
  15 . BUFFER IN(24,DATA,B,994,E,N)
    CALL STATUS (24)
    IF (E .EQ. 3 .OR. E .EQ. 4) .CLOSE NEWFIL
    FLAG=1
    GO TO 10
  END IF
  END FOR
  read(25, fmt="(31I4)", end=10) (PX(i),i=1,61)
  read(25, fmt="(31I4)", end=10) (SX(i),i=1,61)
  for i=1,61
    p= int(255*NCLASS(PX(i))/float(nc))
    S= int(255*NCLASS(SX(i))/float(nc))
for j=1,16  
pamp(j+16*(i-1))=p  
Samp(j+16*(i-1))=S  
end for  
end for  
for k=1,2  
  FOR L=11,994  
    DATA(L)=(PAMP(L-10) .AND. '377)  
    DATA(L)=DATA(L) .OR. (SAMP(L-10) .AND. '377)*256  
  END FOR  
  BUFFER OUT( 26,DATA,B,994,E,N)  
  CALL STATUS( 26)  
  END FOR  
end loop  

99 ENDFILE 26
STOP
END

SUBROUTINE GETIME(INBUF,IDAY,IHR,IMIN,SEC)

C SUBROUTINE TO RETURN THE TIME AS DAY (FROM JAN0)
  HOUR
  MINUTES
  SECONDS
AS GOTTEN FROM HEADER RECORDS

C INPUT
C INBUF(1) = MILLISECS
  4 BITS/DIGIT, GIVEN TO .XXXX ACCURACY
C INBUF(2) = MINUTES AND SECONDS
  4 BITS/DIGIT, GIVEN AS MMSS
C INBUF(3) = DAY AND HOUR
  BITS 1-4 = UNITS OF HOUR
  5-6 = TENS OF HOUR
  7-10 = UNITS OF DAY
  11-14 = TENS OF DAY
  15-16 = HUNDREDS OF DAY

C OUTPUT
C IDAY = DAY (FROM JAN 0)
C IMIN = MINUTES
C IHR = HOUR
C IMIN = MINUTES
C SEC = SECONDS TO .XXXX

C DIMENSION INBUF(1)
C REAL*12 SEC
DATA MASK2,MASK4/'3,'17/

C C GET MILLISECS
MSEC=INBUF(1)
MS=(((MSEC/4096).AND.MASK4)*1000 +
X((MSEC/256).AND.MASK4)*100 +  
Y((MSEC/16).AND.MASK4)*10 +  
Z((MSEC).AND.MASK4)

C GET MINUTES AND SECONDS

MSEC=INBUF(2)

IMIN=((MSEC/4096).AND.MASK4)*10 +  
X((MSEC/256).AND.MASK4)

ISEC=((MSEC/16).AND.MASK4)*10 +  
X((MSEC).AND.MASK4)

SEC=ISEC+MS/1.1E4

C GET DAY AND HOUR

MSEC=INBUF(3)

IDAY=((MSEC/16384).AND.MASK2)*100 +  
X((MSEC/1024).AND.MASK4)*10 +  
Y((MSEC/64).AND.MASK4)

IHR=((MSEC/16).AND.MASK2)*10 +  
X((MSEC).AND.MASK4)

C RETURN
END
BIBLIOGRAPHY


