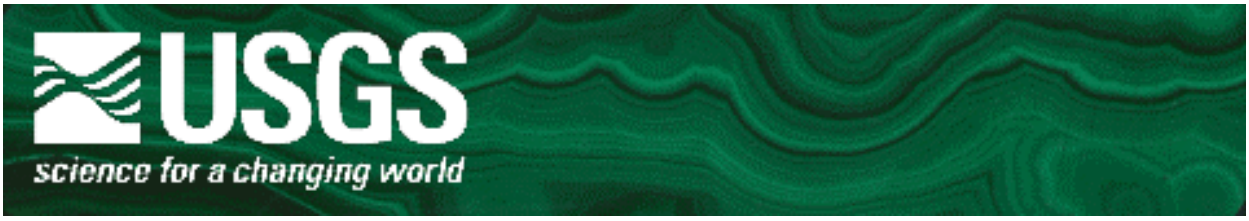


MORPH-I (Ver 1.0), A Software Package for the Analysis of Scanning Electron Micrograph (Binary Formatted) Images for the Assessment of the Fractal Dimension of Enclosed Pore Surfaces

by

Victor G. Mossotti[1], A. Raouf Eldeeb[1] and Robert Oscarson[1]

U.S. Geological Survey Open-File Report OF 98-248

1998

U.S. DEPARTMENT OF THE INTERIOR
BRUCE BABBITT, Secretary

U.S. GEOLOGICAL SURVEY
Charles G. Groat, Director

This report is preliminary and has not been reviewed for conformity with U.S. Geological Survey editorial standards or with the North American Stratigraphic Code. Any use of trade, product or firm names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

[1] U.S. Geological Survey, Menlo Park, CA 94025

NOTE

MORPH-I is a set of C-language computer programs for the IBM PC and compatible minicomputers. The programs in MORPH-I are used for the fractal analysis of scanning electron microscope and electron microprobe images of pore profiles exposed in cross-section. The program isolates and traces the cross-sectional profiles of exposed pores and computes the Richardson fractal dimension for each pore. Other programs in the set provide for image calibration, display, and statistical analysis of the computed dimensions for highly complex porous materials. Requirements: IBM PC or compatible; minimum 640 K RAM; mathcoprocessor; SVGA graphics board providing mode 103 display.

OF98-248.PDF : Acrobat PDF version of documentation with source code.

OF98-248.EXE: zipped executable files available via FTP

DISCLAIMERS

Although the programs in this set have been used by the U.S. Geological Survey, no warranty, expressed or implied, is made by the USGS as to the accuracy and functioning of the programs and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

INTRODUCTION

In recent years, fractal mathematics has provided an alternative to Euclidian mathematics as an approach to the problem of modeling the complex microstructure of porous geological materials (Turcotte, 1992). Our interest in this report is in the fractal analysis of electron micrograph images of porous specimens with highly variegated surfaces.

For the purposes of this discussion, the total volume of dry rock specimens can be regarded as consisting of three phases. Two of these phases, the solid state rock matrix and the gas phase pore-space, exhibit Euclidian dimension equal to 3 (D_E=3). The third phase, a thermodynamically separate entity with Euclidian dimension equal to 2 (D_E=2), is the excessively folded surface at the interface between the intermingled mass and pore-space volumes; we refer to this area as the pore-space surface. If the pore-surface under examination is statistically self-similar over a bounded range-of-scale, then the fractal dimension, D, is the most fundamental scale-independent parameter associated with the shape of the surface (Barton and La Pointe, 1995).

IMAGE PROCESSING

Overview of MORPH-I v. 1.0 programs

Figure 1 schematically shows the relationships between the various program modules in MORPH-I v. 1.0 and their respective applications. The programs can be considered in three sections: (1) file formatting, (2) image display, and (3) image calibration and analysis.

Hardware requirements

MORPH-I is designed to run on an IBM or compatible PC with a mathcoprocessor and a minimum of 640 K RAM. The main analysis and calibration programs require SVGA graphics capability providing mode 103 display.

Format requirements

The image analysis programs provided in MORPH-I v. 1.0 operate on 512 x 512 x 8 binary files (images with 512 rows, 512 pixels/row, and 8 bits/pixel on a 256 shade gray scale palette); the GIF format with an image specification of 512 x 512 x 8 is compatible with MORPH-II v. 1.0[1/]. If the requisite file format is provided by the image source, the images can be displayed, calibrated, and analyzed without any further concern for format.

[1]MORPH-II, to be released in 1999, computes the fractal dimension of GIF-formatted images of exterior surfaces viewed in cross-section.

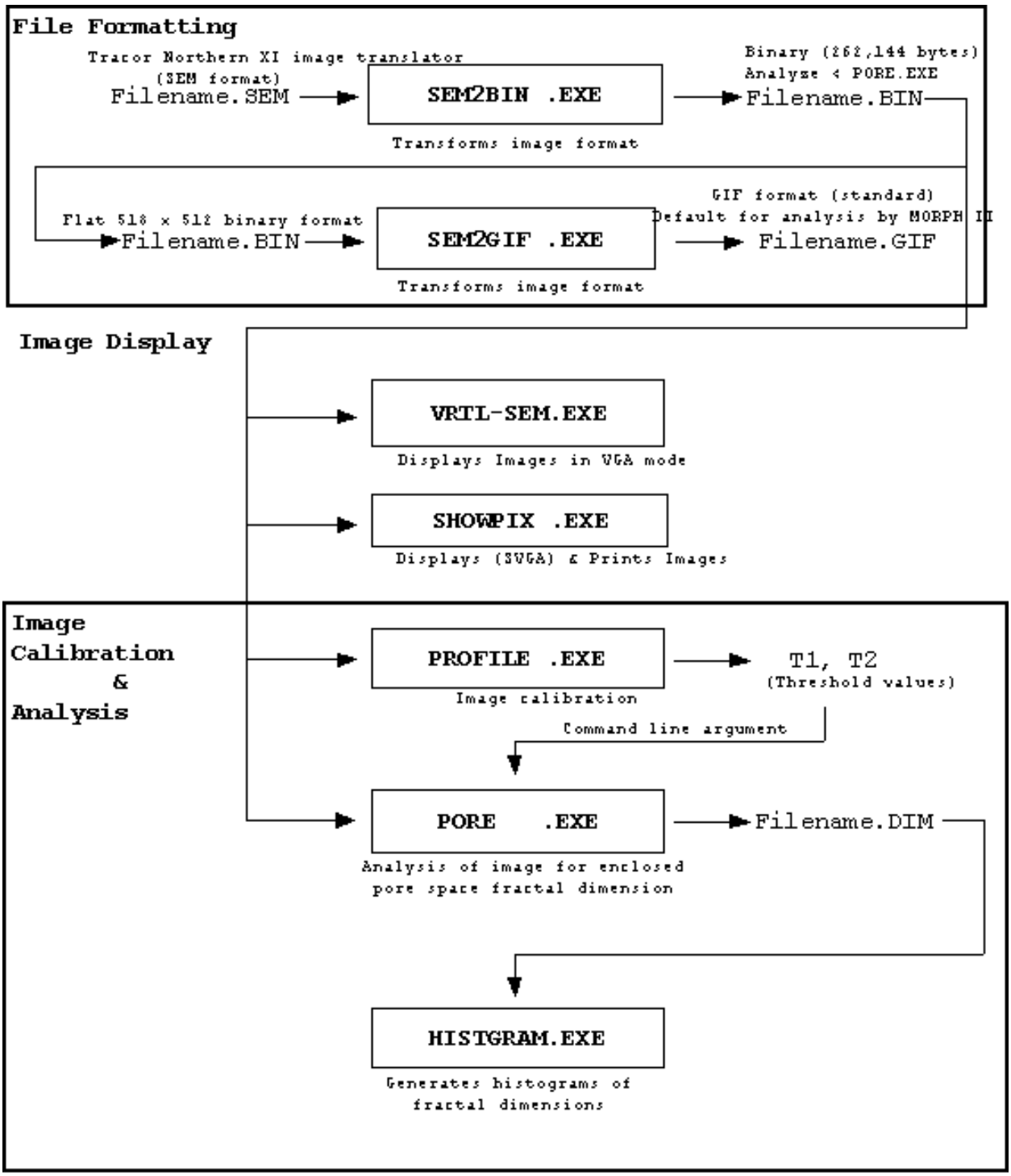


Figure 1, Overview of modules in MORPH v. 1.0

PROGRAM OPERATION

Format conversion utilities

Program name SEM2BIN

Program application

SEM2BIN converts the special Tracor Northern (TN) IMG format to a binary format suitable for image processing. The original analog SEM image is transferred in a proprietary format with the TN VISTA program via the external scan interface to the TN 5500 mini computer attached to the SEM. The TN image file, consisting of 512 rows each containing 512 pixels, is encoded with 8 bits of information/pixel representing 256 shades of grey. The image file is down-loaded from the TN 5500 serial port to a local block-device with the XI (TN) terminal emulation program; a Macintosh-II (40 MB hard disk) was used at the USGS (Menlo Park) SEM laboratory to log the image files.

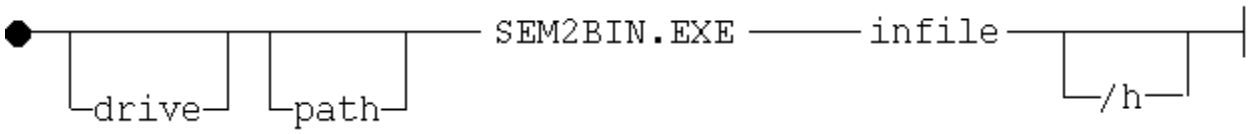
The specially formatted image files are given an IMG extension by the VISTA (TN) program, which coincidentally, has nothing to do with the standard Ventura Publishing IMG-format. These files can then be further down-loaded to a work station either over the telephone line, over a LAN, or with a physical diskette. Programs in the public domain such as MAC&DOS are available for translation of the Macintosh-II files to a PC-compatible format. For transmission over the telephone lines, any set of standard communication packages may be been used. The KERMIT protocol is especially useful because of its ability to support batch processing.

Program environment

SEM2BIN.EXE is invoked from an MSDOS-compatible command line.

Synopsis of the command line

As a default, SEM2BIN converts image files in the TN IMG format to a binary format, subject to the following syntax diagram[2]:



[2]When viewing a syntax diagram, read from left to right and from top to bottom. The command name and items required to make the command work appear on the baseline; the items below the baseline are optional. A line ending with an arrowhead means that the diagram is continued. A line starting with an arrowhead means that the syntax is continued from the previous line. A vertical bar marks the end of the command syntax.

where the switch /h cues the program to write the header information to the monitor (stdout) and to the output file.

Program I/O

The input file is an image file in the IMG (TN) format. The output file is given the input file stem-name with an SEM extension. With the header information discarded SEM2BIN.EXE generates an output file consisting of an array of 262,144 bytes.

Image file header

Line-1 in the input IMG file contains the ASCII string "BINARY BLOCKS = nnnnn" (21 bytes not counting LF/CR). Here, nnnnn is a decimal number representing the number of pixel-blocks in the file (512 pixels/pixel-block). This information, along with the 2048 bytes of header space, is written to the original data file by the TN file management facility before the file is passed to down-loaded to the TN 5500 mini computer. Typically, header information might include the user's name, the date, and the conditions under which the image was collected. If the header information provided by the user is insufficient to fill the header space, the unused space is padded with '*' characters by the TN program. If the command line switch is set to include the header information, the first line of the IMG file is discarded and the next 64 lines of header information are converted to their original ASCII characters and written to the monitor. The actual image data, which starts at the 2049th byte, is indexed 2048_D (800_H).

Image data

Each byte of image information represents the intensity of one pixel on a scale of 256 shades of gray. Each row in the 512 x 512 byte image is represented by serially ordered subsets of 512 bytes in the array and are associated with the original image as follows:

```
{   pixel at row0, column0  ;  pixel at row0, column1  ;   ...
    pixel at row1, column0  ;  pixel at row1, column1  ;   ...
    etc ...   ( no line feeds between rows )               ...
}
```

The XI (TN) terminal emulator carries out its function without error checking. From time-to-time, XI fails to accurately transmit part of a line or a number of a complete lines. If the image file is corrupted by an odd number of bytes, the resulting image may appear as a blur of random pixels after a certain point. If an even number of bytes are lost, a shift may appear at some point in the image. When such problems occur, it is possible to repair the image file with a text editor by padding the corrupted line or by grafting additional lines into the file. As a

diagnostic for such problems, SEM2BIN reports on all lines that do not contain exactly 64 characters and on the total number of lines in the file. This information is useful for identifying the corrupted location in the image file. The reader who is interested in converting the image files to a standard format (TIFF, PCX, GIF, Postscript, etc.) can refer to the book: Graphics File Formats by O'Reilly & Associates.

Instructions for use of program

SEM2BIN provides no graphics display and accepts no interaction from the user.

Detailed functioning of the program

The SEM image information is organized as 512 rows and 512 columns. The XI translator scanning proceeds from left to right starting at the upper left corner of the image. The exact location of any given pixel in the image is virtually encoded by the relative position of the pixel in the stream. Accordingly, a pointer to a pixel is sufficient for the recovery of all the information related to the pixel.

Instead of providing data compression, the translator dilutes the data-stream by a factor of two by creating two data bytes for transmission from each pixel-byte. The translator parses each byte of intensity information into the most significant and the least significant four-bit nibbles. Reminiscent of mainframe tactics, the program then creates a full byte from each nibble by adding the ASCII character 'A' (hex 41) to each part. Thus, each pixel is represented by two low bit ASCII characters, all falling in the range 'A' to 'P'. This technique creates characters that are printable on even the most primitive devices. The diluted information is then reverse-scanned and the most significant byte (MSB) is transmitted before the least significant byte (LSB).

The image files sent to the block-device have data encoded by the translator in the following format. The translator organizes the serial data-stream into lines with 64 characters delimited by a carriage return (CR). If the communication package on the host computer is receiving the data with a text protocol in a DOS environment, it will add a linefeed (LF) character before all CR characters as the CRs are detected. If the host is receiving the data with binary protocol, the LF will not be inserted before the CR. The 64 bytes on a line, not counting LF/CR, represent the intensity of 32 pixels in the image. Thus, it takes 16 lines of encoded data to represent one row of 512 pixels in the image. Hence, the 512 rows in the original image require 8192 lines.

Line-1, which is not encoded, is followed by 64 filled lines of encoded header information. The encoded image data starts at line 66 in the IMG file. Because each row of 512 pixels in the image

is diluted to 1024 bytes in the IMG format, each row requires 16 lines in the down-loaded file. Hence, the 512 rows in the original image require 8192 lines. The end-of-encoded data is indicated by the character '!' on a separate line followed by the LF/CR combination on a PC. In summary, the structure of the image file, diluted to the TN IMG format, consists of the following:

Unit	**Bytes**	**Lines**
First line (w/ terminators)	21	1
Header (w/ terminators):	2 x 2,048 = 4,096	64
Image	2 x 512^2 = 524,288	8,192
Line terminators: CR	$512^2/64$ = 4,096	-
Line terminators: LF/CR	2 x $512^2/64$ = 8,192	-
EOF sentinel	3	1

It follows that the IMG formatted file, including the first line (21 bytes), the encoded header (4096 bytes), the expanded image (2 x 512^2), all attached LF/CRs (2 x $512^2/64$), and the EOF sentinel, contains 8258 lines requiring 536,600 bytes of memory on a PC. If lines in the IMG format are terminated by only a CR, the IMG file will require 532,502 bytes. The difference in file size is significant when the file size is used to check the fidelity of the data transmission.

Special libraries
Standard ANSI C.

Source code
The kernel source code for SEM2BIN.EXE is provided in Appendix-A, Exhibit A-1.

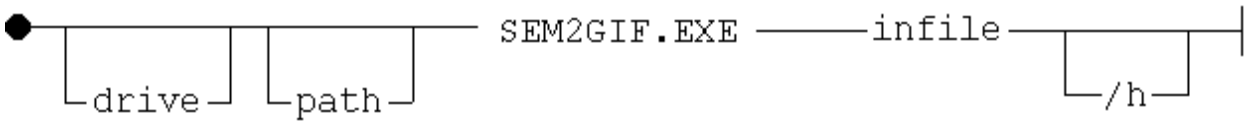
Program name SEM2GIF

Program application
SEM2GIF converts the special TN IMG format described in detail above to a format suitable for GIF image processing. The original SEM image, consisting of 512 rows each containing 512 pixels, is ideally suitable for representation in the GIF format. The information content in the original image, encodes with 8 bits of information/pixel representing 256 shades of grey, is retained the conversion of the TN IMG format to the GIF format.

Program environment
SEM2GIF.EXE is invoked from an MSDOS-compatible command line.

Synopsis of the command line

As a default, SEM2GIF converts image files in the TN IMG format to a GIF format, subject to the following syntax diagram:



where the switch /h cues the program to write the header information to the monitor (stdout).

Program I/O

The input file is a 512 x 512 x 8 binary image file; the output file is given the input file stem-name with a GIF extension. Depending on the amount of redundancy in the image file, the GIF formatted file will be about 40% smaller than the original data file as a result of data compression.

Instructions for use of program

SEM2GIF provides no graphics display and accepts no interaction from the user.

Detailed functioning of the program

The program SEM2GIF.EXE operates by converting electron micrograph image files in the TN IMG format into the standard GIF format. Details of the TN IMG file structure are given in the preceding section on the operation of SEM2BIN.EXE.

Special libraries

Standard ANSI C; GIF_LIB.C; and GIF_LIB.H. A set of routines which perform encoding and decoding of GIF formats were written in-house, including compression and decompression of the binary image data. The calls to this library are limited to two types. The first type is a set of routines to open, close, read and write screen descriptors as well as image descriptors to or from the GIF file. The second type is a set of line-oriented routines to encode or decode a row of image data. The functions in GIF_LIB.C and GIF_LIB.H. include:

Type-1

- CloseGifFile ()
- OpenGifFile ()
- GifPutScreenDesc ()
- GifPutImageDesc ()
- GifGetImageDesc ()
- GifGetScreenDesc ()

Type-2

- GifPtuLine ()
- GifGetLine ()

Source

The full source code for SEM2GIF is provided in Appendix-A, Exhibit A-2. The GIF_LIB.C and GIF_LIB.H. library provides the

application program interface (API) needed for the development and maintenance of MORPH-I v. 1.0; the source code for the GIF_LIB.C and GIF_LIB.H library is provided in Appendix-A, Exhibit A-3.

Image display utilities

Apart from the display functions provided in the analysis programs, image display for binary image files is provided by two stand-alone programs, VRTL-SEM.EXE for display in the VGA mode, and SVGA-SEM.EXE for display in the SVGA mode. Since the display function is provided in the analysis programs, the use of these programs is mainly for trouble shooting transmission errors in binary files.

Program names VRTL-SEM.EXE, SHOWPIX.EXE

Program

Image display. Two DOS applications, VRTL-SEM.EXE and SHOWPIX.EXE were written using different image manipulations and graphics modes to display the binary image on a PC monitor.

Program environment

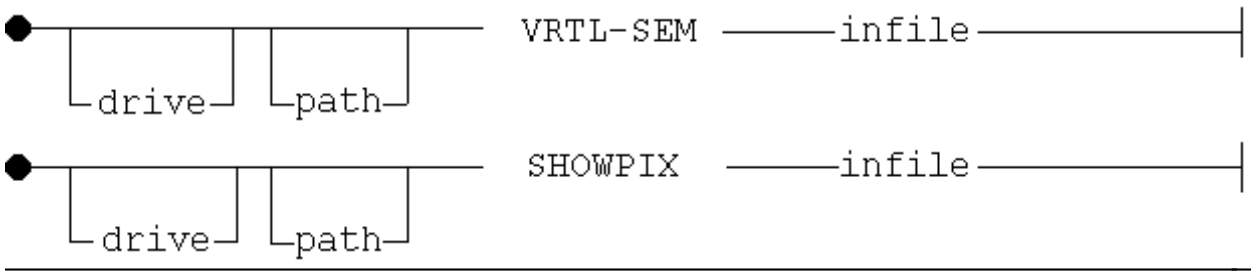
VRTL-SEM This program uses VGA mode 320 x 200 x 256 colors to display a section of 320 x 200 pixels of the image in 64 shades of gray. The arrow keys can be used to display contiguous regions of the image (the arrow keys manipulate near pointers in video memory). This creates the illusion of SEM operation on the PC monitor.

SHOWPIX This application uses SVGA 800 x 600 (mode 103) to display the whole image in 128 shades of gray; the name of the file is displayed in color. The program requires the proprietary Flash Graphics library[3] which produces faster and more efficient code than the Borland graphics library.

The display programs can be invoked with a DOS mask to display the first 50 files that match the mask. Because DOS cannot recognize its own mask symbols, the masking function is included in the program code. All versions of SHOWPIX have the same basic structure and share much of the same code.

[3] Copyright© FlashTek, Inc. 1993, 121 Sweet Ave., Moscow, ID 83843; Joe Huffman, February 2, 1993, FAX: (208) 882-7275, email: joe@proto.com.

Synopsis of the command line



Program I/O Both image display programs accept a 512 x 512 x 8 binary image input file.

Instructions for use of programs Both programs display the electron micrograph on the PC monitor. VRTL-SEM accepts input from the arrow keys to translate the displayed part of the image on the monitor. This creates the illusion that the user is scanning the specimen surface with the SEM. SHOWPIX displays static images on the monitor.

Detailed operation of display programs The display programs show the images that match the command line mask sequentially. Available Commands in all versions include:

- Enter or PageDown displays the next image
- PageUp displays the previous image
- Escape, 'Q', 'X', or F10 will exit the program to the command line after restoring text mode.

VRTL-SEM provides additional control with the arrow keystrokes:

- Up, Down, Left and Right will display the corresponding area of the image (if any)

Special libraries The display programs use the standard ANSI C library. VRTL-SEM also uses the Borland Graphics library (BGI); and SHOWPIX uses the Flash Graphics library.

Description of C-functions used in VIEW-VGA.EXE, SHOWPIX.EXE, and VRTL-SEM.EXE

setup	Parses the command line
get_filelist	Retrieves up to 50 files with names that match the command line mask
initiate_graphmode	Sets the display in the proper graphics

	mode
get_input	Gets user input
set_grayscale	Loads color register with gray scale combinations
read_image	Reads the file containing the next image to be displayed
show_image	Displays the current image in memory
terminate_graph	Retuns the display to text mode
SetColorReg	Sets individual color registers
Warn	Emits an audible warning in case of error

Source code The full source code for the diaplay programs is provided in Appendix-A, Exhibits A-4 and A-5.

Image calibration

The image analysis program, pore.c, uses a finite state approach for the isolation of the pore boundaries. This approach allows the user to calibrate each separate image by establishing a threshold brightness-level for the definition of MASS and PORE states.

Program name PROFILE

Program application

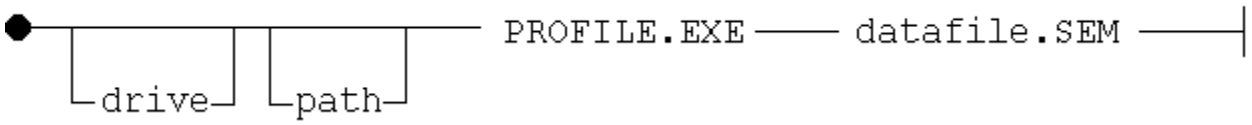
Image calibration. In the MORPH-I v. 1.0 package, the image calibration function is provided in a stand-alone program called PROFILE.EXE, along with a continuously updated display of the pixel distribution across the 256 gray scale palette. Additional analytical functions provided in MORPH-I v. 1.0 include the isolation and analysis of each pore-trace in the image with fractal statistical information reported on each trace, and the ability to reject on-the-fly the inclusion of particular pore-traces from the summary statistics on the image as a whole

Program environment

PROFILE is scheduled from an MSDOS-compatible command line.

Synopsis of the command line

PROFILE requires an input image file in binary format as discussed in the foregoing section on format conversion. On termination, the program displays the upper and lower values of the threshold pseudocolor. PROFILE.EXE is run from the DOS command line in accordance with the following syntax diagram:



Program I/O

PROFILE displays the input binary image on the monitor with a histogram of the gray-values in the image on the same screen.

Instructions for use of program

Because the selection of the calibration parameters T_L & T_U is somewhat subjective, PROFILE.EXE is an interactive program designed to assist in the calibration. Using PROFILE.EXE, the user can partition the full set of image pixels into three sets. The user graphically selects upper and lower threshold values from the histogram of pixel pseudocolors while viewing a dynamically updated tri-color three-state image. Figure 2 shows the graphical image displayed by PROFILE.EXE. The bimodal histogram in figure 2 shows a high occurrence-frequency of both dark and light pixels. The location of the bimodal peaks along the gray-scale differs from image to image because images are not produced with the same average brightness. Threshold values can be interactively selected on the histogram such that all pixels below the lower threshold are displayed as black (logical value: PORE), and all pixels above the upper threshold are displayed as white (logical value: MASS). The pixels falling between the threshold levels are displayed as a particular gray-value pseudocolor (logical value: EDGE, shown in red on the computer screen and as black in figure 2). By moving the lower threshold to the left, the user effectively accepts for consideration contour levels deeper into the pore-cavity. This action reduces the measured pore-volume by increasing the number of MASS pixels at the expense of PORE pixels. Likewise, a shift of the upper threshold level to the right may increase the apparent pore-volume. In practice, threshold levels can be independently evaluated by comparing the relative pore-volume in a transformed image to experimentally determined porosity values.

Detailed functioning of the program

The C-code provided in Appendix-A, Exhibit A-6 for PROFILE.C is self documenting.

Special libraries

PROFILE.C, designed to run in regular VGA mode, uses the Borland BGI library and the Quinn Curtis (QC) library. With the exception of the functions F1andF3() and F2andF4(), functions that start with an upper case letter are QC functions.

Source code

The complete source code for PROFILE.C is given in Appendix-A, Exhibit A-6. The C-functions used in PROFILE.C include:

get_file Parses the command line, reads data

setup	Opens the image file, error message file, and report file; initiates graphics mode; allocates memory for data; sets up palette for 15 levels of gray; exits if anything goes wrong
initiate_graphmode	Sets the display in the proper graphics mode
SetColorReg	Sets individual color registers
read_image	Retrieves image data
SetGrayScale	Sets 15 gray scale and one highlight color
show_image	Displays the current image in memory
screen_message	Outputs screen message in graphics mode
Warn	Emits an audible warning in case of error
initial_display	Displays the original image and usage messages (menu of instructions)
make_hist	makes histogram of image pixel intensities
display_hist	Refreshes the histogram and thresholds displays (This function uses the Quinn Curtis library)
get_input	Gets user input
F1andF3	Removes MASS pixels and adds PORE pixels in tandem to the left side of the histogram
F2andF4	Removes PORE pixels and adds MASS pixels in tandem to the right side of the histogram
PJetBWPrintRow	Prints one row to the paintjet printer in B&W
process_image	Displays image, histogram, processes user input
curtain	Closes file and ends graphics mode and displays the final choices of the user (upper and lower thresh)

add_mass	Changes BOUNDARY pixels to MASS pixels
removed_mass	Changes MASS to BOUNDARY pixels
add_pore	Changes BOUNDARY pixels to PORE pixels
removed_pore	Changes PORE to BOUNDARY pixels
max_upperbound	Maxes out upper bound (turns all red on the upper side)
min_lowerbound	Minimizes lower bound (turns all red on the lower side)
print_screen	prints the histogram as well as the image on an HP printer
terminate_graph	Returns the display to text mode

Image analysis

Program name PORE.EXE

Program application

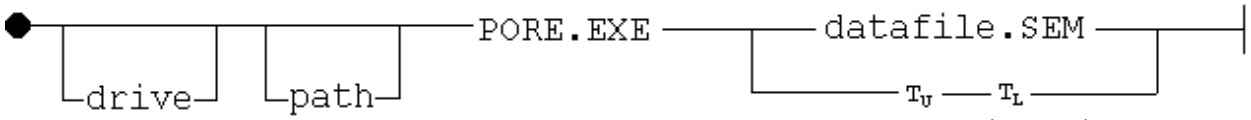
The program PORE.EXE, which requires the calibration parameters to be input on the command line, computes the fractal dimension of the enclosed pore-space surface for every pore revealed by the cross-sectional cut through the rock. It is not uncommon for rocks of complex microstructure to exhibit more than a single fractal dimension. In such cases, it is useful for the analysis data to be written to a report file. In MORPH-I, the report file is given a DIM extension. With a 33 MHz 486 DX computer, PORE.EXE requires approximately 1 second to compute the fractal dimension of several hundred pores. As indicated in figure 1, the report file can be statistically analyzed and displayed in the form of a histogram of fractal dimensions by the program HISTGRAM.EXE

Program environment

PORE.EXE is scheduled from an MSDOS-compatible command line.

Synopsis of the command line

PORE.EXE requires an input image file in binary format as discussed in the foregoing section on format conversion. PORE.EXE is run from the DOS command line in accordance with the following syntax diagram:



where T_U and T_L are the upper and lower threshold calibration parameters for the image. *If T_U and T_L are not provided on the*

command line, the program assumes default values for T_U and T_L of xxx and xxx, respectively.

Program I/O

On termination, PORE.EXE writes an ASCII file containing statistical information (fractal dimension, pixel-length of pore perimeter) on every pore scanned across the cross-section of the sample.

Instructions for use of PORE.EXE

Figure 3 shows the image displayed by PORE.EXE during the analysis of a test specimen. The fractal dimension and pore perimeter length for a particular pore is shown in the left margin of the display while the particular pore perimeter is highlighted in yellow in the image. With the exception of the use of the ESC key to trigger the display of the analytical findings on each individual pore found in the image, the program PORE.EXE requires no additional input from the user.

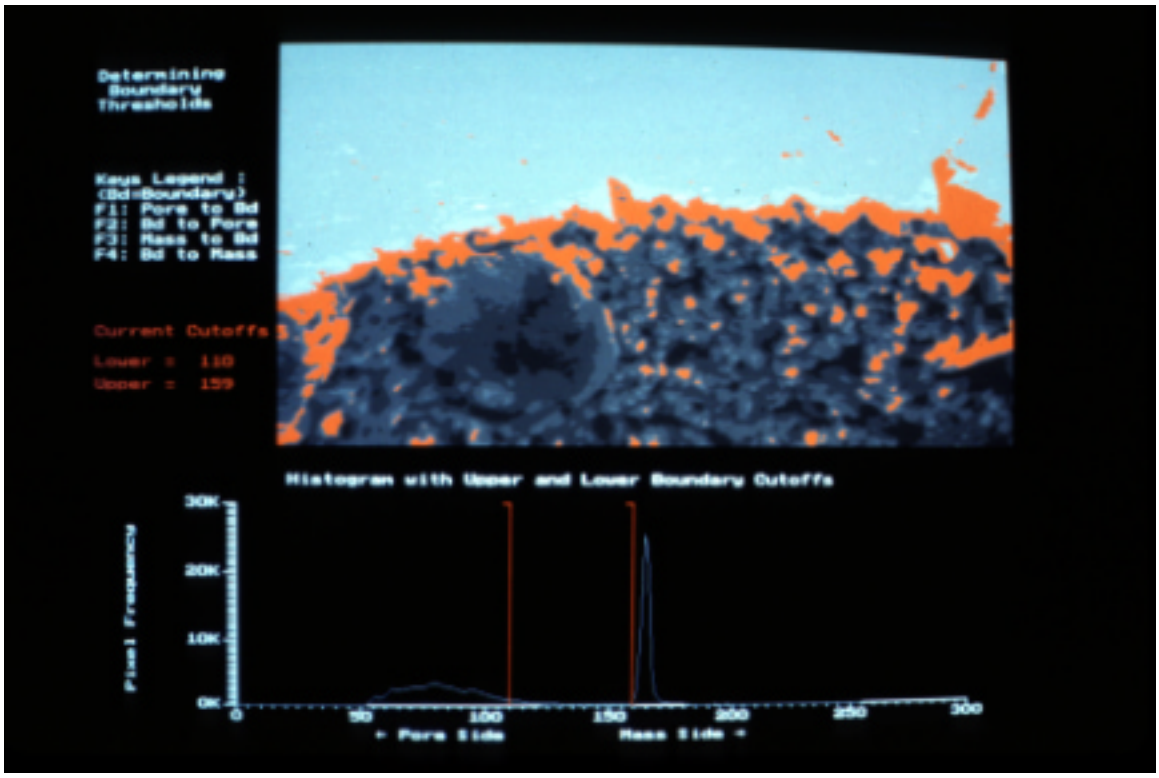


Figure 2, Graphical image displayed by PROFILE.EXE

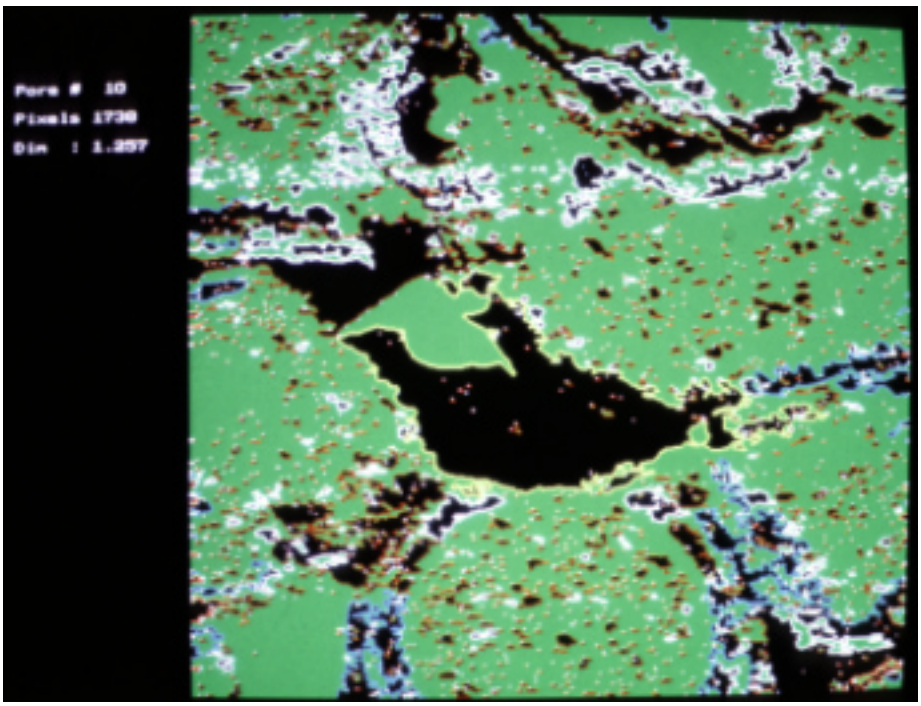


Figure 3, Image displayed by PORE.EXE during the analysis of a test specimen

Detailed functioning of PORE.EXE

The image analysis kernel operates by first evolving the image through three generations of cellular automata where the state of each pixel in each generation is defined by the state of its 1st and 2nd nearest neighbors. As the image evolves, contention issues related to the state of each pixel are continuously resolved. Such issues, which arise along the edges (EDGE pixels) delimiting pore space and the solid phase, are resolved on the basis of the spatial environment of each pixel in the image and by reference to unambiguous definition of pixels representing the solid phase (MASS state) and pixels representing pore space (PORE state). Third generation pixels with a brightness signal greater than the PORE/MASS threshold are designated MASS; those below the threshold are designated PORE. If we assume that the PORE space pixels are generally darker than the gray-level of the lower image calibration threshold, T_L, then the pore boundaries will be strongly influenced by the T_L parameter.

Special libraries

PORE.C, designed to run in SVGA mode 103, uses the Borland BGI library and the Quinn Curtis (QC) library C-programming language library.

Source code

The complete source code for PORE.C is given in Appendix-A, Exhibit A-8.

Program name HISTGRAM.EXE

Program application

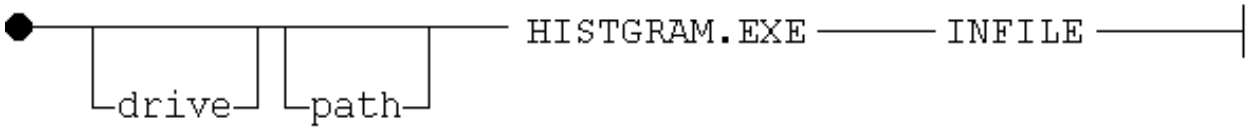
HISTGRAM.EXE generates a histogram of the fractal dimensions computed by the program PORE.EXE as collected in a report file with a DIM extension.

Program environment

HISTGRAM.EXE is invoked from an MSDOS-compatible command line.

Synopsis of the command line

HISTGRAM.EXE reads an ASCII file with a data format produced by PORE.EXE. HISTGRAM.EXE is subject to the following syntax diagram:



Program I/O

The input file is an ASCII file produced by PORE.EXE containing conventional and fractal statistics on an analyzed image.

Instructions for use of program

HISTGRAM.EXE shows a graphics display of the histogram of the set of fractal dimensions computed for all pores detected in the image analysis.

Detailed functioning of the program

The C-code provided in Appendix-A, Exhibit A-9 for HISTGRAM.C is self documenting.

Special libraries

Standard ANSI C and the Quinn Curtis (QC) C-programming language library are called in HISTGRAM.C.

Source code

The full source code for SEM2BIN.EXE is provided in Appendix-A, Exhibit A-9.

REFERENCES

Barton, C. C. and La Pointe, P. R., 1995. Fractals in petroleum geology and earth processes. Plenum Press, New York.

Turcotte, D. L., 1992. Fractals and chaos in geology and geophysics. Cambridge University Press, Cambridge.

ACKNOWLEDGMENT

The authors wish to express their appreciation to the National Park Service National Center for Preservation Technology and Training, Natchitoches, Louisiana, for their support of this project.