Image enhancement and analysis of leukocyte adhesion

Yee-Ruh Lin
New Jersey Institute of Technology

Follow this and additional works at: https://digitalcommons.njit.edu/theses

Part of the Biomedical Engineering and Bioengineering Commons

Recommended Citation
https://digitalcommons.njit.edu/theses/1301

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Theses by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.
Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement.

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation.

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen.
The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.
Abstract

Title of Thesis: Image Enhancement and Analysis of Leukocyte Adhesion

Yee-Ruh Lin, Master of Science in Biomedical Engineering, 1991.

Thesis directed by: Dr. P. Armenante

Department of Chemical Engineering, NJIT, Newark, NJ

Dr. Walter Durán

Division of Microcirculation Research, Department of Physiology

UMDNJ-New Jersey Medical School, Newark, NJ.

Dr. David Kristol

Department of Biomedical Engineering, NJIT, Newark, NJ

The image of leukocyte adhesion to microvascular walls in the isolated rat perfused heart model has been observed and processed. Some of the fluorescent labeled leukocytes are barely visible due to microscope focal plane and the motion of objects. The purpose of this thesis is to extract leukocyte from the image of the experimental data using image enhancement and analysis (segmentation) techniques and develop a program to handle automatically counting of leukocytes in a predefined scene.
Image Enhancement and Analysis of Leukocyte Adhesion

by
Yee-Ruh Lin

Thesis submitted to the Faculty of the Graduate School of the New Jersey Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Biomedical Engineering 1991
Approval Sheet

Title of Thesis: Image Enhancement and Analysis of Leukocyte Adhesion

Name of Candidate: Yee-Ruh Lin

Master of Science in Biomedical Engineering

Thesis and Abstract Approved: ___________________________ ______________

Dr. P. Armenante Date
Department of Chemical Engineering, NJIT, Newark, NJ

_________________________ __________________
Dr. Walter Durán Date
Division of Microcirculatory Research, Department of Physiology, UMDNJ-New Jersey Medical School, Newark, NJ

_________________________ __________________
Dr. David Kristol Date
Department of Biomedical Engineering, NJIT, Newark, NJ
Vita

Name: Yee-Ruh Lin

Degree and date to be conferred: MSBME, 1991

Secondary education: The First Tai-chung senior high school

<table>
<thead>
<tr>
<th>Collegiate institutions attended</th>
<th>Dates</th>
<th>Degree</th>
<th>Date of Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chung-Yuan Christian University</td>
<td>1980-84</td>
<td>BSBME</td>
<td>1984</td>
</tr>
<tr>
<td>New Jersey Institute of Technology</td>
<td>1989-91</td>
<td>MSBME</td>
<td>1991</td>
</tr>
</tbody>
</table>

Major: Biomedical Engineering
Acknowledgement

The thesis would not have been completed without helps from a lot of kind people. Special thanks goes to my advisor Dr. Walter N Durán. He inspired my interest in this research, and carefully revised draft manuscript. Further, I would like to thank some members of computer vision lab in NJIT, Mr. Yui-Laing Chen and Mr. Chris Pu, who helped me to develop algorithms and softwares; Mr. Kung-Wei Chirn and Mr. J. Moh, both of them are the best tutors in computer program languages. I also like to thank Dr. Leslie Michaelson for his help in data processing.
## Contents

1 Introduction 1

1.1 Biological background ................................................. 1

1.2 Objective and Procedure .............................................. 2

2 Preparation and Experimental System 4

2.1 General Preparation .................................................. 4

2.2 Experimental System .................................................. 4

3 Principles, Methods, and Results 7

3.1 Image Pattern of Leukocyte Adhesion ................................. 7

3.2 Image Enhancement by Histogram Equalization Technique ........ 9

3.3 Thresholding ............................................................ 14

3.4 Noise Filtering .......................................................... 14

3.4.1 Modified interpolation .............................................. 16

3.4.2 Morphological filtering .......................................... 21

3.5 Tracking Technique .................................................... 29

4 Discussion and Conclusion 33

4.1 Discussion ............................................................... 33

4.2 Conclusion .............................................................. 36

A: Digital image program 12 37
List of Figures

1 The experimental system .......................................................... 6

2 Flow chart of the procedure to enhance images of leukocyte adhesion 8

3 (a) Original picture of leukocyte adhesion in 256 × 256 matrix, some
of leukocytes are barely visible result from microscope misfocus and
relative motion. (b) The distribution of this histogram is concentrated
in one region, from 70 to 110 in gray scale. ................................. 12

4 (a) After histogram equalization, the hidden information beneath ep-
ithlium are enhanced, simultaneously the noise has been graded up. (b)
The equalized histogram is achieved by spreading effect of histogram-
equalization techniques. ...................................................... 13

5 Using the thresholds techniques to define the gray level of background
to 0, and leave the leukocytes and noise to old values, such process
called semi-thresholding. And the picture show the "salt and pepper"
noise caused by electro-optical component in experimental system. . . . 15

6 The concept of neighborhood(connectedness) ............................. 18

7 Noise is deleted by modified interpolation, the gray level of leukocytes
is assigned to 127, the connected points smaller than 5 are excluded. . 20

8 Effect of erosion from the result of fig. 7, show the outlayer of leuko-
cytes are shrink by a 3 x 3 matrix. .......................................... 23
9 Effect of dilation from the result of fig. 7, show the outlayer of leukocytes are expanded by a $3 \times 3$ matrix. ................. 25

10 Erosion-dilation pairs show different effects on the morphologic sphere X probed by structuring element B. Left hand side tell X shrunked by B first, and dilated by B following, the structure of X is separated into 3 parts, this process we call it opening. Right hand side tell X dilated by B, and shrunked by B in order, the hole in X is filled by B, we call it closing. ...................... 26

11 Use the technique of opening to process the result of fig. 5 can delete the noise in the scene, but will give us false information. Compare with the technique of modified interpolation, some leukocytes have critical size are "killed" by structuring element. ................. 27

12 Use the technique of closing to process the result of fig. 5 show the merge effect of separate objects. .................. 28

13 The counting mechanism of leukocyte adhesion ............. 32
1 Introduction

1.1 Biological background

The adhesion of polymorphonuclear leukocytes (PMN’s) to endothelium is an essential interaction in inflammatory and immunological responses. The control of these processes in the cell membrane of both endothelium and inflammatory cells is poorly understood. To know the nature of these processes well, the determination of PMN adhesion that after using different treatments, increasing or inhibiting adhesive activity, may evaluate the relative importance of endothelium as a regulator of neutrophil adhesion in an intact microcirculation.

In general, local injury immediately causes an influx of PMN’s to the affected area involving a series of complex signaling and recognition processes. Some of the possible causes are local changes in calcium signaling or in membrane potential that may reversibly and instantaneously produce an endothelial pro-adhesive surface for PMN’s at the injured site. Calcium entry into endothelium stimulates a variety of activities, such as the release of prostacyclin, ensothelium-derived relaxing factor, Von Willebrand factor, and platelet activating factor. Endothelial contraction and high permeability to macromolecules in microvessel may also induced by calcium entry.

There are some important findings revealed by Paul and Durán (inpreparation)[7] that in high potassium condition, the adhesive activity of endothelium to PMN’s is
greater than in normal potassium conditions in the coronary microcirculation. Pre-
treatment of the high potassium heart with calcium ionophore A23187 generates anti-
adhesive activity. They proposed that a rapidly synthesized endothelial leukocyte
inhibitory molecule (ELAIM) would participate in the balance between adhesion and
antiadhesion signal in the normal coronary microcirculation, and would serve as an
endogenous anti-inflammatory substance by preventing or decreasing the number of
leukocytes adhering to the endothelium. Their findings also suggest that shear rates,
coronary resistance, and flow rate in coronary vessels did not change the adhesion
significantly, indicating that hemodynamic factors are not important determinants of
leukocyte adhesion.

1.2 Objective and Procedure

The objective of this work is the development of suitable image enhancement
techniques to deblur and analyze the blurred image of fluorescent labeled PMN's. The
blur may in the form of sensor noise, microscope misfocus, relative object- microscope
motion, and so on. In Paul and Durán's work, the researchers analyzed 40 frames
in each heart with the help of a transparent overlay to determine the density of
PMN's per unit area. Such kind of work demands great patience and concentration
in analyzing the experimental data from the video tape.

The developments in this thesis incorporate software for analyzing the experi-
ment data from videotapes frame by frame to release the uncertainty from scene of
the human eye and perseverance of analysis. This software includes techniques of histogram equalization, thresholding, modified interpolation, mathematical morphology and search technique in track to process the segment image. This enhancement process will offer us some important information hidden beneath the rugged surface of the epicardium and yield the PMN's number that appear on a predefined window.
2 Preparation and Experimental System

2.1 General Preparation

Male Wistar-Furth rat hearts were excised and set up as an isolated, perfused preparation. Blood was collected (10 ml/animal) by cardiac puncture in polystyrene tubes containing citrate, and the PMN's were separated from other components. PMN's were fluorescently labeled with 5-carboxy-fluorescein diacetate and infused the heart over 20 min. The number of adhering PMNs/mm² reached a plateau within the first 10 minutes. After a 10 min wash out period, the heart was placed on a specially designed lucite chamber and mounted on the stage of a Nikon Optiphot microscope for intravital microscopy observation. To determine the number of adhering PMN's per unit surface area, the entire left anterior surface of heart was scanned using a 6.3× objective following imaginary lines drawn between the left circumflex artery and apex. [7]

2.2 Experimental System

The fluorescent labeled PMN’s were epilluminated through the microscope object using a 100 W mercury arc lamp with 488 um excitation filter and 515 barrier filter. The image was transmitted to a video camera (Model COHU 4410 SIT camera) and
displayed on a TV monitor and simultaneously transmitted to the memory of a video image digitizer DS20F (Quantex Corp). The output signal from the digitizer can be sent in analog form (EIA standard RS170 video) to a video cassette recorder (Sony 5600) or in digital form over an HP-IB (IEEE 488) bus to a Hewlett-Parkard 1000 series A-900 computer. Video frames in the digitizer can be controlled both manually and by computer.

The DS20F digital video processor provides the capability of digitizing and storing TV images in one field or one frame (33ms) time. Image digitized by the DS20F can originate from any source providing standard RS 170 such as video cameras or video tape recorders. Each video image is resolved into a 512 x 512 matrix of picture elements. The gray scale range is 8 bits per picture element (256 gray scale). The experimental system is shown on fig.1.[4]

Videotapes of fluorescent label PMN's played back frame by frame through the digitizer memory. Digital images of selected frames are then further processed using several image processing techniques. Program IMAG89 was revised to I2 to provide specific usage for leukocyte adhesion image. Processed image can then be recorded on videotape or stored in digital form on the high speed disk (HP 7958,10 M Bytes ) attached to the A-900 computer.
Figure 1: The experimental system
3 Principles, Methods, and Results

3.1 Image Pattern of Leukocyte Adhesion

In image compression or enhancement, the desired output is a picture, an approximation to, or an improved version of, the input picture. Another major branch of picture processing deals with image analysis or scene analysis; here the input is still pictorial, but the desired output is a description of the given picture or scene.[16]

The objective of this work is to use image analysis techniques to develop some software to recognize the fluorescent labeled PMN’s on the screen and determine the number of PMN’s within a predefined window. The softwares generated description refers to specific objects in the picture; thus, it is necessary to segment the picture of leukocyte adhesion and the background must be "scissored out". In order to extract an object from a picture explicitly after image enhancement and segmentation, the objects needed to be clearly defined to discriminate objects from noise. After completing segmentation and filtering, the standard pattern recognition can be applied to the processed image. Fig.2 shows the procedures to process the image model of leukocyte adhesion.

The following are the principles and concepts of digital image techniques that were applied to this analysis.
Figure 2: Flow chart of the procedure to enhances images of leukocyte adhesion
3.2 Image Enhancement by Histogram Equalization Technique

Let the variable \( r \) represent the gray level of the pixels in the fluorescent labeled PMN's image to be enhanced and they lie in the range

\[
0 \leq r \leq 255
\]

with \( r=0 \) representing black and \( r=255 \) representing white in the gray scale. For any \( r \) in the interval \([0,255]\), attention will be focused on a transformation of the form

\[
s = T(r)
\]

(1)

which produce a level \( s \) for every pixel value \( r \) in the original image, and assuming eq(1) satisfies the conditions

(a) \( T(r) \) is single-valued and monotonically increasing in the interval \( 0 \leq r \leq 255 \)

(b) \( 0 \leq T(r) \leq 255 \) for \( 0 \leq r \leq 255 \)

where condition (a) preserves the order from black to white in the gray scale, and condition (b) guarantees a mapping that is consistent with the allowed range of pixel values.

The inverse transformation from \( s \) back to \( r \) will be denoted by

\[
 r = T^{-1}(s) \quad \text{for} \quad 0 \leq s \leq 255
\]

(2)

where it is assumed that \( T^{-1}(s) \) also satisfies conditions (a) and (b) with respect to the variables. The gray levels in an image are random quantities in the interval \([0,255]\). Assuming for a moment that they are continuous variables, the original
and transformed gray levels can be characterized by their probability density function $P_r(r)$ and $P_s(s)$ respectively.

It follows from elementary probability theory that if $P_r(r)$ and $T(r)$ are known, and $T^{-1}(s)$ satisfies condition(a), then the probability density function of the transformed gray level is given by the relation

$$P_s(s) = [P_r(r) \frac{dr}{ds}]_{r=T^{-1}(s)} \tag{3}$$

Now let’s consider the transformation function

$$s = T(r) = \int_0^r P_r(w)dw \tag{4}$$

where $w$ is a dummy variable of integration. The rightmost side of eq(4) is recognized as the cumulative distribution function of $r$. Then from eq(4) the derivative of $s$ with respect to $r$ is given by

$$\frac{ds}{dr} = P_r(r) \tag{5}$$

substituting $\frac{ds}{dr}$ into eq(3)

$$P_s(s) = [P_r(r) \frac{1}{P_r(r)}]_{r=T^{-1}(s)}$$

$$= [1]_{r=T^{-1}(s)}$$

$$= 1 \quad 0 \leq s \leq 255 \tag{6}$$

which is a uniform density in the interval of definition of the transformed variable $s$, using a transformation function equal to the cumulative distribution of $r$ produces an image whose gray levels have a uniform density.
In order to be useful for digital image processing, the concept developed above
must be formulated in discrete form. For gray levels that assume discrete values we
deal with probabilities given by the relation

\[ P_r(r_k) = \frac{n_k}{n} \quad 0 \leq r_k \leq 255 \quad k = 0, \ldots, 255 \quad (7) \]

\( P_r(r_k) \) is the probability of kth gray level, \( n_k \) is the number of pixels in the image.

The discrete form of eq(1) is given by the relation

\[ s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{n} = \sum_{j=0}^{k} P_r(r_j) \quad 0 \leq r \leq 255 \quad 0 \leq k \leq 255 \quad (8) \]

The inverse transformation is denoted by

\[ r_k = T^{-1}(s_k) \quad 0 \leq s_k \leq 255 \quad (9) \]

As a practical illustration of histogram equalization, we applied the subroutine
HSTEQ in I2.(see Appendix) Consider the original image of fluorescent labeled PMN’s
shown in fig. 3(a) which is influenced by microscope misfocus and relative motion,
so that some leukocytes are barely visible. The narrow range of values occupied by
the pixels of this image is evident in the histogram shown in fig.3(b). The equalized
histogram is shown in fig.4(b), and the processed image in fig.4(a); while the equalized
histogram is not perfectly flat throughout the full range of gray levels as expected,
considerable improvement over the original image was achieved by the spreading effect
of the histogram-equalization technique.[9]

The result display explicitly some of the hidden valuable information. With the in-
formation of the enhanced image, that offers us a criteria to perform the segmentation.
Figure 3: (a) Original picture of leukocyte adhesion in $256 \times 256$ matrix, some of leukocytes are barely visible result from microscope misfocus and relative motion. (b) The distribution of this histogram is concentrated in one region, from 70 to 110 in gray scale.
Figure 4: (a) After histogram equalization, the hidden information beneath epithelium are enhanced, simultaneously the noise has been graded up. (b) The equalized histogram is achieved by spreading effect of histogram-equalization techniques.
3.3 Thresholding

The main application of thresholding is the extraction of objects. Simple gray level thresholding is effective in extracting objects when the objects have a characteristic range or set of gray level.[4]

From the image of fluorescent labeled PMN's, the approach to threshold selection is to try a range of thresholds, and choose the one for which the threshold picture has some desired property. After histogram equalization, we can "lock on" some barely visible leukocytes to measure their gray levels and average them, then choose a threshold value adjacent below to this mean value. In image digital image program, subroutine THRH allows the user to assign any gray level range to a specified gray level. For the formula for thresholding

\[ f_z(j,i) = \begin{cases} N & \text{if } f(j,i) \text{ is element of } z \\ \text{unchanged} & \text{otherwise} \end{cases} \quad (10) \]

As fig.5 shows, we assign background gray level equal to 0, and objects (leukocytes) and noise lay on the range [105,255]. After the first of segmentation, now the procedure is noise cleaning.

3.4 Noise Filtering

Two major types of image noise dominate the image of leukocyte adhesion. The first type is an electro-optical component associated with the image formation and
Figure 5: Using the thresholds techniques to define the gray level of background to 0, and leave the leukocytes and noise to old values, such process called semi-thresholding. And the picture show the "salt and pepper" noise caused by electro-optical component in experimental system.
quantization process within the TV camera, videotape recorder and digitizer. The sensed image intensities cannot be recorded without sensor noise. The characteristics of sensor noise are complex function of the sensed intensities. In addition, the characteristics are usually nonlinear so that one cannot assume superposition of signal and noise either in the recorded image or in the reconstruction of the recorded image.

The second type of noise is optical noise due to the experimental system. There are two components of this noise. The first is due to the optical properties of the object observed (the rugged surface of rat heart). The second component is due to the factors associated with lens design and manufacturing, light source, focusing and optical filters. Noise has a distribution over the gray level range 0 to 255. [13]

After optimal thresholding, as shown in fig.5, the techniques for removing "salt-and-pepper" noise in the two value case now can be implemented. In this study, we use two method to remove noise.

3.4.1 Modified interpolation

Before discussing this method we have to define the neighborhood (connectedness) for an object in one certain image. A point \( P=(j,i) \) of a digital picture \( \Sigma \) has four horizontal and vertical neighbors, namely the points

\[
(j-1,i), \quad (j,i-1), \quad (j,i-1), \quad (j+1,i)
\]
We call these points the 4-neighbors of P, and say that they are 4-adjacent to P. In addition, P has four diagonal neighbors, namely

\[(j - 1, i - 1), (j - 1, i + 1), (j + 1, i - 1), (j + 1, i + 1)\]

These, together with the 4-neighbors are called 8-neighbors of P (8-adjacent to P). Note that if P is on the border of \( \Sigma \), some of its neighbors may not exist. In the image pattern of leukocyte adhesion, we define the object is 8-adjacent connectedness. A path \( \pi \) of length n from P to Q in \( \Sigma \) is a sequence of points \( P = P_0, P_1, ..., P_n = Q \) such that \( P_i \) is a neighbor of \( P_{i-1} \), \( 1 \leq i \leq n \). Thus we can speak of \( \pi \) being 8-path. Let an object be a subset of \( \Sigma \), and let \( P, Q \) be points of object. We say that \( P \) is 8-connected to \( Q \) in the object if there exists an 8-path from \( P \) to \( Q \) consisting entirely of points of object. For any \( P \) in the object, the set of points that are connected to \( P \) in the object is called a connected component of the object. [16] From fig.6 two leukocytes lay down on the x-y plane shown in the form of 8-connected component. The concept concerns the geometric relation of noise and object. When the noise consists of isolated points or group of connected points smaller than the size of leukocyte, we can attempt to detect noise points by comparing each point's gray level \( z \) with the level \( z_i \) of its neighbors, if \( z \) is substantially larger (or smaller) than all, or nearly all, of all \( z_i \), we can classify it as a noise point, and remove it by interpolation, i.e. replace \( z \) by the background gray level.

As fig.5 shown in "salt and pepper" noise and the given picture is "two valued", zero and nonzero gray levels. Here we detect the noise point by counting the number
Figure 6: The concept of neighborhood (connectedness)
of their neighbors from which they differ. A white point that has too many ("nearly all") black neighbors can be changed from white to background gray level.

The method makes a forced-choice decision as whether or not the given point is a noise point. If we decide that the point or a group of points is smaller than the size of a leukocyte; in choosing the new gray level for these points, we pay no attention to its old level. On the other hand, if we decide that a group of connected points greater than leukocyte in shape, then we do not change the gray level at all. Using this method to develop the subroutine SMOTH, we can assign the nonzero gray level points to a certain value, further to give the number of neighbors connect to the processing point will delete these points if its' (processing point) neighbors number is smaller than the assign number, for example we assign the gray level to 127 and neighbor number to 4 that the result tell us from fig. 7 means every leukocyte in scene is homogeneous in gray level and "object" smaller than 5-connected component is excluded.

In some cases, if we apply SMOTH subroutine to process sizes of noise or undesired objects, greater than 3 x 3 matrix, these ones will remain. So we need a second method to give us more flexibility to process the image.
Figure 7: Noise is deleted by modified interpolation, the gray level of leukocytes is assigned to 127, the connected points smaller than 5 are excluded.
3.4.2 Morphological filtering

Digital image processing and analysis in the USA has been developed since the 1960s motivated mainly by problems in remote sensing and scene analysis; its main mathematical tools included classic linear filters, Fourier analysis, and statistical or syntactic pattern recognition. Parallel to, but independent from these ideas, mathematical morphology has evolved in Europe since the 1960s as a set-theoretic method for image analysis, motivated mainly by problems in quantitative microscopy. Its mathematical tools are related to integral geometry and stereology. The theoretical foundations of mathematical morphology which stem from Minkowski set operation, also called morphological filters, are more suitable for shape analysis than are linear filters.\[10][11]

Image morphology can be used to reveal the structure of objects by transformation which correspond to shape filtering. An image can be represented by a set of pixels. The morphological operators can be thought to work with two images. The image being processed is referred to as active image and the other image being a kernel is referred to as the structuring element. Each structuring element has a designed shape which can be thought of as a probe or filter of the active image. We can modify the active image by probing it with various structuring elements.

Erosion and dilation are two basic concepts through set operations in morphological filtering.
A. Erosion

Erosion combines two sets using vector subtraction of set elements. If A and B are sets in Euclidean space which elements a and b respectively. \( a = (a_1, \ldots, a_n) \) and \( b = (b_1, \ldots, b_n) \) being N-tuples of elements coordinates, then the erosion of A by b is the set of all elements x for which \( x+b \in A \) for every \( b \in B \). Erosion \( A \ominus B \) can be interpreted as the locus of all center c such that the translation \( B_c \) is entirely contained within the set A. One should be aware that erosion is different from Minkowski subtraction which is the intersection of translations of A by the elements \( b \in B \). Whereas, dilation is identical to Minkowski addition. Some erosion equivalent terms are "shrink" and reduce. The binary erosion of A by B is denoted by \( A \ominus_b B \) and is defined as

\[
A \ominus_b B = \{ x \in E^N \mid x + b \in A \text{ for every } b \in B \} 
\]  

Equivalently, we may write

\[
A \ominus_b B = \bigcup_{b \in B} (A)_{-b} 
\]  

Fig. 8 shows the effect of erosion from fig. 7. The structuring element is a \( 3 \times 3 \) matrix is big enough to delete the noise, but shrink some valuable information.

B. Dilation

Dilation is the morphological dual to erosion, and combines two sets using vector addition of set elements, thus the dilation of A by B is the set of all possible vector

22
Figure 8: Effect of erosion from the result of fig. 7, show the outlayer of leukocytes are shrink by a $3 \times 3$ matrix.
sum of pair of elements, one coming from A and the other from B. In dilation, the roles of the sets A and B are symmetric. Dilation has a local interpretation. That is, if we think of each point a of A as a seed that grows the flower $B_a$ (by placing the origin of $B$ at $a$), and the union of all the flowers is the dilation of $A$ by $B$. Dilation by disk structuring elements corresponds to isotropic expansion algorithm popular to binary image processing. Dilation by small square ($3 \times 3$) is an 8-neighborhood operation easily implemented by adjacently connected array architectures and is the one known by the name "fill", "expand" or "grow". The binary dilation of $A$ by $B$ is denoted by $A \oplus_b B$ and is defined as

$$A \oplus_b B = \{ c \in E^N \mid c = a + b \text{ for some } a \in A \text{ and } b \in B \}.$$  \hspace{1cm} (13)

Equivalently, we may write

$$A \oplus_b B = \bigcup_{b \in B} (A)_b = \bigcup_{a \in A} (B)_a.$$  \hspace{1cm} (14)

From fig.9 the size of leukocyte is expanded using the dilation technique.

Simply speaking in our analysis, suppose that we change all white points to black if they have any white neighbors, and then change all black points to white if they have any white neighbor. The first step shrinks all white region while the second step reexpands them; When the structuring element is set to $3 \times 3$ matrix, a white object that is two point wide or less will disappear completely at the first step, so that the second step cannot restore it. Thus this process deletes not only isolated points, but
Figure 9: Effect of dilation from the result of fig. 7, show the outlayer of leukocytes are expanded by a $3 \times 3$ matrix.
Figure 10: Erosion-dilation pairs show different effects on the morphologic sphere \( X \) probed by structuring element \( B \). Left hand side tell \( X \) shrinked by \( B \) first, and dilated by \( B \) following, the structure of \( X \) is separated into 3 parts, this process we call it \textbf{opening}. Right hand side tell \( X \) dilated by \( B \), and shrinked by \( B \) in order, the hole in \( X \) is filled by \( B \), we call it \textbf{closing}.

also thin line, such process we call \textit{opening}, will give us an effect of fig.11 shown. And some morphological shape leukocytes will divide into many parts.

In practical applications, dilation and erosion pairs are used in sequence, either
dilation of an image followed by the erosion of the dilated result (opening), or erosion followed by dilation (closing). From fig.10 shown us the basic application of morphological filtering with two processing, opening and closing. In either case, the result of iteratively applied dilations and erosions is an elimination of specific image's detail
Figure 11: Use the technique of opening to process the result of fig. 5 can delete the noise in the scene, but will give us false information. Compare with the technique of modified interpolation, some leukocytes have critical size are "killed" by structuring element.
Figure 12: Use the technique of closing to process the result of fig. 5 show the merge effect of separate objects.
smaller than the structuring element without the global geometric distortion of un-suppressed features.[17] In this analysis, from fig.5 known the given picture is" two valued", noise that is smaller than the picture detail can be removed by a process of erosion and dilation. Fig. 12 is the effect of closing comparing with opening that it will merge the nearby spots.

3.5 Tracking Technique

For the purpose of counting the PMN’s in a 256 × 256 window, the image shall turn to bilevel image without noise existing. The image processing techniques we discussed above are for preparing the next procedure.

The basic of this algorithm is from the sequential segmentation, the concept is concerned in processing a point, of the result at previously processed points that is neighborhood (connectedness) base on the same criteria. In these inherently sequential methods, the processing that is performed at a point, and the criteria for accepting it as part of an object, can depend on information obtained from earlier processing of other points, and in particular on the nature and location of the points already accepted as parts of objects.

Using the sequential approach to detect possible object points, once some such points have been detected then more complex computations can be used to track the objects. The detection computations following the processed image can be relatively simple, may have to be performed at every point, since it need only detect every
points of object.

The tracking criterion based on gray level and direction can depend in a comparison between the current point \((j,i)\) and its candidate neighbors, so as to discriminate against candidates whose gray levels etc. do not resemble that of \((j,i)\), if desired, in our analysis, we want to accept all candidate points that satisfy the tracking criterion. And we define that leukocyte is an connected (8-adjacent to point \((j,i)\)) components.

The following is the algorithm for the counting technique in the processed image of leukocyte adhesion.

1. Set up a count main routine to scan the predefined window from the address of \((1,1)\), the raster direction is right and down, accept the first point meeting the detection criterion, and assign this point to be the initial point of a object that is to be tracked.

2. For each object (leukocyte) currently being tracked, apply the appropriate tracking criterion to the points in its acceptance; adjoin the resulting accepted point to the object. This criterion depends on the gray level of the points, if no new points are accepted into the object, tracking of object has terminated.

3. Erase the points of object, return to main count routine to add 1, and starting at adjacent pixel of preceding initial point of object to detect the following object.

4. Repeat procedure (2) and (3).

5. When the bottom row is reached, the count process is complete, and the description yields the number of PMN in a predefined window.
Fig. 13 displays the counting mechanism in term of the concept of connectedness (8-path) of objects to trace the same criteria using recursive program. This algorithm is effective in counting the numbers in fig.7 and the others processed binary images.
Figure 13: The counting mechanism of leukocyte adhesion

**COUNT**

SCAN THE WINDOW FROM THE ADDRESS OF (1,1) 
WHEN (J,J).NE.0 THEN CALL TRACK

**TRACK**

TO FIND THE NEIGHBORS OF(J,J) 
IF (J,J-1).NE.0 CALL TRACK 
IF (J-1,J-1).NE.0 CALL TRACK 
IF (J-1,J).NE.0 CALL TRACK 
IF (J-1,J+1).NE.0 CALL TRACK 
IF (J,J+1).NE.0 CALL TRACK 
IF (J+1,J+1).NE.0 CALL TRACK 
IF (J+1,J).NE.0 CALL TRACK 
IF (J+1,J-1).NE.0 CALL TRACK

AND ERASE THE POINTS WHICH HAVE BEEN TRACKED TILL THE LAST POINT AND ALL ITS' NEIGHBORS ALL EQUAL TO 0 ERASE ITSELF AND RETURN TO COUNT BACK TO THE ADDRESS OF(J,J) SEARCH ANOTHER PIXEL NOT EQUAL TO 0
4 Discussion and Conclusion

4.1 Discussion

In our analysis, if it were to be required to define enhancement, it would be particularly difficult because one man's enhancement may be another man's noise. As a rule, enhancement broadly refers to manipulation of imagery to present to the viewer. Nevertheless, since the target object is the leukocyte, then the situation is obvious to process.

As known, broad categories of enhancement techniques might be listed as follows: (a) intensity mappings or point process; (b) edge sharpening or spatial processes; (c) artifact generation. The method of histogram equalization is classified to points processes, and it has many motivating factors. From a practical viewpoint, motivation for the use of this method arise from the simplicity of the operation as well as potential for real time implementation, additional one is that of attempting the removal of monotonic nonlinearities experienced in our image. Turn to statistical motivated point processes, the image histogram becomes particularly significant. The question often arises as to what enhancement techniques can be utilized when the gray levels are not well distributed by quantizing process. This method is most useful when there exists a finely quantized image from which there are an abundant but narrowly distributed number of quantization levels that can be stretched over a broader but
fewer number of gray shades. The subtle adjacent-in-gray scale changes which occur most often in the original image are stretched to become more obvious at the expense of losing gray levels that occur less frequently.

There are three general argument in favor of histogram equalization. They are (a) maximum zero order entropy. (b) monotonic nonlinearity compensator. (c) preprocessing normalizer. [1] Maximum zero order entropy means that an equalized image presents to the viewer every gray level in an equally likely mode. This is a maximum entropy mode, but may or may not be a desirable operation as fig.4(a) shows. The motivation provided by monotonic nonlinearity compensator is derived from the fact that, for a large number of bits of quantization, the equalized image will be the same as the equalized object, provided the object was passed through strictly monotonic processes. The third motivation of a preprocessing normalizer that each numeric value occurs equally as often as every other value and, as such, guarantees the same amount of image energy as in any other histogram equalized image. This is a particularly useful property for comparing images as inputs to further data extraction algorithms. as next step in thresholding. It is not necessary to simply stick with histogram equalization as the only statistically motivated point process of interest. Naturally an output image could be generated with any distribution desired. The histogram of an image combined with collateral data may suggest other point process mapping, such as histogram modification.[9]

Definitely, spatial processes applied in the Fourier domain, using ramp or other
monotonically increasing functions in the spatial frequency plane is another possible assessment in this analysis.

Obviously, modified interpolation or mathematical morphological filtering, are not suitable for global processes. The greatest advantage of these two noise filtering is that applied together, as in this study, they can be used for any size object, whether it is greater than $3 \times 3$ matrix, or smaller than $3 \times 3$ matrix. The possible methods to delete the noise have also been introduce by many studies. [13] [14] Image enhancement, smoothing and noise filtering by use of local statics or global information have been developed for many years, and it applied to many complicated computation for individual case (picture). Within the experimental data, 40 frames per rat heart, every picture has its unique characteristic of gray level distribution, and there are many diverse differences between frames. It is difficult to find any optimal global parameter to fit all the 40 frames per rat heart. Differences may be caused by the unstable experimental environment, such as vibration of focal plane, focus sharpness judged by the human eye, temperature influence to vidicon camera to affect system gain during longtime experiment, etc. From inspecting the experimental data, use of any local statics or global statics information to delete noise or smoothing will not give us any significant results.

One of the difficulties we encounter is from the experimental system. In the RTE system, the HP machine only offer FORTRAN 77 to develop software. In our analysis, we need to use recursion to do the tracking techniques, (see appendix: subroutine
COUNT) as we know many commonly used programming languages such as FORTRAN, COBOL do not allow recursive programs. The solution to our experimental problem can be derived and stated quite simply using recursive techniques, and can be programmed in FORTRAN or COBOL by simulating the recursive solution using more elementary operation.[18][2]

In general, a nonrecursive version of a program will execute more efficiently in term of time and space than will a recursive version. Sometimes a recursive solution is the most natural and logical way of solving a problem. The conflict between machine efficiency and programmer efficiency is another interesting topic arise from this software developing.

4.2 Conclusion

Image processing algorithms are presented in this work in order to provide biomedical investigators with tools to get results devoid human uncertainty. Further research in this area is to evaluate the accuracy of the number of fluorescent labeled PMN’s experimental data between pre and after processing images. It is also possible to extend the image processing techniques to gray-level morphology method to enhance and extract the object directly.
Appendix

A : Digital image program I2
...is a program to interact with the Quantex Digital Image MEMORY. It is a dialog oriented program, designed to explain the functions and ask for necessary inputs and outputs. Emphasis is made on achieving a multiuse modular and self-documenting design. IMAG91 is a merge of the old IMAG2 and SLAVE programs with the PTOP and FMP calls eliminated and the changes in exec read and write calls the Quantex modified to run on the HP A-900.

Program description:
The mainroutine initializes the program to a defined state. It computes all necessary logical unit numbers on the HP IBus with the given address in the function call. It will check if the power on the digitizer is turned on! While the dimension for the workspace is dynamic (i.e. the dimensions are dependent on loser input) it will ask for the x-y boundaries.

Note: In this program setup the maximum space defined in COMMON /A/ is 256*256!

After this initialization process the program starts its guiding menu to lead you through the program. Before stopping the program the ending sequence will close all opened files.

The parameters are as follows:

IARRY : workspace accessive for the user
IPXL : reserved project space for picture processing
LX,LY : actual line length, number of lines of the image residing in the workspace.
LSX,LSY: starting address of the residing image
LU,LQTX: Logic unit number of the terminal, Quantex, HP2240A

IFLAG,: represents the state of a Writefile
IFLGR : " " " " " " Readfile
-1 : not yet logged on
0 : close open files
1 : no file open
2 : NF option was used; no files will be allowed
> 10 : a file was created and is opened

Note: all these parameters are global and passed via COMMON!
JEQT,IEQT,IMENU are local dummy parameters.

External subroutines:
CONTR,FILER,LOGIN,PROC,MATH,QWRIT,TFMGR
Calling procedure : RU,IMAGXX

This is the main routine of the program.
It initializes and guides the user through a mode menu.

INITIALIZATION

COMPLEX SPECT
COMMON /A/ IARRY(256,256),IPXL(256,256)
COMMON LU,LQTX,ISC,JSC,LUBUS
COMMON /INIT/LX,LY,LSX,LSY,NAPX,NAPY,MAXLY
COMMON /FLAG/IFLAG,IFLGR
COMMON /BUFR/ JBUF(128)
COMMON /STACK/ ISTAK(56),IPTR
COMMON /HISTO/ IHIST(0:255)
LU = LOGLU(LU)
LX = 0
LY = 0
IBELL = 3400B
NAPX = 20
NAPY = 75
MAXLY = -1
IRETR = 0
IPTR = 1
ISWITCH = 0

C->The address of Quantex is set to 4! (Note in LBGET function

LQTX = LBGET(LU,4,ISC,ISWITCH) ! ISWITCH = 0; RETURN
SELECT CODE

C OF QUALIFYING DEVICE. ISWITCH = 1; FIND QUALIFYING DEVICE
WHOSE
C SELECT CODE MATCHES ISC.

ISWITCH = 1
ENDIF
ENDIF
IF (IMENU.EQ.6.AND.IRETR.EQ.1) CALL PROC(IARRY,IRETR)
IF (IMENU.EQ.7) GOTO 30
GOTO 20

C->Ending sequence

30 WRITE (LU,32)
32 FORMAT(/" IMAGE: Ending sequence begun . . .")
IF (IFLAG.LT.10.AND.IFLGR.LT.10) GOTO 34
IFLAG = 0
CALL FILER(0,IARRY)
34 CALL LOGIN
STOP 0001
END

FUNCTION LBGET(LU,IAD,KSC,ISWITCH)
C
------- ------- -- ---

C->Loop to search for QUANTEX bus number

DO 10,LLU=7,63
LBGET = LLU
CALL EXEC(100015B,LLU,I,J,K) ! The no-abort option is used here
GOTO 10 ! This line is executed on error
10 CONTINUE
LBGET = 0

C->Logical match and fooling the compiler

2 IF (IAND(J,37400B).EQ.17400B.AND.IAND(K,37B).EQ.IAD)THEN
IF(ISWITCH.EQ.0)THEN
KSC=IAND(J,77B)
RETURN
ELSE
IF(KSC.EQ.IAND(J,77B))RETURN
ENDIF
ENDIF
10 CONTINUE
LBGET = 0

C->Error message

WRITE (LU,12) IAD
12 FORMAT(/" LU address "I2," cannot be computed on this bus!")
RETURN
END

C
--------------------------
C ** SUBROUTINE TFMGR **
C --------------------------
Transfers a file into memory (or reads from Quantex mem.
It is the read supervisor. It asks for necessary info an
branches to the requested operation.

SUBROUTINE TFMGR(IARRY)

--- ----- ---- -----

EMA IARRY
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION IARRY(256,256)

WRITE (LU,12)
12 FORMAT(//' Would you like to "/
 1" 1. read from Quantex memory (what’s on the
screen),"/
 2" 2. read from an existing data file, or "/
 3" 3. return to main menu ?")
READ (LU,*,ERR=10) IANS
IF (IANS.EQ.1) GOTO 20
IF (IANS.EQ.3) RETURN
IF (IANS.NE.2) GOTO 10
CALL FILER(1,IARRY)
GOTO 40
20 WRITE (LU,22)
22 FORMAT(//' Type 0 to get a screen cursor; type 1000 to
return"/
 1" to the previous menu.")
WRITE (LU,26)
26 FORMAT(//' Type your start address (column, row)? _")
READ (LU,*,ERR=24) LSX,LSY
IF (LSX.EQ.1000.OR.LSY.EQ.1000) GOTO 10
IF (LSX.EQ.0.OR.LSY.EQ.0) CALL CURSR(IARRY)
IF (LSX.GT.512.OR.LSY.GT.512) GOTO 24
IF (LSX.EQ.0.OR.LSY.EQ.0) GOTO 24
30 WRITE (LU,32)
32 FORMAT(//' How many columns and rows? _")
READ (LU,*,ERR=30) LX,LY
IF (LX.GT.256.OR.LY.GT.256) GOTO 30
CALL QDIMR(IARRY)
40 RETURN
END

*********************************************
** SUBROUTINE Q DIM R **
*********************************************

Quantex Digital Image Memory Read...
prepares the array string to set up Quantex for sending
the requested data from a defined location.

SUBROUTINE QDIMR(IARRY)

--- ----- ---- -----

EMA IARRY
COMMON LU,LQTX,ISC,JSC,LUBUS
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION INTO(10),IARRY(256,256),IBUF(256),NPL(5)
JCLMN = LSX
JROW = LSY
DO 10 I=1,LX
ICLMN =JCLMN + I-1
DO 10 J=1,LY
IROW =JROW +J - 1
CALL ADRCV(IROW,ICLMN,NPL)

C->NECESSARY ARRAY VARS

INTO(1) = 00475B
INTO(2) = 210028
INTO(3) = IOR(ISHFT(NPL(1),8),NPL(2))
INTO(4) = IOR(ISHFT(NPL(3),8),NPL(4))
INTO(5) = IOR(ISHFT(NPL(5),8),03B)
CALL ADRCV(0,LX,NPL)
INTO(6) = IOR(ISHFT(NPL(1),8),NPL(2))
INTO(7) = IOR(ISHFT(NPL(3),8),NPL(4))
INTO(8) = IOR(ISHFT(NPL(5),8),017B)
IX = 1

C->Do not send EOI but require it from Q.

CALL EXEC(3,LQTX+2500B,34000B)

C->Send a IFC (interface clear) to get Q. attention

CALL EXEC(3,LUBUS+5100B)
CALL EXEC(2,LQTX+2100B,INTO,8)
CALL EXEC(1,LQTX+100B,IBUF,IX)

10 IARRY(J,I)=IBUF(1)
CALL SEPAR(IARRY)

C->Separate the compressed workspace
RETURN
END

C

************************************************************************
** SUBROUTINE QWRIT **
************************************************************************

...is a control unit for most output functions.

SUBROUTINE QWRIT(IARRY)

--------- ----- -----
EMALARRY
COMMON LU,LQTX,ISC,JSC,LUBUS
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION IARRY(256,256)
IBELL = 3400B

10 WRITE (LU,12)
12 FORMAT//(//" You want to write into Quantex memory."/
1 " To do so I need more information."//
1. Read in an image first.
2. Utilize an available test pattern.
3. Use the data we already have in the program’s memory.
4. Go back into main menu.

Please type the number of your request (1-4)?

READ (LU,*,ERR=10) IREQ
IF (IREQ.EQ.1) GOTO 1111
IF (IREQ.EQ.2) GOTO 2222
IF (IREQ.EQ.3.AND.LX.GT.0) GOTO 3333
IF (IREQ.EQ.3.AND.LX.EQ.0) WRITE (LU,14) IBELL
IF (IREQ.EQ.4) RETURN
GOTO 10

FORMAT(A1,"There isn’t any data in the program’s memory!")

C->Branch to the appropriate request.

1111 CALL TFMGR(IARRY)
    IF (LX.GT.0) GOTO 3333
    GOTO 10

2222 WRITE (LU,30)
30 FORMAT(//" Choose one of the following test patterns:"//
    1" Black dot in white area "/
    2" Vertical line"
    3" Two dots"
    4" Dotted square"
    5" Your selection? _")
    READ (LU,*,ERR=2222) INUM
    IF (INUM.EQ.1000) GOTO 10
    IF (INUM.LT.1.OR.INUM.GT.4) GOTO 2222
    CALL TPAT(IARRY,INUM)

C-> Now definitely have an image

3333 WRITE (LU,42)
42 FORMAT(//' Now we have something to write!'//
    1 ' Select one of the following possibilities:'//
    2 ' 1. Write the image beginning at the predefined location.'//
    3 ' 2. Write the image into a defined window.'//
    4 ' 3. Use the automatic placement routine.'//
    5 ' 4. Store the image in a file.'//
    6 ' 5. Use the cursor to find the best location for the image.'//
    7 ' 6. Go back to menu. _/'
    )
    READ (LU,*,ERR=3333) ISELC
    IF (ISELC.LT.1.OR.ISELC.GT.6) GOTO 2222
    IF (ISELC.EQ.1) CALL WQTX(IARRY)
    IF (ISELC.EQ.2) CALL WINDW(IARRY)
    IF (ISELC.EQ.3) CALL AUTO(IARRY)
    IF (ISELC.EQ.4) CALL FILER(0,IARRY)
IF (ISELC.EQ.5) CALL CURSR(IARRY)
IF (ISELC.EQ.4.OR.ISELC.EQ.5) GOTO 3333
RETURN
END

SUBROUTINE TPAT(IARRY,INUM)
C
------------ ----- -----
EMA IARRY
COMMON /INIT/LX,LY
DIMENSION IARRY(256,256)
LX = 64
LY = 64
DO 10 I=1,64
DO 10 J=1,64
10 IARRY(I,J) = 200
C-> Choose which to do
GOTO (100,200,300,400) INUM
100 DO 120 I=30,35
DO 120 J=30,35
120 IARRY(I,J) = 0
RETURN
200 DO 220 I=31,33
DO 220 J=10,54
220 IARRY(J,I) = 0
RETURN
300 DO 320 I=14,18
DO 320 J=30,35
320 IARRY(J,I) = 0
IARRY(J,I+32) = 0
RETURN
400 DO 420 I = 17 , 47 , 6
DO 420 J = 17 , 47 , 6
IF (I.EQ.17.OR.I.EQ.47.OR.J.EQ.17.OR.J.EQ.47) THEN
  IARRY(I,J) = 0
  IARRY(I,J+1) = 0
  IARRY(I+1,J) = 0
  IARRY(I+1,J+1) = 0
ENDIF
420 CONTINUE
RETURN
END

SUBROUTINE AUTO(IARRY)
C
------------ ---- ----
EMA IARRY
COMMON LU
COMMON /INIT/ LX,LY,LSX,LSY,NAPX,NAPY,MAXLY
COMMON /STACK/ ISTAK(56),IPTR
DIMENSION IARRY(256,256)
C->Initialize

ILFT = 20
IRT = 450
ITOP = 75
IBTM = 475
IFLAG = 0

C->If NAPX & NAPY (20,75) then just go to 400 (write to left top)

IF (NAPX.EQ.20.AND.NAPY.EQ.75) GOTO 400

C->Menu

10 WRITE (LU,12)
12 FORMAT(/"Auto Placement Menu"/

1" 1. Place the image in the top left corner of the screen."

2" 2. Overwrite the previous image."

3" 3. Write the image to the next free area on the screen."

4" 4. Overwrite one of the other preceding images."/

READ (LU,*,ERR=10) IANS
IF (IANS.LT.1.OR.IANS.GT.4) GOTO 10
GOTO (100,200,400,300) IANS

C-> Clear stack

100 IPTR = 1
NAPX = ILFT
NAPY = ITOP
GOTO 400

C-> Pop one, write to that location

200 CALL POP (NAPX,NAPY)
GOTO 400

C-> Pop until correct one found, overwrite

300 WRITE(LU,302)
302 FORMAT(/"Which image, counting in the order in which the were"/

1"put up? _")
READ (LU,*) IANS
KOUNT = (( IPTR - 1) / 2) - IANS + 1
DO 310 I=1,KOUNT
310 CALL POP (NAPX,NAPY)

C->Check to see if new row needed
400  NN = NAPX + LX
     IF (NN.GT.IRT) THEN
          NAPX = IFLT
          NAPY = NAPY + MAXLY + 5
     ENDIF

C->Check to see if bottom out of range

     NN = NAPY + LY
     IF (NN.GT.IBTM) THEN
          ISLY = LY
          LY = IBTM - NAPY
          IFLAG = 1
     ENDIF

C->Store old values of LSX,LSY; write to screen

     ISLSX = LSX
     ISLSY = LSY
     LSX = NAPX
     LSY = NAPY
     CALL WQTX(IARRAY)

C->Restore old values

     NAPX = LSX
     NAPY = LSY
     LSX = ISLSX
     LSY = ISLSY
     IF (IFLAG.EQ.1) LY = ISLY
     CALL PUSH(NAPX,NAPY)

C->Compute new NAPX

     NAPX = NAPX + LX + 5

C->Check for new MAXLY

     IF (LY.GT.MAXLY) MAXLY = LY
     RETURN
     END

SUBROUTINE PUSH(IVAL,JVAL)

C

-------------------- ---- ---- ----
COMMON /STACK/ ISTAK(56),IPTR
ISTAK(IPTR) = IVAL
IPTR = IPTR + 1
ISTAK(IPTR) = JVAL
IPTR = IPTR + 1
RETURN
END

SUBROUTINE POP(IVAL,JVAL)

C

-------------------- --- ---- ----
COMMON /STACK/ ISTAK(56),IPTR
IPTR = IPTR - 1
JVAL = ISTAK(IPTR)
IPTR = IPTR - 1
IVAL = ISTAK(IPTR)
RETURN
END

******************************
** SUBROUTINE WQTX **
******************************

Write to Quantex memory
...initializes the Quantex for write procedures.

It sets up the outputstring in Quantex "NIBBLE" format
then it puts out the dimensioned IARRY.

SUBROUTINE WQTX(IARRY)

EMA IARRY
COMMON LU,LQTX,ISC,JSC,LUBUS
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION NPL(5),INTO(5),ITEMP(128),IARRY(256,256)

CALL COMB(IARRY)
ICLMN = LSX
JROW = LSY
NLX=INT((LX+1)/2.)
DO 10 J=1,LY
  IROW = JROW + J - 1
  CALL ADRCV(IROW,ICLMN,NPL)
  INTO(1) = 00461B
  INTO(2) = 21002B
  INTO(3) = IOR (( ISHFT (NPL(1),8)) , NPL(2))
  INTO(4) = IOR (( ISHFT (NPL(3),8)) , NPL(4))
  INTO(5) = IOR (( ISHFT (NPL(5),8)) , 0017B)

DO 5 J1=1,LX,2
  JJ=INT((J1+1)/2.)
  ITEMP(JJ)=IARRY(J,J1)
5  CONTINUE

CALL EXEC(3,LUBUS+5100B)
CALL EXEC(2,LQTX+2100B,INTO,5)
10 CALL EXEC(2,LQTX+2100B,ITEMP,NLX)
C 10 WRITE (LQTX) (INTO(I),I=1,5),(IARRY(J,J1),J1=1,LX,2)
CALL SEPAR(IARRY)
RETURN
END

*****************************************************************************
** SUBROUTINE W I N D W **
*****************************************************************************

Description:
window replaces the dormant workspace to any defined position in the memory(=screen).

SUBROUTINE WINDW(IARRY)

--- ----- ----

EMA IARRY
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION IARRY(256,256)

C->Store the original window

ISLX = LX
ISLY = LY
ISLSX = LSX
ISLSY = LSY

C->Dialog, and do the window

10 WRITE (LU,12)
12 FORMAT(" Type your new start address (column,row)? _")
READ (LU,*,ERR=10) LSX,LSY
IF (LSX.EQ.1000) RETURN
IF (LSX.GT.512.OR.LSY.GT .512) GOTO 10
20 WRITE (LU,22)
22 FORMAT(" Type length of field (x,y)? _")
READ (LU,*,ERR=20) LX,LY
IF (LX.GT.256.OR.LY.GT.256) GOTO 20
CALL WQTX(IARRY)
WRITE (LU,24)
24 FORMAT(" Would you like to save this image? _")
READ (LU,26) IANSW
26 FORMAT(A1)
IF (IANSW.EQ.1HY) CALL FILER(0,IARRY)

C->Restore the original window

LX = ISLX
LY = ISLY
LSX = ISLSX
LSY = ISLSY
RETURN
END

*****************************************************************************
** SUBROUTINE CURSR **

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

Description:
Cursor display on the monitor. A dot on the monitor, at the defined position, will appear. After typing G (go) the cursor is removed and the original data is written back.

SUBROUTINE CURSR(IARRY)

--------- ------- -------

EMA IARRY
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION IIRST(3,3),IARRY(256,256)
ISLSX = LSX
ISLSY = LSY
LX = 10
LY = 10
ISLX = LX
ISLY = LY

C->Save the IARRY contents of the cursor position

DO 2 I=1,LY
  DO 2 J=1,LX
  ISAVE(I,J) = IARRY(I,J)
10  WRITE (LU,12)
12  FORMAT(/" Type the cursor address (column,row)? ",/)

C->Read the cursor position and store the memory contents in IRST.

READ (LU,*), LSX,LSY
IF (LSX.EQ.1000) RETURN
CALL QDIMR(IARRY)
ICC = 0
IF (IARRY(1,1).LT.128) ICC = 255
LX=3
LY=3
DO 20 I=1,LX
  DO 20 J=1,LY
    IIRST(J,I) = IARRY(J,I)
20  IARRY(J,I) = ICC

C->Write the cursor to monitor

CALL WQTX(IARRY)
WRITE (LU,30)
30  FORMAT(/" Hit any key and <return> to continue! ")
32  READ (LU,34) IANSW
34  FORMAT(A1)
    IF (ITLOG(IANSW).EQ.0) GOTO 32
C->Restore the monitor greycolours & orig. IARRY values

DO 40 I=1,LX
DO 40 J=1,LY
40
IARRY(J,I) = IRST(J,I)
CALL WQTX(IARRY)
C
C
DO 50,I=1,LY
C
DO 50,J=1,LX
C 50 IARRY(I,J) = ISAVE(I,J)
C->Restore the original window

LX = ISLX
LY = ISLY
LSX = ISLSX
LSY = ISLSY
RETURN
END
SUBROUTINE SCALR(IARRY,IPXL)
C
EMA IARRY,IPXL
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION IARRY(256,256),IPXL(3,256)
ISLSX = LSX
ISLSY = LSY
ISLX = LX
ISLY = LY
IANS = 1HX
1
LX = 170
LY = 1
J1 = 0
ICC = 300
C->Let’s open our textbooks to page 1, class . . .

10 WRITE (LU,12)
12 FORMAT(/"Input row number: _")
READ (LU,*,ERR=10) LSY
IF (LSY.EQ.1000) RETURN
IF (LSY.LT.1.OR.LSY.GT.512) GOTO 10
14 IF (IANS.NE.1HX) GOTO 19
WRITE (LU,16)
16 FORMAT(/"White or black line (W/B) : _")
READ (LU,18) IANS
18 FORMAT(A1)
19 IF (IANS.EQ.1HW) ICC = 170
IF (IANS.EQ.1HB) ICC = 30
IF (ICC.GT.255) THEN
IANS = 1HX
GOTO 14
ENDIF

C->Here we go Quantex, here we go!! (knock wood)

DO 22 I=1,3
LSX = 2 + 170 * (I - 1)
CALL QDIMR(IARRY)
DO 20 J=1,170
J1 = J + 170 * (I - 1)
IPXL(I,J) = IARRY(1,J)
IARRY(1,J) = ICC
RJ1 = FLOAT(J1)
IF (J1 / 5.EQ.RJ1 / 5.0) IARRY(1,J) = 100 - IDIM(50,ICC)
IF (J1 / 50.EQ.RJ1 / 50.0) IARRY(1,J) = 250 * IDIM(31,IC

20 CONTINUE
CALL WQTX(IARRY)
22 CONTINUE
WRITE (LU,30)
30 FORMAT(/"Right row (Y/N) : _")
READ (LU,18) JANS
IF (JANS.EQ.1HY) GOTO 50

C->Wheah yu thin yu goin wit dat!

40 DO 44 I=1,3
LSX = 2 + 170 * (I - 1)
DO 42 J=1,170
IARRY(1,J) = IPXL(I,J)
42 CONTINUE
CALL WQTX(IARRY)
44 CONTINUE
IF (JANS.EQ.1HY) GOTO 70
GOTO 1

C->He likes it - hey Mikey!

50 WRITE (LU,52)
52 FORMAT(/"Please locate the scalar nearest to your"/
1"starting point; count the number of markers and input
(Note:"/
2"the lightest/darkest markers occur at 10,20,30,etc.")
READ (LU,*),IPT
ICOL = 5 * IPT + 1
54 WRITE (LU,56)
56 FORMAT(/"Please type in whether your starting point is"/
1" 1. Far Left"/
2" 2. Near Left"/
3" 3. Center"/
4" 4. Near Right"/
5" 5. Far Right"/
6"Enter the corresponding number : _")
READ (LU,*),IEXCT
ICOL = ICOL + IEXCT - 3
C->You tell 'em---No,you(!) tell 'em!

WRITE (LU,60) ICOL,LSY
60 FORMAT(/"The address of your starting pixel is
("I3","I3,"./
1"Take the starting point of your window accordingly!")
GOTO 40

C->I'us dard as a dooornail, but he ressuurreecctted me!

70 LSX = ISLSX
LSY = ISLSY
LX = ISLX
LY = ISLY
RETURN
END

C

*************************
C

** SUBROUTINE GRAB **
C

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

SUBROUTINE GRAB

----------- -----

COMMON LU,LQTX,ISC,JSC,LUBUS
COMMON /INIT/ LX,LY,LSX,LSY,NAPX,NAPY,MAXLY
DIMENSION INTO(2)

C->Reset NAPX & NAPY for new screen

NAPX = 50
NAPY = 75
MAXLY = -1

C->Necessary array vars for quantex

INTO(1) = 00442B
INTO(2) = 07777B
CALL EXEC(3,LUBUS+5100B)
CALL EXEC(2,LQTX+2100B,INTO,-3)
CALL EXEC(3,LUBUS+5100B)
RETURN
END

C

****************************
C

** SUBROUTINE LOGIN **
C

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

SUBROUTINE LOGIN
C

---------- -----

52
This subroutine, 'LOGIN', just prints a header and a closing statement on logging off.

COMMON LU
COMMON /FLAG/IFLAG, IFLGR
IF (IFLAG.EQ.-1) GOTO 30

C->Logging off

WRITE (LU,2)
2 FORMAT(" IMAGE: Logging off now . . . bye!"//)
RETURN

C->Logging on

30 WRITE (LU,32)
32 FORMAT(//////" IMAGE: A data acquisition and processing program"/
1" servicing the Quantex digital image memory in communication "/
2" with the Hewlett-Packard 1000 computer system."/)
CALL EXEC(12,0,2,0,-1) ! wait 1 sec, then continue
IFLAG = 1
IFLGR = 1
RETURN
END

****************************
** SUBROUTINE ADRCV **
****************************

SUBROUTINE ADRCV(IRW, ICLMN, NPL)

Address Conversion takes a given row and column, and calculates the Quantex memory address or pixelcount. As output a array in nibble QUANTEX format is provided.

Remark: the Quantex memory has a 18 Bit address. Since we are working on a 16 Bit integer base we prevent overflow by dividing the row into blocks of 128 lines.
If the row number is >512 the program will print an error message and terminate the program!

Input : IRW, ICLMN
Output : NPL(1-5)

DIMENSION NPL(5)
IROW = IRW
DO 10, I=1,4
J = I-1
IF (IROW.LE.128) I = 4
IROW = IROW - 128

53
10 CONTINUE  
NPL(5) = J + 000140B  
IROW = IROW + 128  
IF (IROW.LE.128) GOTO 20  
WRITE (LU,12)  
12 FORMAT("Row too big!!!")  
IFLAG = 0  
CALL LOGIN  
  
C->Calculate the 16 Bit address and combine the 4 Bit data wit
C 4 Bit Quantex control data (2-6).  
20 IF (IROW.EQ.0) IROW = 1  
IADR = ((IROW-1) * 512) + ICLMN  
ISHRG = IADR  
DO 30 I=1,4  
J = I + 1  
NPL(I) = IAND (ISHRG,017B)  
JJ = ISHFT(J,4)  
NPL(I) = IOR (NPL(I),JJ)  
ISHRG = IADR  
ISHRG = ISHFT (ISHRG,-4)  
IADR = ISHRG  
30 CONTINUE  
RETURN  
END  
  
C  
********************************  
C ** SUBROUTINE COMBINE **  
C ********************************  
C  
C combines 2 adjacent integers in a 16 bit word (2n-1 spaced)  
C  
C Input : IARRY containing 0 to 255 integers  
C Output : IARRY with compressed contents.  
C  
SUBROUTINE COMB(IARRY)  
--- ---- ---- ----  
EMA IARRY  
COMMON /INIT/LX,LY,LSX,LSY  
DIMENSION IARRY(256,256)  
DO 111 I=1,LX  
DO 111 J=1,LY  
J1 = J+1  
111 IARRY(I,J) = IOR (ISHFT (IARRY(I,J),8) , IARRY(I,J1))  
RETURN  
END  
  
C  
*************************************************************  
C ** SUBROUTINE SEPARATE **  
C *************************************************************  
C  
C Separates the two compressed pixelintegers (16 bit)
Input: IARRY containing a 2*8bit information in the (2n-1) field location.

Output: IARRY containing 0 to 255 integers.

SUBROUTINE SEPAR(IARRY)

---

EMA IARRY
COMMON /INIT/LX,LY,LSX,LSY
DIMENSION IARRY(256,256)
DO 111 I=1,LY
  DO 111 J=1,LX,2
    ISAVE = IARRY(I,J)
    J1 = J+1
    IARRY(I,J) = ISHFT (IARRY(I,J) , -8)
    IARRY(I,J1) = IAND (ISAVE,0377B)
111 CONTINUE
RETURN
END

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

** SUBROUTINE FILER **
******************************************************************************

Description:
does all file-handling in a dialog mode. It keeps track of opened files with FLAG and IFLGR, and allows only one open file at a time.

A data compression is done to save memory (see COMB).

call proc. : CALL FILER(IRQST,IARRY) & COMMON's

SUBROUTINE FILER(IRQST,IARRY)

---

EMA IARRY
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
COMMON /FLAG/IFLAG,IFLGR
COMMON /BUFR/JBUF(128)
DIMENSION INBUF(64),IARRY(256,256),JTIME(15),LBUF(128)
DIMENSION IHEAD(16),ITIME(5)
DATA IHEAD/16*1H /
CALL LGBUF(JBUF,128)

C->control to read or write sections

IF (IRQST.EQ.1) GOTO 500

C->write section

   IF (IFLGR.GT.10) CLOSE
   (88,IOSTAT=IERR,ERR=999,STATUS='KE')
IF (IFLAG.GT.10) GOTO 100 ! file is already open, ready to be written
IF (IFLAG.EQ.0) GOTO 200 ! file needs to be closed

C->dialog with loser

IFLAG = IFLAG + 10
WRITE (LU,2)
2 FORMAT(//////"IMAGE: Program Filer"///)
WRITE (LU,4)
4 FORMAT("Please input file name: _")
READ (LU,6) (INBUF(I),I=1,64)
6 FORMAT(64A2)

C->open file

OPEN(88,IOSTAT=IERR,ERR=999,FILE=INBUF,STATUS='UN')
WRITE (LU,8)
8 FORMAT(/"IMAGE: file opened successfully")

C->puts header on file

WRITE (88,10,IOSTAT=IERR,ERR=999)
10 FORMAT(1X,10HIMAGE FILE)
CALL FTIME(JTIME)
WRITE (88,12,IOSTAT=IERR,ERR=999) (JTIME(I),I=1,15)
12 FORMAT(15A2)

C->get system time; store msecs with size of image

CALL EXEC(11,ITIME)
100 ENCODE(32,102,IHEAD) ITIME(2),LX,LY,LSX,LSY
102 FORMAT(5I4)
WRITE (88,104,IOSTAT=IERR,ERR=999) (IHEAD(I),I=1,16)
104 FORMAT(16A2)
WRITE (LU,106)
106 FORMAT(/"IMAGE: parameters written to file")

C->compress data; write to file one line at a time

CALL COMB(IARRY)
IFLAG = 11
NEWLX = (LX + 1) / 2
DO 110 I=1,LY
DO 108 J=1,LX,2
108 LBUF((J + 1) / 2) = IARRY(I,J)
110 WRITE (88,112,IOSTAT=IERR,ERR=999) (LBUF(K),K=1,NEWLX)
112 FORMAT(128A2)
WRITE (LU,114)
114 FORMAT(/"IMAGE: data written to file ")

C->separate data

CALL SEPAR(IARRY)
C->close file

200  CLOSE(88, IOSTAT=IERR, ERR=999, STATUS='KE')
     WRITE (LU,202)
202  FORMAT(/"IMAGE: file closed successfully")
     IF (IFLAG.GT.10) IFLAG = IFLAG-10
     IF (IFLGR.GT.10) IFLGR = IFLGR-10
     RETURN

C->READ SECTION
C ---- ------

500  IF (IFLAG.GT.10) CLOSE(88, IOSTAT=IERR, ERR=999, STATUS='KE')
     IF (IFLAG.EQ.0) GOTO 200  ! close up & get out
     IF (IFLGR.GT.10) GOTO 600  ! file is open already

C->open file

     IFLGR = IFLGR+10
     WRITE (LU,2)
     WRITE (LU,4)
     READ (LU,6) (INBUF(I),I=1,64)
     OPEN (88, IOSTAT=IERR, ERR=999, FILE=INBUF, STATUS='OL')

C->read header

     WRITE (LU,8)
600  READ (88,602, IOSTAT=IERR, ERR=999) (IHEAD(I),I=1,16)
602  FORMAT(16A2)
     WRITE (LU,602) (IHEAD(I),I=1,16)
     READ (88,602) (IHEAD(I),I=1,16)
     WRITE (LU,602) (IHEAD(I),I=1,16)

C->assign file data to LX,LY,etc.

     READ (88,604, IOSTAT=IERR, ERR=999) (IHEAD(I),I=1,10)
604  FORMAT(10A2)
     DECODE (32,606,IHEAD) ITIME(2),LX,LY,LSX,LSY
606  FORMAT(5I4)

C->read file data

     NEWLX = (LX + 1) / 2
     DO 610 I=1,LY
         READ (88,112, IOSTAT=IERR, ERR=999) (LBUF(J),J=1,NEWLX)
         DO 608 J=1,LX,2
608    IARRY(I,J) = JBUF((J + 1) / 2)
     CONTINUE

C->separate workspace

     CALL SEPAR(IARRY)
     WRITE (LU,612)

57
612 FORMAT("IMAGE: file read successfully")
GOTO 200
C->if error occurs, call FIERR routine
999 CALL FIERR(IERR)
RETURN
END
SUBROUTINE FIERR(IERR)
COMMON LU
IF (IERR.EQ.462. OR. IERR.EQ.506) THEN
WRITE (LU,10)
ELSE IF (IERR.EQ.459. OR. IERR.EQ.508) THEN
WRITE (LU,11)
ELSE IF (IERR.EQ.502) THEN
WRITE (LU,12)
ELSE IF (IERR.EQ.507) THEN
WRITE (LU,13)
ELSE IF (IERR.EQ.514) THEN
WRITE (LU,14)
ELSE IF (IERR.EQ.515) THEN
WRITE (LU,15)
ELSE IF (IERR.EQ.532) THEN
WRITE (LU,16)
ELSE IF (IERR.EQ.533) THEN
WRITE (LU,17)
ELSE IF (IERR.EQ.546) THEN
WRITE (LU,18)
ELSE
WRITE (LU,19) IERR
ENDIF
10 FORMAT("IMAGE: File not found")
11 FORMAT("IMAGE: File already open")
12 FORMAT("IMAGE: Duplicate file name")
13 FORMAT("IMAGE: Wrong security code")
14 FORMAT("IMAGE: Directory full")
15 FORMAT("IMAGE: Illegal file name")
16 FORMAT("IMAGE: Cartridge not found - probably not mounted")
17 FORMAT("IMAGE: No room on cartridge")
18 FORMAT("IMAGE: File has too many extents")
19 FORMAT("IMAGE: File handling error "I3," occurred.")
WRITE (LU,100)
100 FORMAT("Would you like to continue with the program,")
1"or stop here? _")
102 READ (LU,104) IANS
104 FORMAT(A1)
IF (IANS.EQ.1HC) RETURN
IF (IANS.NE.1HS) THEN
WRITE (LU,106)
SUBROUTINE PROC(IARRY,IRETR)

C
-------- ---- ---- ----

COMPLEX SPECT
EMA IARRY,IPXL,SPECT
COMMON LU
COMMON /INIT/LX,LY
DIMENSION
ITAG(20),IARRY(256,256),IPXL(256,256),SPECT(256,256)
DIMENSION IOFF(36)
IBELL = 3400B
ITAG(4) = LX
ITAG(5) = LY
ITAG(10) = 0
IF (LX.EQ.0.OR.LY.EQ.0) WRITE (LU,10)
10 FORMAT(///" Warning !!!! there is no workspace defined.")
20 WRITE (LU,22)
22 FORMAT(///" Here are the goodies for picture processing!
1" The following routines work on your given workfield"/
2" Note: Some of the routines will change the picture"/
3" information. If you want to keep your original"/
4" it is recommended to go back to menu and store"/
5" the workfield first!")
24 WRITE (LU,26)
26 FORMAT(" Choose one of the following: "/
1" 1. Neighborhood Enhancement-Background spot
elimination. "/
2" 2. Expand/Compress the graycolor range of the image.
3" 3. Histogram equalization."/
4" 4. Smoothing by average filtering."/
5" 5. Enhancement for more edge contrast. "/
6" 6. Enhancing with laplacian."/
7" 7. Kirsch Edge Detection Algorithm."/
8" 8. Butterworth high-pass filter."/
9" 9. Homomorphic filtering."/
CC &" 10. Power Spectrum equalization."/
CC 1" 11. Weiner filtering."/
2" 12. Signal Averaging."/
3" 13. Subtraction of two windows."/
4" 14. Threshold filtering."/
5" 15. Display the histogram."/
6" 16. Percentage of image in grayscale range."/
7" 17. Pause."/
8" 18 Smooth the noise."/
9" 19. Count the number of leukocyte adhesion."/
WRITE (LU,28) IBELL ! Ring bell
FORMAT(A1)
READ (LU,*,ERR=24) ITAG(1)
IF (ITAG(1).EQ.22) GOTO 999
IF (ITAG(1).GT.20.OR.ITAG(1).LT.1) GOTO 20
IF (IRETR.EQ.2.AND.ITAG(1).LT.14) GOTO 24
IF (ITAG(1).GE.2) GOTO 40
WRITE (LU,32)
FORMAT(/"Please enter lower & upper threshold boundaries ")
READ (LU,*,ERR=30) ILTB,IUTB
IF (ILTB.EQ.1000) GOTO 24
IF (ILTB.LT.0.OR.IUTB.GT.255.OR.ILTB.GT.IUTB) GOTO 30
ITAG(2) = ILTB
ITAG(3) = IUTB
CALL NGHBR(IARRY,IPXL,ILTB,IUTB)
GOTO 24
IF (ITAG(1).NE.2) GOTO 50
C->Get old maximum value; store in ITAG(2)
WRITE (LU,42)
FORMAT(/"Please enter the desired maximum graycolor ")
READ (LU,*,ERR=40) MAXGCL
IF (MAXGCL.EQ.1000) GOTO 24
IF (MAXGCL.LE.0.OR.MAXGCL.GT.255) GOTO 40
ITAG(2) = MAXGCL
CALL EXPND(IARRY,MAGCL)
GOTO 24
IF (ITAG(1).EQ.3) CALL HSTEQ(IARRY)
IF (ITAG(1).EQ.4) CALL FILTR(IARRY,IPXL)
IF (ITAG(1).EQ.5) CALL LAPL(IARRY,IPXL)
IF (ITAG(1).EQ.6) CALL HOLPL(IARRY,IPXL)
IF (ITAG(1).GE.12) GOTO 70
IF (ITAG(1).LE.7) GOTO 24
C->Here user decides which transform to use
WRITE (LU,62)
FORMAT(/"Which transform would you like to use:"/
1") 1- Fourier Transform"
2") 2- Walsh Transform"
3") 3- I don’t know what the fuck you’re talking about."/
READ (LU,*,ERR=60) ITAG(3)
ITAG(2) = ITAG(1) ! This lets SHELL know which routine use.
IF (ITAG(3).GE.1.AND.ITAG(3).LE.2) GOTO 68
WRITE (LU,64)
Unfortunately, the filter you've chosen uses a procedure known as a Fast Fourier Transform. This process is wonderfully efficient when using an image whose size is a power of two (e.g., 64x64 or 128x128), but horrible for other sizes. If your image is not a power of two in size, please don't try this filter. This program will die a slow, painful death and your picture will be lost forever.

CALL EXEC (12,0,2,0,-5) ! Give user a few seconds to digest the bad news

C->Store old values of LX and LY

LX1 = LX
LY1 = LY
IF(ITAG(1).GE.8.AND.ITAG(1).LE.11) CALL SHELL(IARRY,SPECT,ITAG)
GOTO 24

C->Here we take care of all the interactive routines

IF (ITAG(1).EQ.12) CALL SAVRG(IARRY)
IF (ITAG(1).EQ.13) CALL SDIFF(IARRY)
IF (ITAG(1).EQ.14) CALL THRH(IARRY)
IF (ITAG(1).EQ.15) CALL HIST(IARRY)
IF (ITAG(1).EQ.16) CALL PCT(IARRY)
IF (ITAG(1).EQ.17) CALL EXEC(7)
IF (ITAG(1).EQ.18) CALL SMOTH(IARRY)
IF (ITAG(1).EQ.19) CALL COUNT(IARRY)
IF (ITAG(1).EQ.20) CALL ERSON(IARRY,IPXL)
IF (ITAG(1).EQ.21) CALL DATON(IARRY,IPXL)
GOTO 24

C->Pause routine

PAUSE 1
GOTO 24

RETURN
END

SUBROUTINE SAVRG(IARRY)

EMA IARRY,IPXL
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
COMMON /FLAG/ IFLAG
DIMENSION IARRY(256,256),IPXL(256,256)
IMAX = 0
IMIN = 255

61
C->Feh!

10 WRITE (LU,12)
12 FORMAT(/"All windows must be of same size and saved in ",1"data files for signal averaging - like to continue (Y/N ",")
READ (LU,14) IANS
14 FORMAT(A1)
IF (IANS.EQ.1HN) RETURN

C->Get count here

20 WRITE (LU,22)
22 FORMAT(/"How many windows will be averaged? ",")
READ (LU,*,ERR=20) KOUNT
IF (KOUNT.LT.2) GOTO 20

C->CALL THE PLUMBER,DEAR! (Seriously, call FILER to get LX & L

CALL FILER(1,IARRY)

DO 5 I = 1,LX
DO 5 J = 1,LY
IPXL(J,I) = IARRY(J,I)
5 CONTINUE
C
C NOW GET THE OTHER ARRAY
C
DO 40 I = 2, KOUNT
WRITE (LU,32) I
32 FORMAT(/"Please repeat for file ",")
IFLAG = 1
CALL FILER(1,IARRY)
DO 25 J = 1 , LY
DO 15 12= 1 , LX
IPXL(I2,J) = IPXL(I2,J) + IARRY(I2,J)
15 CONTINUE
25 CONTINUE
40 CONTINUE
C
C NOW WE DIVIDE BY KOUNT
C
DO 30 I=1,LX
DO 30 J=1,LY
IARRY(J,I) = IPXL(J,I) / KOUNT
IF (IARRY(J,I).GT.IMAX) IMAX = IARRY(J,I)
IF (IARRY(J,I).LT.IMIN) IMIN = IARRY(J,I)
30 CONTINUE
C->Wanna store this shit?

60 WRITE (LU,62)
62 FORMAT(/"Would you like to store this image in a file no ",")
READ (LU,14) IANS
IF (IANS.EQ.1HN) RETURN
IFLAG = 1
CALL FILER(0,IARRY)
RETURN
END
SUBROUTINE SDIFF(IARRY)
C
EMA IARRY,IPXL
COMMON LU
COMMON /INIT/LX,LY,LSX,LSY
COMMON /FLAG/ IFLAG
DIMENSION IARRY(256,256),IPXL(256,256)
IMAX = 0
IMIN = 255
C->Make sure loser has prepared to use this routine
10 WRITE (LU,12)
12 FORMAT(/"IMAGE: This image subtraction routine requires that"
  1"both images involved must be the same size, and stored"
  2"in data files."//
  3"Is it ok to continue at this point? _")
READ (LU,14) IANS
14 FORMAT(A1)
IF (IANS.EQ.1HN) RETURN
C->Here we get the first file
20 WRITE (LU,22)
22 FORMAT(/"This program will now ask you for the two filenames - "//
  1"Please keep in mind that you may put either one in first,"//
  2"since the program will take the absolute value of the"//
  3"difference."///)
IFLAG = 1
CALL FILER(1,IARRY)
C STORE THE FIRST ARRAY
DO 8 I=1,LX
DO 8 J=1,LY
IPXL(J, I)=IARRY(J, I)
8 CONTINUE
C->Now we do the second file
IFLAG = 1
CALL FILER(1,IARRY)
WRITE (LU,52)
52 FORMAT(/"Crunch, crunch, crunch...")
C
DO 30 I=1,LX
DO 30 J=1,LY
30 CONTINUE
IARRAY(J,I) = IABS(IARRAY(J,I) - IPXL(J,I))
IF (IARRAY(J,I) .GT. IMAX) IMAX = IARRAY(J,I)
IF (IARRAY(J,I) .LT. IMIN) IMIN = IARRAY(J,I)
30 CONTINUE

C -> Files?

70 WRITE (LU,72)
72 FORMAT(/"Would you like to store this image in a file no 
")
READ (LU,14) IANS
IF (IANS.EQ.1HN) RETURN
IFLAG = 1
CALL FILER(0,IARRAY)
RETURN
END

C

--------- ----- ---- ----

LU = LOGLU(LU)
IFUNC = ITAG(8)
WRITE (LU,10) IERR,IFUNC
10 FORMAT(/"IMAGE: Signal Averaging error "I6," occurred."
1"The function was "I2," . This can be deciphered by"
2"looking at the slave program"
3"Would you like to continue with the program, or stop"
4"here? 
")
11 READ (LU,12) IANS
12 FORMAT(A1)
IF (IANS.EQ.1HC) RETURN
IF (IANS.EQ.1HS) STOP 0007
WRITE (LU,14)
14 FORMAT(/"Please answer the question with C or S 
")
GOTO 11
END

--------- ---- ---- ----

EMA IARRAY
COMMON LU
COMMON /INIT/LX,LY
DIMENSION
IARRAY(256,256), ISTGC(10), ISUTL(10), ISLTL(10), ISCNT(10)
DATA IQM,IEM/1H?,11-
11 /
DO 111 LN=1,10
10 WRITE (LU,12)
12 FORMAT(/" *** Program THRESHOLD *** "/
1" ? : prints an explanation of the thresholding 
procedure;"
2" ! : will display your executed operations;"
3" G : will go back to the signal processing menu;"
4" <return> continues the program.")
WRITE (LU,14)
14 FORMAT("* ")
READ (LU,201) ISERV
IF (ISERV.EQ.IQM) CALL EXPL(LU)
IF (ISERV.EQ.IEM) THEN
   CALL DISPL(LN,ISTGC,ISUTL,ISLTL,ISCNT,LU)
   GOTO 10
ENDIF
IF (ISERV.EQ.1HG) RETURN

C->Initialization
ICNT = 0
ILTL = 0
IUTL = 255

C->Loser input
WRITE (LU,22)
FORMAT(/" Type your Upper Threshold Limit (UTL): _"/)
READ (LU,*,ERR=20) IUTL
WRITE (LU,26)
FORMAT(/" Type your Lower Threshold Limit (LTL): _"/)
READ (LU,*,ERR=24) ILTL
WRITE (LU,30)
FORMAT(/" To which graycolour value should your given"/
1" range be assigned? _"/)
READ (LU,*,ERR=28) ITGCV

C->LOSER INPUT CHECK
FLAG = 0
IF (IUTL.LT.0.OR.IUTL.GT.255) FLAG = 1
IF (ILTL.LT.0.OR.ILTL.GT.255) FLAG = 1
IF (ITGCV.LT.0.OR.ITGCV.GT.255) FLAG = 1
IF (FLAG.NE.1) GOTO 333
WRITE (LU,106)
FORMAT(//" Input out of range!! "/)
LN = LN-1
GOTO 10

IF (IUTL.EQ.0) IUTL = 255
ISTGC(LN) = ITGCV
ISUTL(LN) = IUTL
ISLTL(LN) = ILTL

C->Thresholding
DO 222 LM=1,LY
   DO 222 N1=1,LX
      IF (IARRY(N1,LM).LT.ILTL.OR.IARRY(N1,LM).GT.IUTL) GOTO 2
      IARRY(N1,LM) = ITGCV
      ICNT = ICNT+1
   CONTINUE
222        CONTINUE
   ISCNT(LN) = ICNT
111        CONTINUE
201        FORMAT(A1)
SUBROUTINE EXPL(LU)

C
RETURN
END

SUBROUTINE EXPL(LU)

C
--

WRITE (LU,101)

101 FORMAT(" THRH EXPLANATION !/
1" Program THRH works with the following algorithm "/
2" fz(x,y)= / 1 if f(x,y) is ELEMENT of z"/
3" 0 otherwise"/
4" This formula provides 3 different ways of operation:"/
5" 1. If a upperthreshold limit is set "/
6" and the lower thrh. limit is default "/
7" then all greycolour values between 0"/
8" and utl. are changed to your given"/
9" greycolour value."/
"&
1" 2. If the ltl is set and default on the"/
2" utl.; greycolour values between ltl."/
3" and 255 are changed."/
4" 3. If ltl. and utl. are set all values"/
5" in between are changed."/
6" combination of 1.,2. and 3. is called semi-"/
7" thresholding. 1. is used to extract objects"/
8" from background or noise. 2. removes sharp"/
9" contrasts. 3. enhances greycolour ranges(if"/
10" known exactly), which can be used for edge"/
"&" detection./")
RETURN
END

SUBROUTINE DISPL(L,ISTGC,ISULT,ISLTL,ISCNT,LU)

C
C...displays the range and assigned graycolour value
C of the THRH process including the number of changed
C pixels.
DIMENSION ISLTL(10),ISULT(10),ISTGC(10),ISCNT(10)

DO 111,I=1,(L-1)
WRITE (LU,101) I,ISLTL(I),ISULT(I),ISTGC(I),ISCNT(I)
101 FORMAT(I2", RANGE:"I3",""I3," CHANGED TO:"I3," # OF
CP:"I8)
111 CONTINUE
WRITE (LU,102)
102 FORMAT:///)
RETURN
END

SUBROUTINE HIST(IARRY)

C
C->Initialization

EMA IARRY
COMMON LU

RETURN
END
COMMON /INIT/LX,LY
COMMON /HISTO/ IHIST(0:255)
DIMENSION IARRY(256,256),IFT(130),ISAVE(17)
DATA ICHAR/1H+/

C->Reset the IHIST array

DO 555, I = 0, 255
555 IHIST(I) = 0
MAX = 0
IOFFS = 0

C->Default group for histogram display.

LM = 15
SUM = 0

C->Sort greyc. values

1 DO 111, LN=1, LY
DO 111, L=1, LX
INDX = IARRY(LN,L)
SUM = SUM + IARRY(LN,L)
111 IHIST(INDX) = IHIST(INDX)+1

C->Reset the first 17 ISAVE vars

DO 666, I=1,17
666 ISAVE(I) = 0

C->Print header

2 WRITE (LU,101)
101 FORMAT(/" # N I frequency"/
1"-- 0 -------------------------------"),
2"-----------------------------------")

C->Grouping and scaling

DO 222, J=1,17
DO 333, I=1,LM
JJ = (IOFFS+((J-1)*LM)+I)
ISAVE(J) = ISAVE(J)+IHIST(JJ)
333 CONTINUE
222 IF (ISAVE(J).GT.MAX) MAX = ISAVE(J)

C->Normalize the histogram frequency

DIV = 50./FLOAT(MAX)

C->Output

DO 444, I=1,17
II = IOFFS+(LM*I)
LN = INT(DIV*ISAVE(I))
C->No 0A1 format possible -) LN must be >0

IF (LN.LT.1) LN = 1
ENCODE (130,102,IFT) LN
WRITE (LU,IFT) II,ISAVE(I),(ICHAR,ICONT=1,LN)
FORMAT(('I4,I8," I","I3,'A1')
CONTINUE

C->The option (zoooooom)
WRITE(LU,1007) SUM
1007 FORMAT(/" IOD for the selected window ",I12)
WRITE (LU,105)
105 FORMAT(/" Would you like to see a more detailed histogra 
"")
READ (LU,201) IANS
201 FORMAT(A1)
IF (IANS.EQ.1HN) RETURN
WRITE (LU,107)
107 FORMAT(/" Type your new range (low,high): _")
READ (LU,*) NR1,NR2

C->Calculate (and correct) the new group size

LM = INT (IABS ( (NR2 -NR1) / 17) + 0.5)
IF ((17 * LM).LT.(MAX0 (NR1 , NR2))) LM = LM + 1

C->Reset the old ISAVE array

DO 777,I=1,17
777 ISAVE(I) = 0
IOFFS = NR1
IF (NR2.LT.NR1) IOFFS = NR2
GOTO 2
END
SUBROUTINE PCT(IARRY)

C-> Set IHIST array to zero

DO 2 I=0,255
2 IHIST(I) = 0

C-> Sort graycolors (perform histogram operation)

DO 20 I=1,LY
DO 20 J=1,LX
IDUM = IARRY(J,I)
20 IHIST(IDUM) = IHIST(IDUM) + 1
C-> Loser input

30 WRITE (LU,32)
32 FORMAT(/" Please input graycolor range, low to high ",")
READ (LU,*,ERR=30) ILOW,IHIGH
IF (ILOW.EQ.1000) RETURN
IF (ILOW.LT.0.OR.IHIGH.GT.255.OR.ILOW.GT.IHIGH) GOTO 30

C-> Compute size, sum pixels in range, compute percentage

ISIZE = LX * LY
ISUM = 0
DO 40 I=ILOW,IHIGH
40 ISUM = ISUM + IHIST(I)
PERCT = ( FLOAT (ISUM) / FLOAT (ISIZE) ) * 100.0

C-> Give user result

WRITE (LU,50) PERCT
50 FORMAT(/" This range of graycolors makes up "F5.2," percent"/
1" of the image")
RETURN
END

C SUBROUTINE EXPND(IARRY) ! EXPAND RANGE ROUTINE FOR SLAVE
C ! 8/10/84 AJS
EMA IARRY
COMMON LX,LY,MAX
DIMENSION IARRY(256,256)
IOLD = -1

C->Search for old maximum

DO 10 I=1,LY
DO 10 J=1,LX
IF (IARRY(J,I).GT.IOLD) IOLD = IARRY(J,I)
10 CONTINUE

C->Now we have old maximum and desired new maximum, C so we can compute a conversion factor.

RMULT = FLOAT (MAX) / FLOAT (IOLD)

C->Do it!

DO 20 I=1,LY
DO 20 J=1,LX
IARRY(J,I) = IARRY(J,I) * RMULT
20 CONTINUE

C->Done!
SUBROUTINE HSTEQ(IARRY)

C          -------- ----- ----- ----- 
EMA IARRY
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256),IR(256),S(256),IG(256)
INTEGER K,M
REAL XS,YS
C->Set IR and S to zero, get total number of points for divisi
5  WRITE(LU,15)
15  FORMAT(//'Please input the size of picture'/
1'How many columns and rows?_')
READ (LU,*,ERR=5) XS,YS
IF (XS.GT.256.0.OR.YS.GT.256.0) GOTO 5
DO 10 K=0,255
   IR(K) = 0.0
   S(K) = 0.0
10  CONTINUE
A=DBLE(XS*YS)
C->Get histogram (Note K+1 because R(1) <-> gray level 0)
DO 20 I=1,LY
   DO 20 J=1,LX
      K = IARRY(J,I)
      IR(K) = IR(K) + 1.0
20  CONTINUE
DO 70 K=0,255
   PRINT *, 'NUM',IR(K),A
70  CONTINUE
C->Get S’s (just a summation, or density function)
   S(0) = DBLE(IR(0) /A)
DO 30 K=1,255
   S(K)=DBLE(IR(K)/A)+DBLE(S(K-1))
30  CONTINUE
DO 1 K=0,255
   A=DBLE(XS*YS)
   PRINT*,'S(',K,')=',S(K),'IR(',K,')=',IR(K),A
1  CONTINUE
C->Now multiply by 256 & truncate to get 256 distinct levels
DO 40 M=0,255
   IG(M)=INT (S(M)*256.0)-1
40  CONTINUE
DO 60 M=0,255
   A=DBLE(XS*YS)
   PRINT *,'S(',M,')=' ,S(M),'IG(',M,')=' ,IG(M),A
60  CONTINUE
C-->Map new values into IARRY

DO 50 I=1,LY
DO 50 J=1,LX
M=IARRY(J,I)
IF(IARRY(J,I).EQ.M) IARRY(J,I) = INT (IG(M))
CONTINUE
RETURN
END

C ****************************
C ****** SUBROUTINE FILTR ******
C ****************************
C
C---
C ...applies a digital filter (F) to a time domain array.
C (smoothing by averaging)
C method: takes the average of the 8 surrounding (adjacent
C pixels and assigns it to the center.
C---
C Ext.: none
C call: CALL FILTR(IARRY,IPXL) & COMMON's
C---
C
C Inp. to outp chngs: IARRY
C
SUBROUTINE FILTR(IARRY,IPXL)

EMA IARRY,IPXL
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION F(3,3), IARRY(256,256), IPXL(256,256)
DATA F/.125,.125,.125,.125,0.,.125,.125,.125,.125/,
LYS = LY - 1
LXS = LX - 1
DO 111 I=2,LYS
DO 111 J=2,LXS
P=0.
DO 222 K=1,3
DO 222 L=1,3
P = P + F(K,L) * IARRY(J + K - 2, I + L - 2)
CONTINUE
NP = INT (P +.5)
IF (NP.LT.0) NP = 0
IF (NP.GT.255) NP = 255
IPXL(J,I) = NP
CONTINUE
DO 333 I=2,LYS
DO 333 J=2,LXS
IARRY(J,I) = IPXL(J,I)
CONTINUE
C ...applies the LAPLACIAN method to enhance the picture.
C (2nd. order gradient)
C I am using a laplacian divider of (1/5) to get a
C continuously weighted result rather than a black/
C white contrast.
C method: 0 1 0
C 1 -4 1
C 0 1 0
C the result of the multiplication matrix is divided by 5.
---
C ext. subs: none
C call proc: CALL LAPL(IARRY,IPXL) & COMMON's
---
C Inp. to outp chngs: IARRY
C
SUBROUTINE LAPL(IARRY,IPXL)

EMA IARRY,IPXL
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION F(3,3), IARRY(256,256), IPXL(256,256)
DATA F/0.,1.,0.,1.,-4.,1.,0.,1.,0./
LYS = LY - 1
LXS = LX - 1
DO 111 I=2,LYS
  DO 111 J=2,LXS
    P = 0.
    DO 222 K=1,3
      DO 222 L=1,3
        P = P + F(K,L) * IARRY(I + K - 2 ,J + L - 2)
    222 CONTINUE
    C Use divider!
    NP = INT (.5 + (P / 5.))
    IPXL(J,I) = NP
  111 CONTINUE
  DO 333 I=2,LYS
    DO 333 J=2,LXS
      IARRY(J,I) = IARRY(J,I) - IPXL(J,I)
      IF (IARRY(J,I).LT.0) IARRY(J,I) = 0
      IF (IARRY(I,J).GT.255) IARRY(I,J) = 255
  333 CONTINUE
RETURN
** SUBROUTINE HOLPL **
**********************************************************************
C           applies the LAPLACIAN method to enhance the picture.
C       (N-th. order gradient)
C      k1=1st neighbour field
C      k2=2nd neighbour field
C method:
C
  *******-K2  
  * ,K1 *  
  * *** *  
  * *X* *  
  * *** *  
  *   *  
  *******

C ext. subs.: none
C call proc.: CALL HOLPL(IARRY,IPXL)  & COMMON's
C
---

C SUBROUTINE HOLPL(IARRY,IPXL)
C
EMA IARRY,IPXL
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256) , IPXL(256,256)

C initialisation
  K1 = 3
  K2 = 7

C calculation of the laplacian
C inner boundary
  DO 111 K = 4 , LY - 3
  DO 111 L = 4 , LY - 3
  P=0.
  DO 222 I = 1 , K1
  DO 222 J = 1 , K1
  IF (J.EQ.1.OR.J.EQ.K1) GOTO 10
  IF (I.EQ.1.OR.I.EQ.K1) GOTO 10
  GOTO 222
10   P = P + IARRY (I + K - 2 , J + L - 2)
222   CONTINUE
  SK1 = P
C outer boundary
  P=0.
  DO 333 I=1,K2
  DO 333 J=1,K2

END
IF (J.EQ.1 .OR. J.EQ.K2) GOTO 20
IF (I.EQ.1 .OR. I.EQ.K2) GOTO 20
GOTO 333

20   P = P + IARRY(I + K - 4 , J + L - 4)

333  CONTINUE

SK2 = P

IPXL(K,L) = INT ((SK1/(K1*K1)) - ((SK2-SK1) / ((K2*K2) - (K1*K1))))

111  CONTINUE

C  normalisation to 0-255
DO 444 I=1,LY
DO 444 J=1,LX
IARRY(J,I) = IARRY(J,I) - IPXL(J,I)
IF (IARRY(J,I).LT.0) IARRY(J,I) = 0
IF (IARRY(J,I).GT.255) IARRY(J,I) = 255

444  CONTINUE

RETURN
END

C ****************************
C ** SUBROUTINE K I R S H **
C ****************************

C---

C ...applies the KIRSCH operator method for edge detection.
C (the K-op. method is a 1st order x-y difference equation)
C method: subtracts the sum of the left 3 pixels
C from the right 3 pixels, then subtracts the sum
C from the upper 3 pixels from the lower 3 pixels.
C finally it takes the maximum of this two subtractions.
C---

C ext.sub's: none
C call proc: CALL KIRSH(IARRY,IPXL)  & COMMON's
C---

C Inp. to outp. chngs: IARRY
C
C SUBROUTINE KIRSH(IARRY,IPXL)
C
C initialisation
EMA IARRY,IPXL
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256) , IPXL(256,256)
LYS = LY - 1
LXS = LX - 1

C calculation of the k-op.
DO 111 I = 2 , LYS
DO 111 J = 2 , LXS
IPXL(J,I) = 0
ILFT = IARRY(J-1,I-1) + IARRY(J-1,I) + IARRY(J-1,I+1)
IUP = IARRY(J-1,I-1) + IARRY(J-1,I) + IARRY(J+1,I-1)
IDWN = IARRY(J-1,I+1) + IARRY(J,I+1) + IARRY(J+1,I+1)

74
IRGT = IARRY(J+1,I-1) + IARRY(J+1,I) + IARRY(J+1,I+1)

\[
IPXL(J,I) = \max(\text{IABS}(\text{IRGT} - \text{ILFT}), \text{IABS}(\text{IUP} - \text{IDWN})) / 3
\]

CONTINUE

DO 333 I = 2, LYS
DO 333 J = 2, LXS
IARRY(J,I) = IPXL(J,I)
IF (IARRY(J,I) .LT. 0) IARRY(J,I) = 0
IF (IARRY(J,I) .GT. 255) IARRY(J,I) = 255
CONTINUE
CONTINUE
RETURN

END

SUBROUTINE NGHBR(IARRY, IPXL)

C
EMA IARRY, IPXL
C
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256), IPXL(256,256)

C->TILT!

DO 20 I = 1, LX
DO 20 J = 1, LY
IPXL(J,I) = IARRY(J,I)
20 CONTINUE

C->MAY THE FORCE BE WITH YOU!

DO 111 I = 3, LY-2, 2
DO 111 J = 3, LX-2, 2
IF (IARRY(J,I) .GT. IUTB OR IARRY(J,I) .LT. LTB) GOTO 111
DO 222 I1 = I-2, 1+2
DO 222 J1 = J-2, J+2
I2 = IABS(I1-I)
J2 = IABS(J1-J)
IF (I2.LT.2 .AND. J2.LT.2) GOTO 222
IF (IABS(IARRY(J1,I1) - IARRY(J,I)).GT.3) GOTO 222
I3 = INT(.5 * (I1 + I))
J3 = INT(.5 * (J1 + J))
IPXL(J3,I3) = INT(.5 * (IARRY(J,I) + IARRY(J1,I1)))
IF ((I2+J2).NE.3) GOTO 222
I3 = I3 + 1
J3 = J3 + 1
IPXL(J3,I3) = INT(.5 * (IARRY(J,I) + IARRY(J1,I1)))
222 CONTINUE
111 CONTINUE

C->SILLY RABBIT, TRIX ARE FOR KIDS!

DO 30 I = 1, LX
DO 30 J = 1, LY
IARRY(J,I) = IPXL(J,I)
30 CONTINUE
CONTINUE
RETURN
END

SUBROUTINE SVGCR(IARRY,IPXL,ITAG)

EMA IARRY,IPXL
COMMON LX,LY,IMAX,IMIN
DIMENSION IARRY(256,256),IPXL(256,256),ITAG(20)

SUBROUTINE SDFCR(IARRY,IPXL,ITAG)

EMA IARRY,IPXL
COMMON LX,LY,IMAX,IMIN
DIMENSION IARRY(256,256),IPXL(256,256),ITAG(20)

-->The first thing we have to do is store the first array

SUBROUTINE SHELL (IARRY,SPECT,ITAG)

COMPLEX SPECT
EMA IARRY,SPECT
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256),SPECT(256,256),ITAG(20)

-->Transform to frequency domain
CALL FWD2D (IARRY,SPECT,ITAG)

-->Choose filter
IF (ITAG(1).EQ.8) CALL BHPF(SPECT,ITAG)
IF (ITAG(1).EQ.9) CALL HMPHC(SPECT)
IF (ITAG(1).EQ.10) CALL POWER(SPECT)
IF (ITAG(1).EQ.11) CALL BLIND(SPECT)

-->Now transform back to time domain
CALL INV2D (IARRY,SPECT,ITAG)

-->Go back to image processing menu
RETURN
END

SUBROUTINE FWD2D (IARRY,SPECT,ITAG)

EMA IARRY,SPECT
COMPLEX SPECT,TEMP
COMMON /INIT/ LX,LY
COMMON /BUFRS/ TEMP(256)
DIMENSION IARRY(256,256),SPECT(256,256),ITAG(20)

-->Store one row at a time into buffers TEMPA and TEMPB, do on
dimensional transform, store in temporary arrays

DO 6 I=1,LY
DO 2 J=1,LX
  DMY = FLOAT (IARRY(J,I)) + 0.01 ! The .01 is so log (0) doesn't happen
  IF(ITAG(1).EQ.9) DMY = LOG (DMY) ! Homomorphic needs log first!
2  TEMP(J) = CMPLX (DMY,0.0)
  IF (ITAG(3).EQ.1) CALL FFT(TEMP,LX)
  IF (ITAG(3).EQ.2) CALL FWT(TEMP,LX)
  DO 4 J=1,LX
4  SPECT(J,I) = TEMP(J) * FLOAT(LX)
6  CONTINUE

C->Do transform on each column of transformed data

DO 12 J=1,LX
  DO 8 I=1,LY
8  TEMP(I) = SPECT(J,I)
  IF (ITAG(3).EQ.1) CALL FFT(TEMP,LY)
  IF (ITAG(3).EQ.2) CALL FWT(TEMP,LY)
  DO 10 I=1,LY
10  SPECT(J,I) = TEMP(I)
12  CONTINUE
RETURN
END

SUBROUTINE INV2D (IARRY,SPECT,ITAG)
C
EMA IARRY,SPECT
COMPLEX SPECT,TEMP
COMMON/INIT/ LX,LY
COMMON /BUFRS/ TEMP(256)
DIMENSION IARRY(256,256) , SPECT(256,256),ITAG(20)
C->Have to put each column into temporary buffers, take complex conjugate of data to make forward FFT into inverse

DO 6 J=1,LX
  DO 2 I=1,LY
2  TEMP(I) = CONJG (SPECT(J,I))
  IF (ITAG(3).EQ.1) CALL FFT(TEMP,LY)
  IF (ITAG(3).EQ.2) CALL FWT(TEMP,LY)
  DO 4 I=1,LY
4  SPECT(J,I) = TEMP(I)
6  CONTINUE

C->Now do inverse transform on each row; Since we only had a real C function to start with, take the real part and mult. by LX.

DO 12 I=1,LY
  DO 8 J=1,LX
8  TEMP(J) = SPECT(J,I)
  IF (ITAG(3).EQ.1) CALL FFT(TEMP,LX)
  IF (ITAG(3).EQ.2) CALL FWT(TEMP,LX)
  DO 10 J=1,LX
10  SPECT(J,I) = TEMP(J)
SUBROUTINE BHPF(SPECT, ITAG)

! Butterworth high-pass filter

EMA SPECT
COMPLEX SPECT, TEMP
COMMON LU
COMMON /INIT/ LX, LY
COMMON /BUFRS/ TEMP(256)
DIMENSION SPECT(256, 256), ITAG(20)
REAL * 8 C

C-> Set Do to 1/8 of distance
5 WRITE (LU, 12)
12 FORMAT('Please input the length of x size:_')
READ (LU, *, ERR=5) XS
IF (XS.GT.256) GOTO 5
DZERO = DBLE (XS) * SQRT (2.0) / 8.0

C-> Set n to 3

N=3

C-> Compute D(u,v), then H

DO 10 I=1, LX
DO 10 J=1, LY
  C = DBLE (REAL (I ** 2)) + DBLE (REAL (J ** 2))
  D = DBLE (DSQRT (C))
  BLOB = (( (DZERO/D) ** (2*N)) * 0.414)
  H = ( (2+BLOB) / (1+BLOB) )
10  SPECT(J, I) = SPECT(J, I) * H
RETURN
END

SUBROUTINE HMPHC(SPECT)

EMA SPECT
COMPLEX SPECT, TEMP
COMMON /INIT/ LX, LY
COMMON /BUFRS/ TEMP(256)
DIMENSION SPECT(256, 256)
REAL * 8 C

C

DO 10 I=1, LX
DO 10 J=1, LY
  C = DBLE (FLOAT (I ** 2)) + DBLE (FLOAT (J ** 2))
  D = SNGL (DSQRT (C))
H = (2.0 * EXP (-.1 * FLOAT(LX) / D))  ! This set upper to 2

SPECT(J,I) = SPECT(J,I) * H

RETURN

SUBROUTINE FFT(F,N)

C

DO 1 LN=1,9

! 512 Largest image

1 CONTINUE

STOP 0007

2 NV2=N/2

3 J=1

DO 5 I=1,NM1

4 IF (I.GE.J) GOTO 3

5 J=J+K

DO 7 L=1,LN

6 F(I)=F(I)/FLOAT(N)

7 RETURN

END

SUBROUTINE FWT(CF,N)

C

DO 10 LN=1,9

10 CONTINUE

STOP 0010
DO 30 I=1,128
F(I)=REAL(CF(I))
NV2=N/2
NM1=N-1
J=1
DO 3 I=1,NM1
   IF (I.GE.J) GOTO 1
   T=F(J)
   F(J)=F(I)
   F(I)=T
1   K=NV2
2   IF (K.GE.J) GOTO 3
   J=J-K
   K=K/2
   GOTO 2
3   J=J+K
DO 5 L=1,LN
   LE=2**L
   LE1=LE/2
   DO 5 J=1,LE1
      DO 4 I=J,N,LE
         IP=I+LE1
         T=F(IP)
         F(IP)=F(I)-T
5      CONTINUE
4      F(I)=F(I)+T
5
DO 6 I=1,N
6   CF(I)=CMPLX((F(I)/FLOAT(N)),0.0)
RETURN
END

SUBROUTINE SMOOTH(IARRY,IPXL)
C***********************************************************************
C This subprogram is to define the size of leukocyte
C It is the powerful tool to eliminiate noise which
C small than the defined size
C***********************************************************************
EMA IARRY
COMMON LU
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256)
INTEGER COUNT,OUTGREY,NEB
OUTGREY=255
NEB=0
C This part is to make bigrey level
21 WRITE (LU, 22)
22 FORMAT(//'Please input the grey level you want:_'')
READ (LU,*,ERR=21) OUTGREY
C To define the size of noise
30 WRITE(LU,32)
32 FORMAT(//'Please input the neberhor of pls:_')
READ (LU,*,ERR=30) NEB
IF (OUTGREY .GT. 255 .OR. OUTGREY .LT.0) GOTO 21
IF (NEB .GT. 8 .OR. NEB .LT. 0) GOTO 30
C This portion can cancel the spot which smaller than some
C certain values
DO 10 J=1,LX
DO 10 I=1,LY
IF (IARRY(J,I).NE.0)THEN
  COUNT=0
  IF (IARRY(J-1,I).NE.0) COUNT=COUNT+1
  IF (IARRY(J,I-1).NE.0) COUNT=COUNT+1
  IF (IARRY(J+1,I-1).NE.0) COUNT=COUNT+1
  IF (IARRY(J+1,I).NE.0) COUNT=COUNT+1
  IF (IARRY(J-1,I-1).NE.0) COUNT=COUNT+1
  IF (IARRY(J+1,I+1).NE.0) COUNT=COUNT+1
  IF (IARRY(J-1,I+1).NE.0) COUNT=COUNT+1
  IF (IARRY(J,I+1).NE.0) COUNT=COUNT+1
  IF (COUNT .LT. NEB) THEN
    IARRY(J,I)=0
  ELSE
    C Give spot a certain grey level value
    IARRY(J,I)=OUTGREY
    IF (IARRY(J-1,I).NE.0) IARRY(J-1,I)=OUTGREY
    IF (IARRY(J,I-1).NE.0) IARRY(J,I-1)=OUTGREY
    IF (IARRY(J+1,I-1).NE.0) IARRY(J+1,I-1)=OUTGREY
    IF (IARRY(J-1,I-1).NE.0) IARRY(J-1,I-1)=OUTGREY
    IF (IARRY(J,I+1).NE.0) IARRY(J,I+1)=OUTGREY
    IF (IARRY(J+1,I+1).NE.0) IARRY(J+1,I+1)=OUTGREY
    IF (IARRAY(J-1,I+1).NE.0) IARRAY(J-1,I+1)=OUTGREY
    IF (IARRY(J+1,I).NE.0) IARRY(J+1,I)=OUTGREY
  END IF
END IF
10 CONTINUE
RETURN
END

SUBROUTINE COUNT(IARRY)
C******************************************************************************
C The subroutine is to use the property of connectivity
C to define an individual spot. Within the program existing
C a TRACK subprogram to calculate and search the connective
C pixel with the same grey level. After searching, the TRACK
C will erase the intact spot, and go back to COUNT to tell
C that it has already find one spot and erase it, then
C COUNT will add the number of spot
C******************************************************************************
EMA IARRY
COMMON/INIT/ LX,LY
DIMENSION IARRY(256,256)
INTEGER ICOUNT
ICOUNT=0
DO 10 I=1,LY
DO 10 J=1,LX
IF (IARRY(J,I).NE.0) THEN
  C Now the route can trace the connective pixel to define
  C a complete spot
  EMA IARRY
CALL TRACK(IARRY,J,I)
ICOUNT=ICOUNT+1
END IF
10 CONTINUE
PRINT *, 'THE NUMBER OF COUNTER: ', ICOUNT
RETURN
END

SUBROUTINE TRACK(IARRY,J,I)
C*************************************************************
C The purpose of the subprogram is to find the adjectant pixel
C which exist a certain number of grey level, and determine the
C relation of connectivity to define an intact leukocyte
C*************************************************************
EMA IARRY
COMMON /INIT/ LX,LY
DIMENSION IARRY(256,256)
INTEGER PTR,STACK(10000),ADDR

C Using a simulating method to do the recursive program
PTR=1
10 IARRY(J,I)=0
IF (IARRY(J-1,I).NE.0) THEN
STACK(PTR)=J
STACK(PTR+1)=I
STACK(PTR+2)=1
PTR=PTR+3
J=J-1
GOTO 10
END IF
1 IF (IARRY(J-1,I+1).NE.0) THEN
STACK(PTR)=J
STACK(PTR+1)=I
STACK(PTR+2)=2
PTR=PTR+3
J=J-1
I=I+1
GOTO 10
END IF
2 IF (IARRY(J,I+1).NE.0) THEN
STACK(PTR)=J
STACK(PTR+1)=I
STACK(PTR+2)=3
PTR=PTR+3
I=I+1
GOTO 10
END IF
3 IF (IARRY(J+1,I+1).NE.0) THEN
STACK(PTR)=J
STACK(PTR+1)=I
STACK(PTR+2)=4
PTR=PTR+3
J=J+1
I=I+1
GOTO 10
END IF
4  IF (IARRY(J+1,I).NE.0) THEN
    STACK(PTR)=J
    STACK(PTR+1)=I
    STACK(PTR+2)=5
    PTR=PTR+3
    J=J+1
    GOTO 10
END IF
5  IF (IARRY(J+1,I-1).NE.0) THEN
    STACK(PTR)=J
    STACK(PTR+1)=I
    STACK(PTR+2)=6
    PTR=PTR+3
    J=J+1
    I=I-1
    GOTO 10
END IF
6  IF (IARRY(J,I-1).NE.0) THEN
    STACK(PTR)=J
    STACK(PTR+1)=I
    STACK(PTR+2)=7
    PTR=PTR+3
    I=I-1
    GOTO 10
END IF
7  IF (IARRY(J-1,I-1).NE.0) THEN
    STACK(PTR)=J
    STACK(PTR+1)=I
    STACK(PTR+2)=8
    PTR=PTR+3
    J=J-1
    I=I-1
    GOTO 10
END IF
8  IF (PTR.EQ.1) RETURN
    ADDR=STACK(PTR-1)
    I=STACK(PTR-2)
    J=STACK(PTR-3)
    PTR=PTR-3
    IF (ADDR.EQ.1) GOTO 1
    IF (ADDR.EQ.2) GOTO 2
    IF (ADDR.EQ.3) GOTO 3
    IF (ADDR.EQ.4) GOTO 4
    IF (ADDR.EQ.5) GOTO 5
    IF (ADDR.EQ.6) GOTO 6
    IF (ADDR.EQ.7) GOTO 7
    GOTO 8
END

SUBROUTINE ERSON(IARRY,IPXL)
C Use mathematical morphology to erase the noise, meanwhile it will erose the outer layer of leukocyte

```
EMA IARRY, IPXL
COMMON LU
COMMON /INIT/ LX, LY
DIMENSION IARRY(256, 256), IPXL(256, 256), ST(10, 10)
INTEGER SX, SY, SX2, SY2
C SAM is the gray level of struture element
SAM = 80
C Now set the size of strutrue element and assign its' gr level
SX = 2
SY = 2
SX2 = (SX - 1) / 2
SY2 = (SY - 1) / 2
DO 10 N = 1, SY
    DO 10 M = 1, SX
        ST(M, N) = SAM
10 CONTINUE
DO 20 J = 1 + SX2, LX - (SX - SX2) + 1
    DO 20 I = 1 + SY2, LY - (SY - SY2) + 1
        VALUE = IARRY(J + M - SX2 - 1, I + N - SY2 - 1) - ST(M, N)
        SAM = MIN(SAM, VALUE)
        IF (SAM .LT. 0) SAM = 0
30 CONTINUE
    IPXL(J, I) = SAM
20 CONTINUE
DO 50 J = 1 + SX2, LX - (SX - SX2) + 1
    DO 50 I = 1 + SY2, LY - (SY - SY2) + 1
        IARRY(J, I) = IPXL(J, I)
50 CONTINUE
RETURN
END
```

SUBROUTINE DATON(IARRY, IPXL)
C The Dialation will add an outer layer to the leukocyte
C it is the power tool to fill the hollow hole within the leukocyte

```
EMA IARRY, IPXL
COMMON LU
COMMON /INIT/ LX, LY
DIMENSION IARRY(256, 256), IPXL(256, 256), ST(10, 10)
INTEGER SX, SY, SX2, SY2
C Set the struture element gray level
```
LAG=0
C The size of struture element
SX=2
SY=2
SX=(SX-1)/2
SY=(SY-1)/2
DO 10 M=1,SX
DO 10 N=1,SY
ST(M,N)=LAG
10 CONTINUE
DO 20 J=1+SX2,LX-(SX-SX2)+1
DO 20 I=1+SY2,LY-(SY-SY2)+1
LAG=0
DO 30 M=1,SX
DO 30 N=1,SY
C The formula of dialtion
VALUE=IARRY(J+M-SX2-1,I+N-SY2-1)+ST(M,N)
IF (LAG.LT.VALUE) LAG=VALUE
IF (LAG.GT.255) LAG=255
30 CONTINUE
IPXL(J,I)=LAG
20 CONTINUE
DO 50 J=1,LX
DO 50 I=1,LY
IARRY(J,I)=IPXL(J,I)
50 CONTINUE
RETURN
END
B : References

References


