

205

IMAGE ENHANCEMENT TECHNIQUES FOR DEGRADED FINGERPRINTS

by

RONALD ALLEN GIFFORD

B.S., Kansas State University, 1986

A THESIS

submitted in partial fulfillment of the

requirements for the degree


MASTER OF SCIENCE

ELECTRICAL ENGINEERING

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1988

Approved by:


Major Professor

LD
2668
.74
EECE
1988
G53
C. 2

TABLE OF CONTENTS

A11208 129663

| CHAPTERS | | PAGE |
|------------|---|------|
| 1.0 | Intoduction | 1 |
| 2.0 | Fingerprint Identification and Recovery | 3 |
| 3.0 | Enhancement Techniques | |
| 3.1 | Image Definitions | 9 |
| 3.2 | Spatial-domain Methods | 10 |
| 3.3 | Frequency-domain Methods | 18 |
| 4.0 | Project Description | |
| 4.1 | Image Processing System | 20 |
| 4.2 | Computer Programs | 21 |
| 5.0 | Results | |
| 5.1 | Introduction | 23 |
| 5.2 | Procedure One | 26 |
| 5.3 | Procedure Two | 37 |
| 6.0 | Summary and Conclusions | 70 |
| 7.0 | References | 73 |
| APPENDIXES | | |
| A | Equipment list | 74 |
| B | Computer Listings | 75 |

LIST OF FIGURES

| FIGURE | PAGE |
|--|------|
| 2-1 Example of a whorl with bifurcation and ridge ending | 5 |
| 3.2-1 Gray-level transformation function | 11 |
| 3.2-2 3 x 3 spatial mask | 14 |
| 3.2-3 Spatial mask used for neighborhood averaging | 15 |
| 3.2-4 Geometric interpretation of density slicing | 17 |
| 5.1-1 Sample recovered after two hours | 41 |
| 5.1-2 Histogram of image in Figure 5.1-1 | 42 |
| 5.2-1 Linear contrast stretching function | 27 |
| 5.2-2 Result of linear contrast stretching applied to image in Figure 5.1-1 | 43 |
| 5.2-3 Histogram of image in Figure 5.2-2 | 44 |
| 5.2-4 Result of transforming image in Figure 5.2-2 to the frequency domain and displaying using Equation (5.2-3) | 45 |
| 5.2-5 Histogram of filtered image before normalization | 46 |
| 5.2-6 Result of highpass filtering and normalization applied to image in Figure 5.2-2 | 47 |
| 5.2-7 Histogram of image in Figure 5.2-6 | 48 |
| 5.2-8 Parabolic contrast stretching function | 31 |
| 5.2-9 Result of parabolic contrast stretching applied to image in Figure 5.2-6. | 49 |
| 5.2-10 Histogram of image in Figure 5.2-9 | 50 |
| 5.2-11 Result of histogram equalization applied to image in Figure 5.2-9 | 51 |

| FIGURE | PAGE |
|--|------|
| 5.2-12 Histogram of image in Figure 5.2-11 | 52 |
| 5.2-13 Result of neighborhood averaging applied to image in Figure 5.2-11 | 53 |
| 5.2-14 Histogram of image in Figure 5.2-13 | 54 |
| 5.2-15 Result of applying 3 x 3 median filter to image in Figure 5.2-11 | 55 |
| 5.2-16 Histogram of image in Figure 5.2-15 | 56 |
| 5.2-17 Spatial masks used to compute horizontal and vertical gradients | 33 |
| 5.2-18 Spatial mask used to compute Laplacian gradient | 34 |
| 5.2-19 Histogram of gradient for image in 5.2-13 | 57 |
| 5.2-20 Gradient of image in Figure 5.2-13 after thresholding | 58 |
| 5.2-21 Result of edge enhancement through gradient thresholding applied to image in Figure 5.2-13 | 59 |
| 5.2-22 Block diagram for Procedure One | 36 |
| 5.3-1 Result of smoothing through neighborhood averaging applied to image in Figure 5.1-1 | 60 |
| 5.3-2 Histogram of image in Figure 5.3-1 | 61 |
| 5.3-3 Result of 3 x 3 median filtering applied to image in Figure 5.1-1 | 62 |
| 5.3-4 Histogram of image in Figure 5.3-3 | 63 |
| 5.3-5 Result of transforming image in Figure 5.3-1 to the frequency domain and displaying using Equation (5.2-3) | 64 |
| 5.3-6 Histogram of filtered image before normalization | 65 |
| 5.3-7 Result of highpass filtering and normalization applied to image in Figure 5.3-1 | 66 |

| FIGURE | PAGE |
|---|------|
| 5.3-8 Histogram of image in Figure 5.3-7 | 67 |
| 5.3-9 Result of histogram equalization applied to image 5.3-7 | 68 |
| 5.3-10 Histogram of image in Figure 5.3-9 | 69 |
| 5.3-11 Block diagram for Procedure Two | 40 |

CHAPTER 1: INTRODUCTION

Fingerprint identification is one of the principal methods used by criminal investigators to obtain evidence. The uniqueness of individual fingerprints allows for positive identification of prints left at the scene of a crime. Prints cannot be altered; thus, they provide a permanent form of identification.

Fingerprints can be obtained from different types of surfaces. One type of surface which presents difficulty in obtaining prints is human skin. The difficulty arises from physiological processes which occur in the body that result in rapid deterioration of prints on skin. Therefore, prints on human skin must be recovered quickly before they become unrecognizable. In a situation where a victim has been traumatized, such as rape, a period of time usually passes before the crime is reported. As the time following placement of the print lengthens, it becomes increasingly difficult for investigators to obtain distinguishable prints.

The primary purpose of this project is to increase the recovery time for prints obtained from human skin through digital image processing methods. Initially, pictures of prints are recorded as digital images. The image is then processed using image enhancement techniques. Through these

techniques, it is possible to enhance the ridge structure of degraded fingerprint images in order to produce a recognizable print.

There is a wide variety of image enhancement techniques available for improving image quality. The features to be enhanced determine which type of techniques are best suited for the task. Most routines attempt to accentuate areas of differing contrast. For instance, highlighting edges between contrasting regions is one method often used. Many times a combination of several techniques will produce the best results. Examination of the statistics for an image can often provide insight as to the effectiveness of the enhancement methods.

CHAPTER 2: FINGERPRINT IDENTIFICATION AND RECOVERY

The term dermatoglyphics was coined around 1926 to describe the science of the study of skin patterns [3]. It particularly applies to the patterns of the specialized skin of the inferior surfaces of the hands and feet, those being the soles of the feet and the palms and fingertips of the hand. Dermatoglyphics is concerned primarily with the morphological and physiological aspects of these skin patterns.

The skin of the inferior surfaces of the hands and feet is unlike that of any other part of the body. These areas, which contain neither hair nor sebaceous (oil) glands, possess an abundance of relatively large sweat glands. Perhaps the most important characteristic of the skin in these areas is the continuously corrugated surface with narrow ridges.

One important practical application of dermatoglyphics is in the area of anthropometric identification, specifically, fingerprint identification. A fingerprint is formally defined as a reproduction of the pattern or design formed by the ridges of the first joint of a finger or thumb. To date, no two fingerprints have been found to be exactly alike. Fingertip patterns are formed during the early embryonic stage and never change during the entire course of a human life. Even injury does not change the patterns

since the ridges are known to heal exactly as before to within a millionth of an inch. Many torturous attempts to alter ridge patterns have failed due to the natural healing process.

The pattern arrangements of fingertip ridges have been noticed by man for centuries. It is known that the Chinese used fingerprints to sign documents dating back to the third century B.C. A Chinese contract of loan executed nearly 1200 years ago contains the imprint of a fingertip in a pat of clay which is attached to the document. An inscription on the document states, "The two parties have found this just and clear and have affixed impressions of their fingers to serve as a mark."

Scientific descriptions of the patternings of fingers and palms date back to the late seventeenth century. The first item in modern literature relating to fingerprint identification was published by Henry Faulds in 1880 in an English journal called Nature. In this short article Faulds points out that chance fingerprints left at the scene of a crime would provide positive identification of the offender when apprehended.

The modern system of fingerprint identification originated in the late nineteenth century from work done by Faulds, Sir Francis Galton, Sir Edward Richard Henry, and many others.

The current system is called the Henry system after Sir Henry. In the Henry system, fingerprint characteristics are divided into three general patterns called loops, arches, and whorls [2]. Prints are categorized by the relative locations of specific patterns. The patterns are described by the number of ridges which comprise them. An example of a whorl is shown in Figure 2-1.

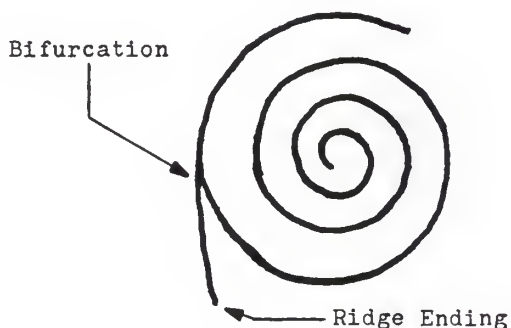


Figure 2-1 Example of a whorl with bifurcation and ridge ending

The most important features of fingerprints are bifurcations and ridge endings [8]. Bifurcations are locations in a fingerprint where ridges branch out in different directions. Ridge endings are simply locations where a specific ridge terminates. Identification of a print is accomplished by pinpointing the exact locations of points corresponding to bifurcations and ridge endings. Each state has its own laws governing the exact number of points required to make a positive identification. For example, the State of Kansas

requires nine such points whereas California requires sixteen points to make a positive identification of a print. Defects such as permanent scars are also used in fingerprint identification since they render a print unique.

The most difficult task confronting criminal investigators is the recovery of fingerprints from the scene of a crime. Fingerprints left at the scene of a crime are basically of three types: visible, plastic, and latent prints. Visible prints are transferred from smearings of the hand with ink, paint, blood, or other substances which come into contact with a surface of contrasting color. Plastic prints are formed from ridge impressions in pliable materials such as soap, putty, wax, and dust. The most difficult type of prints to recover are latent prints. These are ridge impressions which are formed on surfaces by perspiration or oil from the fingers. Latent prints are scarcely visible so development of the print is necessary for further study.

Many methods are used to recover latent fingerprints. The most common method involves dusting a surface with a powder of contrasting color. This method works well on hard, non-porous surfaces. Another popular method is the treatment of a surface, such as paper, with a silver nitrate solution. There are many other esoteric methods such as interferometric and infrared microscopy, laser illumination, metal

evaporation/deposition, and x-ray electronography.

Human skin is one surface from which it is extremely difficult to recover latent prints. A print left on the surface of human skin will deteriorate after a period of time due to perspiration and oil secretion from beneath the skin. Thus, the recovery time for a latent print left on human skin is limited. The actual recovery time is dependent upon the persons involved because oil and skin compositions differ. Good prints have been recovered from live persons within a period of ninety minutes and from cadavers up to forty-one hours after placement.

The recovery of latent prints from human skin is a great asset to criminal investigators involved in cases such as strangulation and rape. The most widely used method is the iodine-silver plate method. It has been adopted by the F.B.I. as well as many State and Federal agencies. The prints for this project were obtained using a modification of this method [5].

The modified iodine-silver plate method begins with the preparation of the recovery medium which is a two-inch piece of 35 mm camera film. This is immersed in a fixer solution to strip away the silver salts so that only transparent acetate remains. The clear acetate is dipped in a solution of leuco crystal violet in xylene. Leuco crystal violet is a

reagent used to detect iodine.

Once the treated acetate is dry, the skin area is prepared for the print transfer. The suspected area is fumed with iodine vapor. Iodine vapor is known to be absorbed by the body oils that generally comprise the ridges of latent prints. Continued fuming of the area reveals the ridge detail of the print. When the print is visible, the treated acetate is pressed over the print with uniform pressure until transferred. A thin coating of rubber cement is then brushed over the acetate to preserve the print.

CHAPTER 3: ENHANCEMENT TECHNIQUES

3.1 IMAGE DEFINITIONS

Image enhancement routines process digital images in order to refine certain features. A digital image may be represented as a two-dimensional light-intensity function denoted as $f(x,y)$ where x and y are independent variables denoting spatial coordinates. Specific coordinates are referred to as picture elements or pixels. The value of f describes the brightness of the image at the coordinates (x,y) . This brightness intensity is divided into discrete values called gray levels. A digital image may be considered as a matrix. Each element of the matrix represents the gray level value of the pixel at that location. The location of a pixel in the matrix is identified by its spatial coordinates.

Enhancement operations may be represented as mathematical transformations. In general, the digital image $f(x,y)$ is transformed to an image $g(x,y)$ according to a specific mathematical rule. A variety of rules may be derived to enhance specific features contained in an image.

Enhancement techniques are generally classified as either spatial-domain or frequency-domain methods [6]. Spatial-domain methods directly manipulate the pixels of an image. Frequency-domain methods modify spatial frequency informa-

tion to enhance corresponding features in the spatial domain. The Fourier transform is used to transform the original image to the frequency domain.

3.2 SPATIAL-DOMAIN METHODS

Spatial-domain methods work on a variety of principles. The most elementary operation is known as thresholding. In this method, a cutoff value called a threshold is specified. Gray levels either above or below this threshold are changed to a specified value by the transformations

$$g(x,y) = \begin{cases} f(x,y) & \text{if } f(x,y) \geq t \\ c & \text{if } f(x,y) < t \end{cases}$$

or

$$g(x,y) = \begin{cases} f(x,y) & \text{if } f(x,y) \leq t \\ c & \text{if } f(x,y) > t \end{cases}$$

(3.2-1)

where c is a specified gray level and t is the threshold value. This method is effective when a large degree of contrast is evident in the original image.

Thresholding is actually a specific case of a technique known as contrast stretching. This is a technique where gray levels from $f(x,y)$ are mapped to $g(x,y)$ according to a

specific transformation function. Let r and s represent gray level values in $f(x,y)$ and $g(x,y)$, respectively. The transformation function is given by $s = T(r)$ where T is a mathematical operator defined on r . A transformation function is constrained to be a continuous, monotonically increasing function whose range is equivalent to its domain. Examples of transformation functions are shown graphically in Figure 3.2-1.

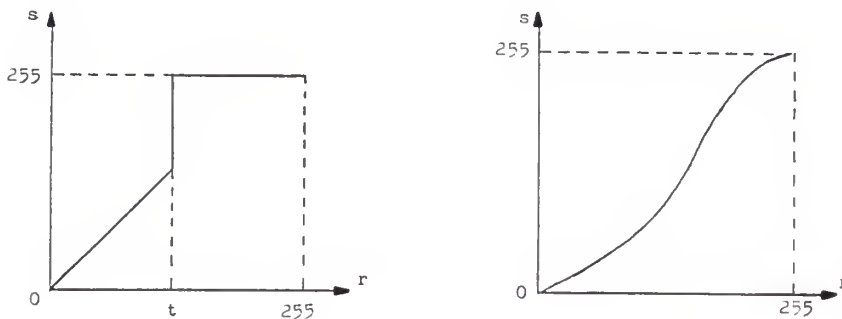


Figure 3.2-1 Gray-level transformation functions

Another class of spatial-domain methods are known as histogram modification techniques [6]. A histogram is a plot which provides a description of the distribution of gray levels in an image. Gray-level distributions are described mathematically using a probability density function. The probability density function is defined according to the relative frequency of gray levels appearing in an image expressed as

$$p_r(r) = t / N \quad (3.2-2)$$

where t is the number of pixels with gray level r and N is the total number of pixels in $f(x,y)$.

Histogram modification techniques map values by specifying the probability density function for the enhanced image.

Let $p_s(s)$ denote a specific probability density function for the enhanced image $g(x,y)$ and $s = T(r)$ denote the transformation function. Histogram modification techniques are based on the relationship

$$p_s(s) = [p_r(r)dr/ds] \Big|_{r=T^{-1}(s)} \quad (3.2-3)$$

where $T^{-1}(s)$ is the inverse transformation function.

The most common histogram modification technique is known as histogram equalization. Enhancement is accomplished through equalization using the transformation function

$$s = T(r) = \int_0^r p_r(w)dw \quad (3.2-4)$$

where w is a dummy variable. This transformation attempts to distribute the gray levels from the density function $p_r(r)$ as a uniform density function, denoted as $p_s(s)$, in order to effect an increase in the dynamic range of the pixels in the image.

A useful variety of enhancement operations are image sharpening techniques [6]. These are employed primarily as a means for highlighting edges in an image. In the spatial domain, differentiation is the operation used to emphasize edge detail. An edge is a boundary between contrasting regions characterized by an abrupt change in the gray levels of adjacent pixels.

Differentiation is usually executed using the gradient operator. The gradient of $f(x,y)$ is a vector defined as

$$G[f(x,y)] = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (3.2-5)$$

The gradient is a directional derivative which points in the direction of the maximum rate of increase of $f(x,y)$.

Gradients are implemented through the use of spatial masks. A mask is a small two-dimensional array with coefficients which are chosen to recognize a certain property in an image. A spatial mask is a local operator in that it is only concerned with a single point and its surrounding neighborhood. A 3 x 3 spatial mask with corresponding pixel locations is shown in Figure 3.2-2.

| | | |
|------------------------|----------------------|------------------------|
| w_1 $f(x-1, y-1)$ | w_2 $f(x-1, y)$ | w_3 $f(x-1, y+1)$ |
| w_4 $f(x, y-1)$ | w_5 $f(x, y)$ | w_6 $f(x, y+1)$ |
| w_7 $f(x+1, y-1)$ | w_8 $f(x+1, y)$ | w_9 $f(x+1, y+1)$ |

Figure 3.2-2 3 x 3 spatial mask

Consider the 3 x 3 mask and image neighborhood as row-ordered column vectors w and f , respectively. These vectors are written as

$$f = \begin{bmatrix} f(x-1, y-1) \\ \vdots \\ f(x-1, y+1) \\ \vdots \\ f(x+1, y-1) \\ \vdots \\ f(x+1, y+1) \end{bmatrix} \quad w = \begin{bmatrix} w_1 \\ \vdots \\ w_3 \\ \vdots \\ w_7 \\ \vdots \\ w_9 \end{bmatrix} \quad (3.2-6)$$

The computation of the mask operation is executed as the inner product of the two vectors:

$$w'f = w_1 f(x-1, y-1) + w_2 f(x-1, y) + \dots + w_9 f(x+1, y+1) \quad (3.2-7)$$

The value $f(x, y)$ is replaced by the magnitude of the inner

product to produce the gradient.

If appropriate masks are used, the magnitude of the gradient reveals points whose neighbors have greatly differing gray levels. These points correspond to singularities in $f(x,y)$. Edges are represented as a connected boundary of singularities. Many types of masks have been derived which locate edges oriented in various directions [9].

Image smoothing operations are used to remove noise which may be present in an image. One such method is known as neighborhood averaging. This is performed by replacing a pixel with the average of the surrounding neighborhood using a spatial mask. A mask used for averaging over a 3 x 3 neighborhood is shown in Figure 3.2-3.

| | | |
|-----|-----|-----|
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |

Figure 3.2-3 Spatial mask used for neighborhood averaging

Median filtering is another smoothing technique. A median filter replaces a pixel with the median of the surrounding neighborhood. The advantage of this method over neighborhood averaging is a reduction in blurring which preserves edge content in an image [7].

Pseudo-color processing is a relatively new class of spatial-domain methods. Many operations are possible within this powerful area of enhancement techniques. Pseudo-color techniques are based on the principle that the human eye has the ability to discern a wide range of colors and intensities. This ability is limited when viewing monochrome images.

The simplest operation in pseudo-color techniques is called density slicing. Density slicing is useful in displaying the spatial locations of specific gray levels. The basic idea behind this process is to assign a color to each pixel based on its intensity. This is analogous to slicing the two-dimensional image function $f(x,y)$ with a plane and assigning a color to that plane. A geometric interpretation of this process is shown in Figure 3.2-4.

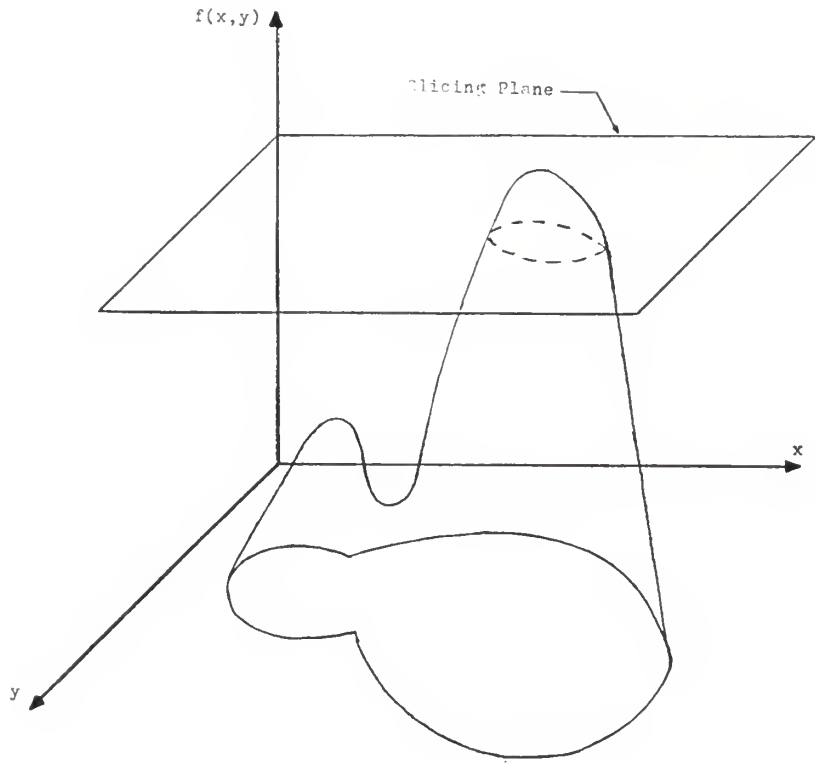


Figure 3.2-4 Geometrical interpretation of density slicing

3.3 FREQUENCY-DOMAIN METHODS

Frequency-domain methods enhance images by modifying spatial frequencies which correspond to particular features in the spatial domain. Generally, images are transformed from the spatial domain to the frequency domain via the discrete Fourier transform (DFT).

For an $M \times N$ image, the two-dimensional DFT is computed as

$$F(u,v) = (1/MN) \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp[-j2\pi(ux/M + vy/N)] \quad (3.3-1)$$

for the spatial frequencies $u = 0, \dots, M-1$ and $v = 0, \dots, N-1$. The DFT is a separable transform, so the two-dimensional DFT can be computed using a one-dimensional FFT algorithm [6]. This reduces the number of operations, which in turn provides a significant increase in computing speed [1].

Filtering is the basic operation used in frequency-domain methods. Frequency information is characterized by its distance from the origin of the frequency plane. High spatial frequencies are located furthest away from the origin while low frequencies are closest to the origin. Filtering modifies spatial frequencies according to their distance from the origin in the frequency plane. The extent to which the frequency values are modified is defined by a transfer func-

tion. Values of this function are restricted to the interval $[0,1]$. A cutoff point corresponding to a radius about the origin determines the rate of transition from the minimum to the maximum values of the transfer function. The value of the transfer function at the cutoff point is generally equal to half of its maximum value.

Let $G(u,v)$ denote the function for the enhanced image in the frequency domain. Consider a transfer function in the frequency domain given by $H(u,v)$. The filtering operation is computed as

$$G(u,v) = F(u,v)H(u,v) \quad (3.3-2)$$

This operation corresponds to convolution in the spatial domain. Transforming the filtered image back to the spatial domain yields the enhanced image $g(x,y)$.

Image sharpening is accomplished in the frequency domain through the implementation of highpass filters. High-frequency components in the frequency domain correspond to edge content in the spatial domain. Edges are accentuated by attenuating the low-frequency components using a highpass transfer function. High-frequency components may be further emphasized by adding a constant to the highpass transfer function. That is,

$$\hat{H} = H + c \quad (3.3-3)$$

where c is a constant.

CHAPTER 4: PROJECT DESCRIPTION

4.1 IMAGE PROCESSING SYSTEM

The objective of this project is to enhance degraded fingerprint images using digital processing methods. This requires that pictures of the prints be recorded, via a digitizing camera, as digital images. Subsequently, the images may be viewed on a video display monitor. These operations are administered via a general-purpose computer. The crucial link among all of the aforementioned components is the image processing system. An image processing system contains storage modules called frame buffers which store the images during the acquisition and display stages.

A Grinnell GMR 270-series image processing system was used to digitize and display pictures of prints. These pictures are captured as monochrome images, using a television camera as the input device. Images are generated as a dot matrix grid consisting of 512 lines with 512 picture elements spaced along each line. The Grinnell provides a dynamic brightness range of 256 discrete gray levels contained in the interval $[0,255]$ for each pixel. A value of 0 corresponds to the dark elements while 255 corresponds to the brightest elements. Images are displayed on a Mitsubishi Model #C3922 LPK high-resolution color-television monitor.

A DEC VAX 11/750 digital computer was used as the host computer to control operation of the Grinnell.

4.2 COMPUTER PROGRAMS

Two programs were written to process the degraded print images: ENHANCE and COLOR. Both programs are interactive, which provides the user with some degree of flexibility. They are also menu-driven, facilitating ease of operation.

ENHANCE is an interactive program which performs a variety of spatial-domain and frequency-domain enhancement routines. A collection of unary and binary arithmetic operations are included so that various mathematical manipulations are possible. Statistical operations are incorporated to allow examination of the distribution of gray level values in an image.

The user is required to provide values of certain parameters in order to tailor the routines to specific applications. Menus are used to prompt the user for these values. Each menu displays the options, current values of parameters, and the last operation performed. The program records up to fifteen operations in a memory stack to assist the user in recalling previous operations.

ENHANCE contains two working registers, X and Y, and two

storage registers, 1 and 2. Each register is a 512 x 512 complex-valued data array. Unary operations are performed on data in the X register. Binary operations require data in both the X and Y registers. Results are always returned to the X register. Data which was previously stored in the X register is not saved due to the cost in speed and memory. The storage registers are available for storing intermediate results.

External data files are input to the program as real values between 0 and 255. The imaginary component is set to zero upon input. Data files are written as integer values between 0 and 255. Only the real component of the X register is written to external data files.

COLOR is a program used to perform psuedo-color enhancement, specifically, density slicing. This is a technique in which gray levels are divided into intervals and a specific color assigned to each interval. The image is then displayed using color planes to show the spatial locations of specific gray levels.

COLOR contains four registers. The registers are 512 x 512 integer-valued data arrays. The main register contains the image to be operated upon. Red, green, and blue color registers are used to display the partitioned intervals. Data may be accessed from any of the four registers.

CHAPTER 5: RESULTS

5.1 INTRODUCTION

This project involves the enhancement of degraded fingerprint images. Ridge detail is the most important feature in fingerprints used for identification. Therefore, the main thrust of the enhancement techniques used was to enhance the contrast between the regions corresponding to the ridges and valleys of the fingerprints.

Samples for this project were provided by Jeff Payne of the Department of Chemistry at Kansas State University. The prints appeared as a faint bluish outline on a piece of transparent 35 mm film acetate. Samples were the results of prints placed on a subject's arm and lifted at regular intervals.

A television camera connected to the Grinnell image processing system was used to capture the samples as monochrome digital images. The samples were placed on a light table in order to highlight the prints while photographing. Programs on the VAX 11/750 contained in the KSU IMAGE library were utilized to control operation of the television camera and frame buffers. The keyword IMAGE must be entered to initialize the programs. Once initialized, the keyword CAMERA is used to activate the television camera. When the desired

image is displayed on the monitor, the keyword STOIMG transfers the image to a 512 x 512 data file. Data files require approximately 514 blocks of memory; thus, care must be exercised so that the user's disk quota is not exceeded.

The sample documented in this thesis was recovered two hours after placement. Figure 5.1-1 shows a picture of this sample. A visual analysis of this image indicates some faint ridge detail in the center of the image, forming a pattern which appears to be a loop. Degradation is most severe in the lower portion of the image to the extent that individual ridges are indistinguishable. Ridges in the left half of the image are extremely faint. The bubbles which appear in the image are a result of the recovery process and cannot be removed.

The program COLOR was used to implement density slicing techniques as an aid in determining which gray levels comprise the ridge detail in the image. This was done by dividing the gray scale into intervals and assigning a color to each partitioned interval. A display of the composite image reveals the interval to which a spatial coordinate belongs.

Ridges were found to lie in the gray-level range from 6 to 160, with the majority of the distinguishable ridges con-

tained in the range from 100 to 120. Ridge information in the lower section of the image, where the worst degradation occurs, falls into the interval from 6 to 100. The far left and right regions of the image reveal very faint ridges which occupy the range from 130 to 160. The one characteristic which became evident through density slicing was that the ridges were not uniform with many different gray values comprising a single ridge.

A statistical analysis of the gray level content in the image was performed using the program ENHANCE. Gray levels for this image were found to lie in the range from 6 to 252. These gray levels are distributed according to the histogram shown in Figure 5.1-2. A mean value of 135 with a variance of 758 was calculated from this distribution. The histogram indicates that most of the gray level information is contained in the interval from 60 to 245.

Enhancement operations were executed through the program ENHANCE. Various combinations of spatial-domain and frequency-domain operations were tried until the best techniques were discovered. No single technique is optimal since the range of contrast and brightness can vary greatly within any given image. Filtering in the frequency domain was found to produce the best results for sharpening ridge detail. Spatial domain techniques proved useful in enhanc-

ing contrast in the image. Two procedures which yielded good results are detailed in this chapter.

5.2 PROCEDURE ONE

The first procedure utilizes contrast stretching operations to increase the degree of contrast present in the image. Contrast stretching functions well when most of the gray levels are contained within a well-defined range. Filtering is also used to enhance the edges between the ridges and the valleys.

A linear transformation function was used initially so that ridge detail would be preserved. The transformation is given as

$$g(x,y) = \begin{cases} 0 & \text{if } f(x,y) < c_1 \\ \frac{255[f(x,y) - c_1]}{(c_h - c_1)} & c_1 \leq f(x,y) \leq c_h \\ 255 & f(x,y) > c_h \end{cases} \quad (5.2-1)$$

where c_1 and c_h are the cutoff points. This transformation is shown graphically in Figure 5.2-1. The cutoff points c_1 and c_h chosen for the image in Figure 5.1-1 were 70 and 240, respectively. These cutoff points caused a small percentage of pixels outside of the cutoff range to be transformed to the extreme values 0 and 255. Figure 5.2-2

shows a picture of the transformed image. Although the improvement in contrast is not glaringly obvious to the human eye, it is quite beneficial to the subsequent filtering operations. A plot of the histogram, shown in Figure 5.2-3, exhibits an expanded gray level range which gives credence to the name contrast stretching. The increase in the degree of contrast is reinforced by the increase in variance to 1904. A reduction in the mean value to 103 occurred as a result of the transformation.

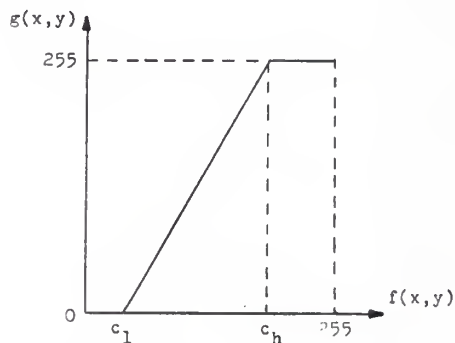


Figure 5.2-1 Linear contrast stretching transformation function

Filtering is performed to accentuate the edges between the ridges and valleys of the print. This operation requires that the origin in the frequency domain be translated to the center of the frequency plane. Translation is achieved through the relationship

$$f(x,y)\exp[j2\pi(u_0x + v_0y)] \Leftrightarrow F(u-u_0,v-v_0) \quad (5.2-2)$$

where the symbol \Leftrightarrow denotes "Fourier transform pair."

Once translated, the image is transformed to the frequency domain via the DFT. A useful relationship for displaying the transformed image is given by

$$F'(u,v) = \text{Log}[1 + |F(u,v)|] \quad (5.2-3)$$

This relationship is useful for displaying pixels of relatively small magnitude. A picture of $F'(u,v)$ is shown in Figure 5.2-4. Examination of this picture indicates that most of the frequency information is located towards the center of the frequency plane. A significant amount of information lies on the horizontal and vertical axes. The spikes located on the diagonals correspond to noise and periodicity resulting from circular symmetry in the ridge pattern.

Highpass filtering in the frequency domain is used to accentuate edge detail in the spatial domain due to the fact that edges contribute greatly to high-frequency components. A highpass-filter transfer function is one which attenuates frequency components within a specific radius about the origin defined by the cutoff point. The cutoff point is chosen by examining the energy distribution in the frequency domain. This distribution represents the amount of energy contained within a certain radius about the origin.

There are a number of transfer functions which can be used

for highpass filtering. The difference among transfer functions is described by the manner of transition between the minimum and the maximum values. A Butterworth highpass transfer function was used to filter the image in Figure 5.2-4. This transfer function is given by

$$H(u,v) = \frac{1}{1 + 0.414[D_0/D(u,v)]^{2n}} \quad (5.2-4)$$

where D_0 is the cutoff point, n is the filter order, and $D(u,v)$ is the distance from the origin of the pixel at the coordinate (u,v) . Note that the value of the transfer function at D_0 is 0.707 rather than 0.5. The energy distribution revealed that 9.24% of the energy was contained within a radius of 15 pixels about the origin. Consequently, the image was filtered using a Butterworth highpass filter of order $n=1$ with a cutoff point of $D_0=15$.

After completion of the filtering operation, the filtered image is transformed back to the spatial domain using the inverse DFT. The translation process is then repeated to restore the image to its original phase.

The values for the filtered image were found to range from -173 to 86 which is well outside the acceptable range of gray level values. The histogram for this image, shown in Figure 5.2-5, indicates that a very small amount of the gray

levels are contained below the value -80. As a result, the range was truncated to the interval from -80 to 86 by thresholding as

$$g(x,y) = \begin{cases} f(x,y) & \text{if } f(x,y) > -80 \\ -80 & \text{otherwise} \end{cases} \quad (5.2-5)$$

Translation to an acceptable range was obtained by addition of an offset constant, specifically,

$$g_1(x,y) = g(x,y) + 80. \quad (5.2-6)$$

Figure 5.2-6 shows a picture of the resulting image. The transformed gray levels range from 0 to 166. Examination of the histogram, shown in Figure 5.2-7, indicates that most of the gray levels are contained within the interval from 0 to 140. The distribution generated a mean value of 81 with a variance of 385.

The gray-level range was expanded through contrast stretching. A parabolic transformation function was used so that larger values would be affected to a greater extent. The transformation function is expressed as

$$g(x,y) = \begin{cases} 0 & \text{if } f(x,y) < c_1 \\ \frac{255[f(x,y) - c_1]^2}{(c_h - c_1)^2} & c_1 \leq f(x,y) \leq c_h \\ 255 & f(x,y) > c_h \end{cases} \quad (5.2-7)$$

where c_1 and c_h are the cutoff points. Figure 5.2-8 graph-

ically depicts the transformation function.

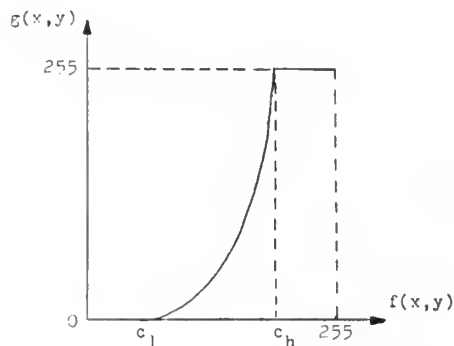


Figure 5.2-8 Parabolic contrast stretching transformation function

The cutoff points $c_l=0$ and $c_h=140$ were chosen for the parabolic transformation. Notable improvement in the ridges of the far left and right portions of the image is observed in the resulting image shown in Figure 5.2-9. Examination of the histogram, shown in Figure 5.2-10, indicates that the distribution is skewed towards the darker end of the gray scale. An increase in the mean value to 91 is a result of the expanded range of gray levels; however, the mean is small due to the skewness of the distribution. Contrast improvement is supported by an increase in the variance to 1598.

Histogram equalization was performed on the image in Figure 5.2-9 to redistribute the gray levels uniformly. An increase in the degree of contrast is evident in the equalized

image pictured in Figure 5.2-11. A substantial improvement in ridge detail is apparent in the lower portion of the image where a majority of the degradation originally occurred. The histogram, shown in Figure 5.2-12, indicates that the distribution is indeed more nearly uniform. Evidence of uniformity is reinforced by a mean value of 128. An increase in the variance to 5408 is representative of the considerable contrast enhancement.

Smoothing operations were used to remove noise from the equalized image as well as produce uniform ridges. The image in Figure 5.2-11 was smoothed through neighborhood averaging using the 3 x 3 spatial mask shown in Figure 3.2-3. A picture of the smoothed image is shown in Figure 5.2-13, with the corresponding histogram plotted in Figure 5.2-14. This operation resulted in a slight decrease in the mean and variance to 127 and 3928, respectively, which is typical of smoothing operations. Nevertheless, the degree of contrast is quite acceptable.

For the purpose of comparison, the equalized image in Figure 5.2-11 was also smoothed using a 3 x 3 median filter. The resulting image is pictured in Figure 5.2-15. To some extent, the image is less blurred than its averaged counterpart. Comparison of the histogram, shown in Figure 5.2-16, with Figure 5.2-12 indicates a slight decrease in contrast,

which is corroborated by a reduction in the variance to 4740. This reduction is not as great as that of the averaging operation, which indicates a greater degree of edge preservation. Essentially, the mean was unchanged by the median filtering operation.

Edges in the smoothed image of Figure 5.2-13 were enhanced via the gradient operation. Detection of edges in the horizontal and vertical directions was achieved through the use of five-level masks denoted as G_x and G_y , respectively, and shown in Figure 5.2-17. The composite gradient, denoted as G , was computed as

$$G = \max[|G_x|, |G_y|] \quad (5.2-8)$$

| | | |
|----|----|----|
| -1 | -2 | -1 |
| 0 | 0 | 0 |
| 1 | 2 | 1 |

Horizontal

| | | |
|----|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Vertical

Figure 5.2-17 Spatial masks used to compute horizontal and vertical gradients

Refinement of the edges was achieved by subtracting the absolute value of the Laplacian gradient from the composite gradient [4]. The Laplacian is a second-order derivative computed using the spatial mask shown in Figure 5.2-18.

| | | |
|---|----|---|
| 0 | 1 | 0 |
| 1 | -4 | 1 |
| 0 | 1 | 0 |

Figure 5.2-18 Spatial mask used to compute Laplacian gradient

Once the modified gradient is computed, an appropriate threshold is chosen so that only relevant edge detail remains. Generally, thresholding is performed using the relationship

$$G_t = \begin{cases} 0 & \text{if } G < t \\ 255 & \text{if } G \geq t \end{cases} \quad (5.2-9)$$

The thresholding operation was modified for the enhancement operation so that

$$G_t = \begin{cases} 1 & \text{if } G < t \\ 255 & \text{if } G \geq t \end{cases} \quad (5.2-10)$$

The histogram for the modified gradient, shown in Figure 5.2-19, is helpful in selecting a proper threshold. Values for the gradient were found to range from 0 to 838. Using trial and error, a threshold of 300 was chosen for the gradient G_t . Figure 5.2-20 shows a picture of the gradient after thresholding.

Edges were enhanced in the smoothed image of Figure 5.2-13 using the relation

$$g(x,y) = \begin{cases} f(x,y) & \text{if } G < t \\ 255 & \text{if } G \geq t \end{cases} \quad (5.2-11)$$

where $f(x,y)$ is the smoothed image and t is the threshold value chosen for G_t in Equation (5.2-10). This operation was accomplished by multiplying the smoothed image in Figure 5.2-13, denoted as $f(x,y)$, by the threshold gradient G_t of Equation (5.2-10) and truncating all values above 255. The enhanced image, shown in Figure 5.2-21, reveals greater contrast and significantly sharper ridge detail as a result of the operation. The mean and variance increased to 139 and 5136, respectively.

A block diagram detailing Procedure One is presented in Figure 5.2-22.

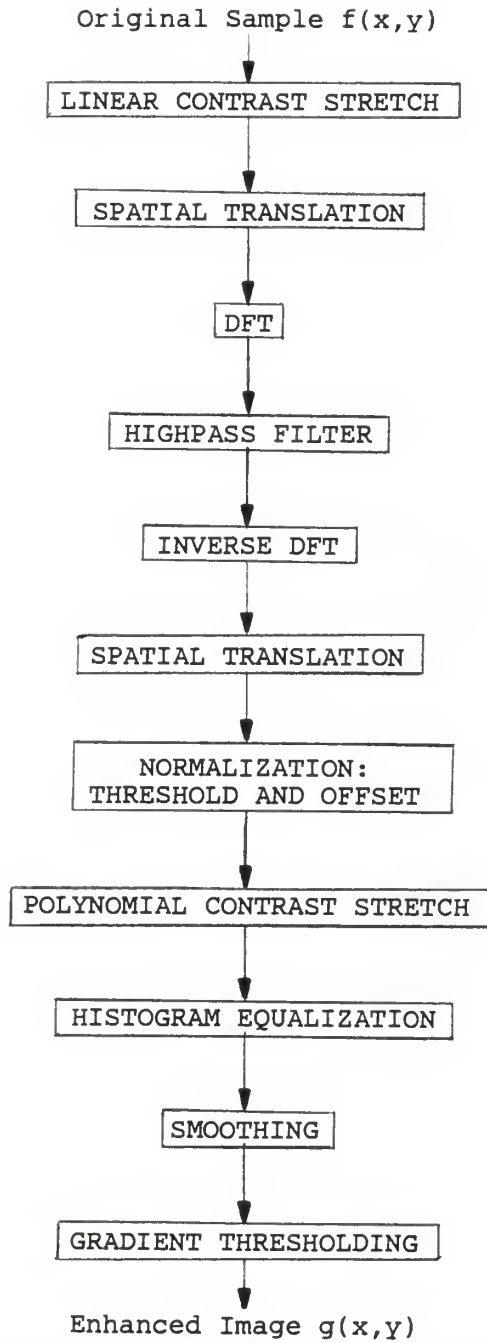


Figure 5.2-22 Block diagram for Procedure One

5.3 PROCEDURE TWO

The second procedure executes smoothing operations prior to filtering. Smoothing the image provides a degree of homogeneity so that uniform ridges are presented to the filtering operations. Smoothing can be accomplished using either neighborhood averaging or median filtering.

Neighborhood averaging was used to smooth the image in Figure 5.1-1. This operation was executed using the spatial mask of Figure 3.2-3. Noticable improvement in ridge definition as well as a meaningful reduction in noise is evident in the smoothed image shown in Figure 5.3-1. The histogram, shown in Figure 5.3-2, produced a slight decrease in the mean value to 134. However, the variance actually increased to 837 which is unusual for a smoothing operation.

Figure 5.3-3 shows the result of smoothing the image in Figure 5.1-1 using a 3 x 3 median filter. Median filtering produced a lesser degree of uniformity than neighborhood averaging as seen in the varying degree of contrast in the image. The accompanying histogram is shown in Figure 5.3-4. This distribution yielded a decrease in the variance to 725 while the mean was unchanged at 135.

The smoothing operation was followed by highpass filtering in order to enhance the ridge detail. As mentioned previ-

ously, the smoothed image is translated prior to transformation to the frequency domain so that the origin is at the center of the frequency plane. Figure 5.3-5 shows the image of Figure 5.3-1 in the frequency domain displayed using the relation in Equation (5.2-3). The effects of smoothing are observed when this image is compared to the image in Figure 5.2-4. A reduction of high-frequency components along with the clustering of low-frequency components is characteristic of image smoothing. Removal of the spikes in the second and fourth quadrants is evidence of a reduction in noise.

Filtering was performed using an exponential highpass filter. The transfer function for the exponential highpass filter is given as

$$H(u,v) = \exp[-0.347(D_0/D(u,v))^{2n}] \quad (3.3-1)$$

where n is the filter order, D_0 the cutoff frequency, and $D(u,v)$ the distance from the origin of the pixel at the spatial coordinate (u,v) .

Calculation of the energy distribution revealed that a radius of 8 pixels about the origin contains 10.335% of the energy. Thus, the filter parameters were chosen as $D_0=8$ and $n=1$. After filtering, the image was transformed back to the spatial domain using the inverse DFT followed by a spatial

translation to recover the original phase information.

Post-filtering gray level values ranged from -189 to 87. Examination of the histogram, shown in Figure 5.3-6, indicates that very few pixels contain values less than -50. Therefore, the image was truncated using a threshold of $t=-50$. An offset constant of $c=50$ was used to translate the gray level range to the interval from 0 to 137. The normalized image is pictured in Figure 5.3-7. A plot of the normalized histogram, shown in Figure 5.3-8, produced a mean of 46 and a variance of 193.

Histogram equalization was executed in an effort to redistribute the gray levels more uniformly throughout the range from 0 to 255. A picture of the equalized image, shown in Figure 5.3-9, exhibits a noticeable improvement in contrast. Considerable improvement is visible in reducing the amount of degradation in the lower portion of the image so that individual ridges are distinguishable. Most importantly, faint ridges in the far left and right portions of the image are appreciably enhanced. The equalized histogram, plotted in Figure 5.3-10, yielded a mean of 130 and variance of 5435.

A block diagram of Procedure Two is detailed in Figure 5.3-11.

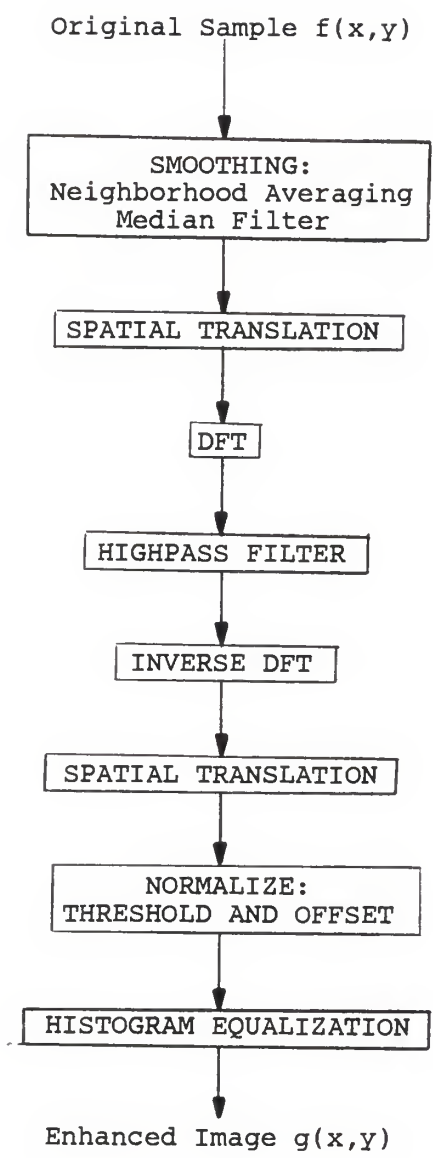


Figure 5.3-11 Block diagram for Procedure Two

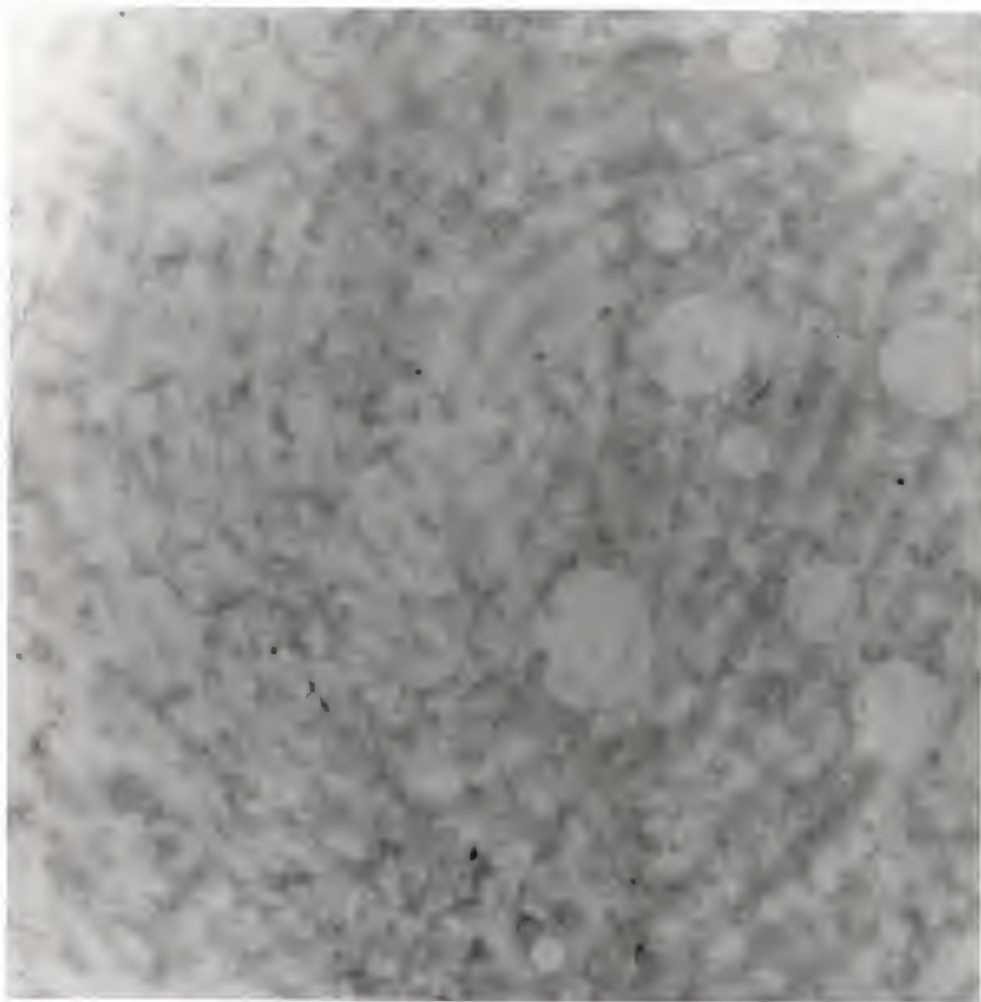


Figure 5. 1-1 Sample recovered after two hours

Histogram: 415A120.DAT

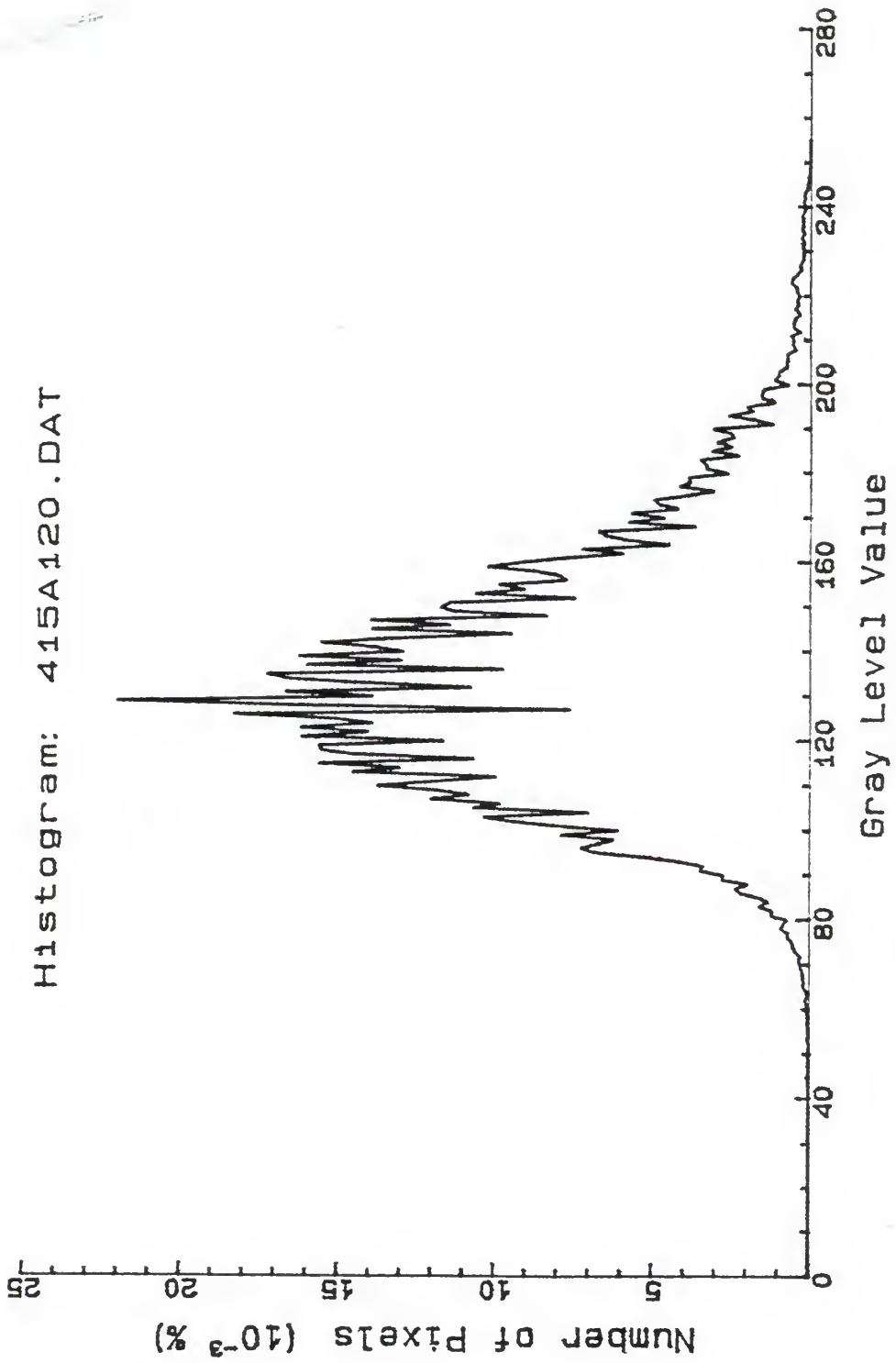


Figure 5.1-2 Histogram of image in Figure 5.1-1

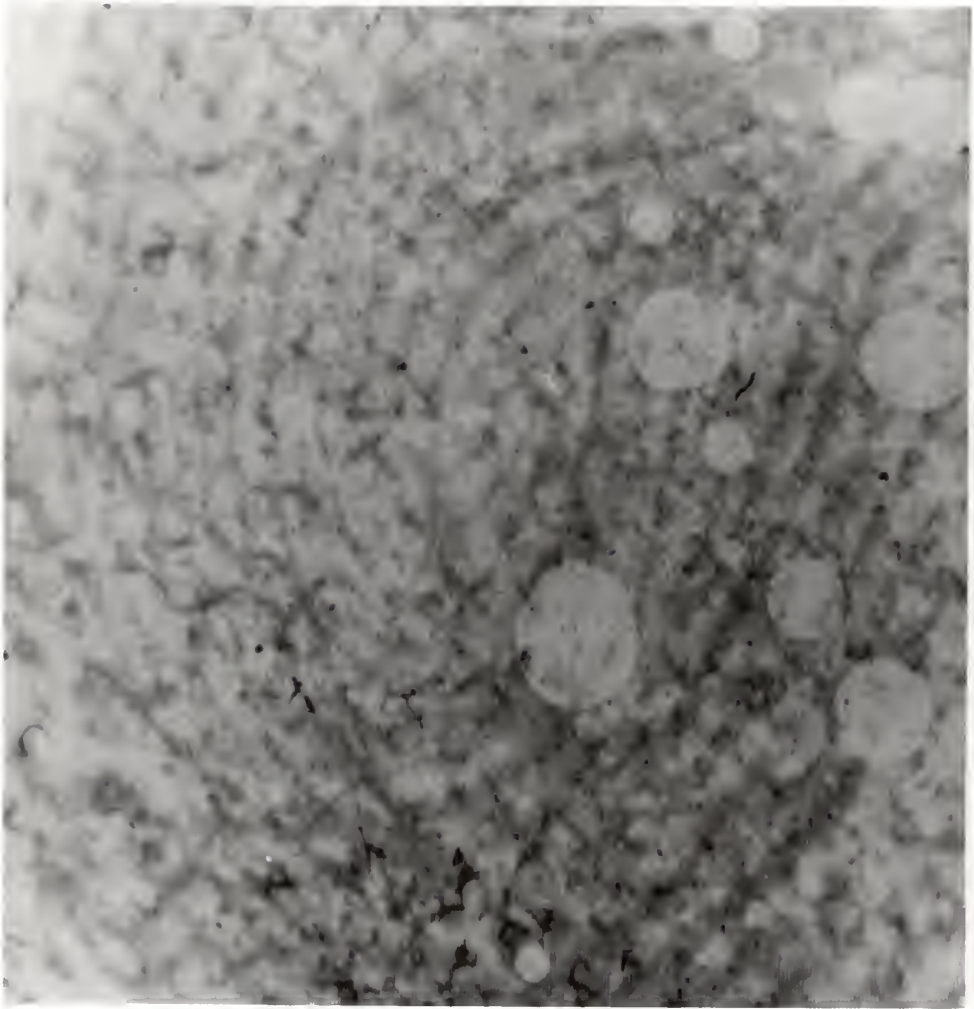


Figure 5. 2-2 Result of linear contrast stretching applied to image in Figure 5. 1-1

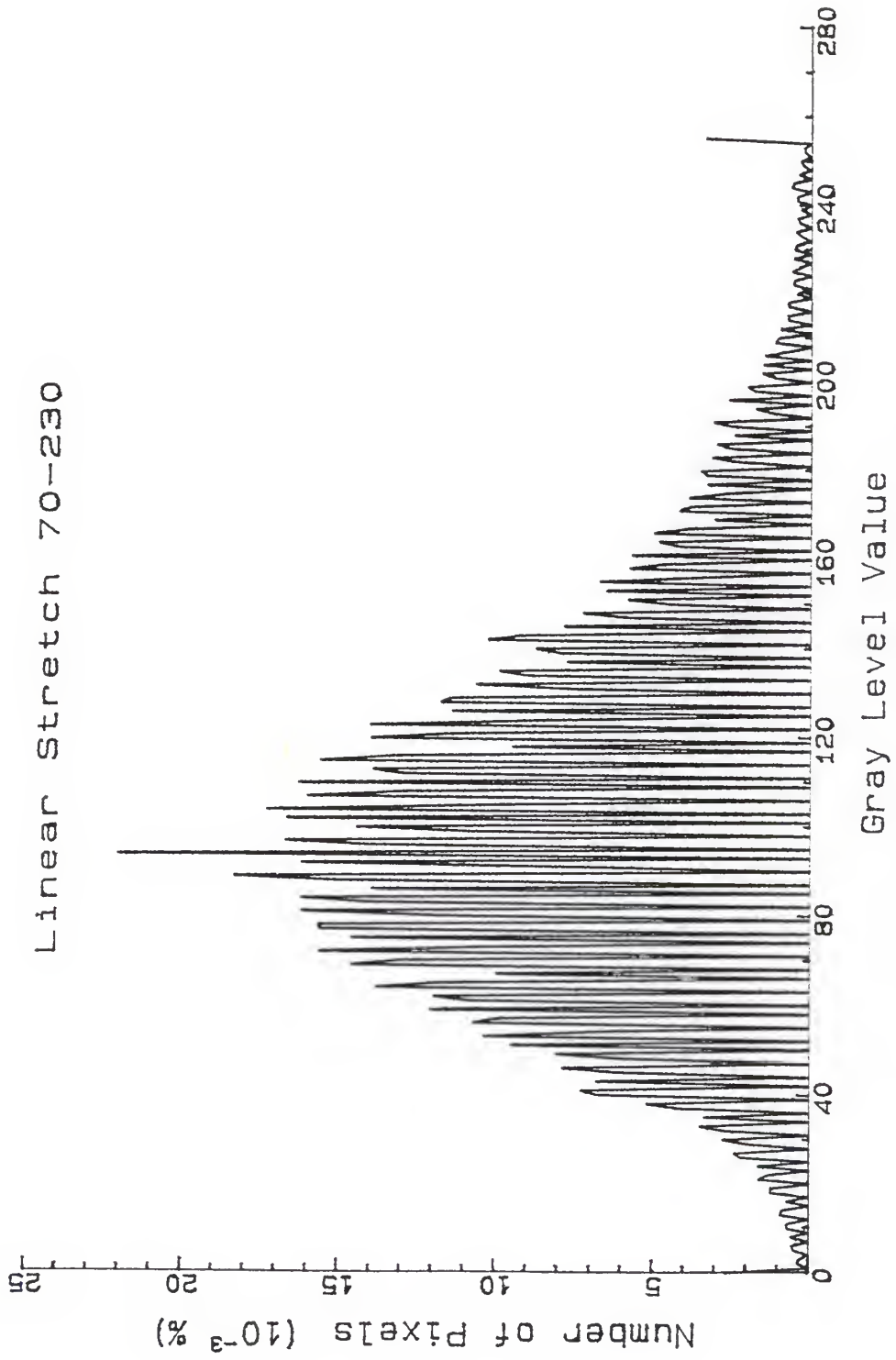


Figure 5.2-3 Histogram of image in Figure 5.2-2



Figure 5.2-4 Result of transforming image in Figure 5.2-2 to the frequency domain and displaying using Equation (5.2-3)

Butterworth, $c=15$, $n=1$

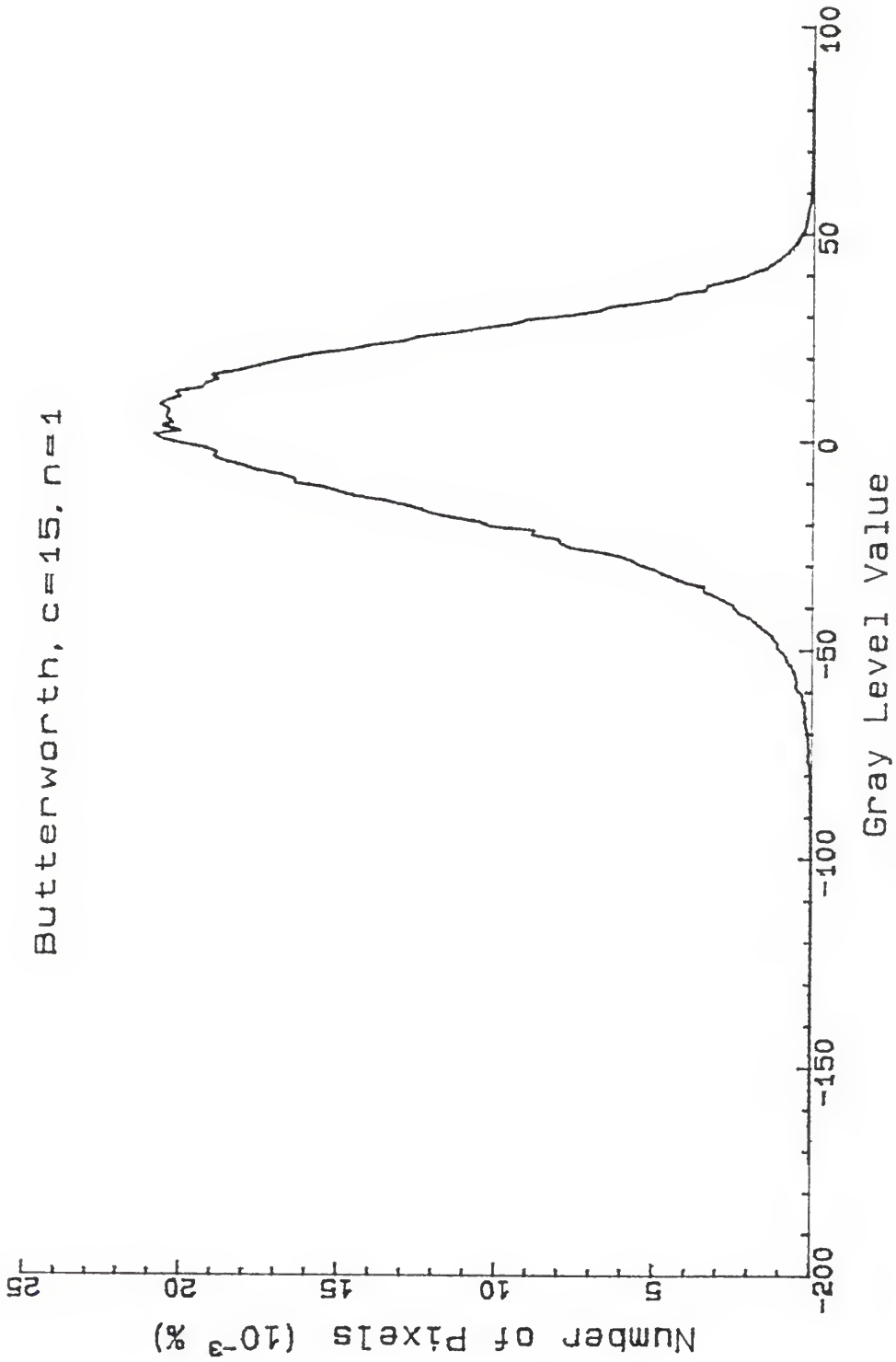


Figure 5.2-5 Histogram of filtered image before normalization

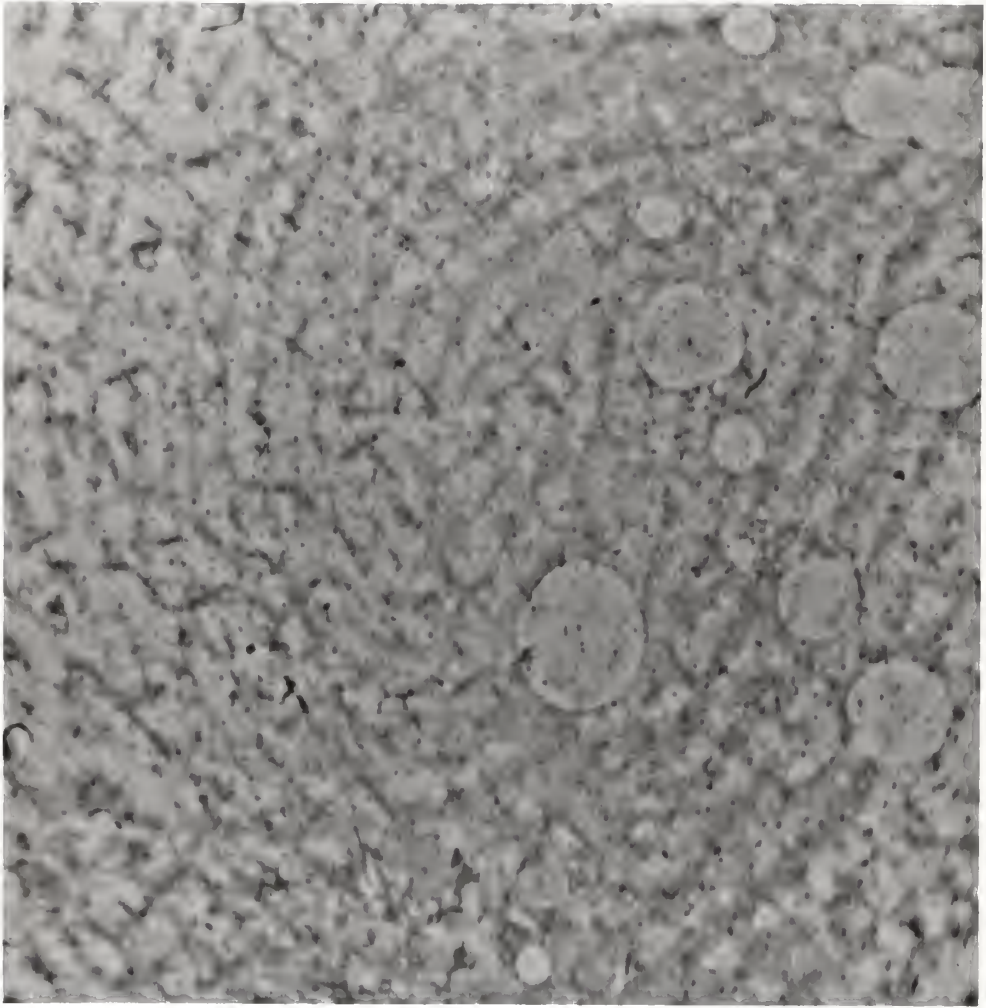


Figure 5.2-6 Result of highpass filtering and normalization applied to image in Figure 5.2-2.

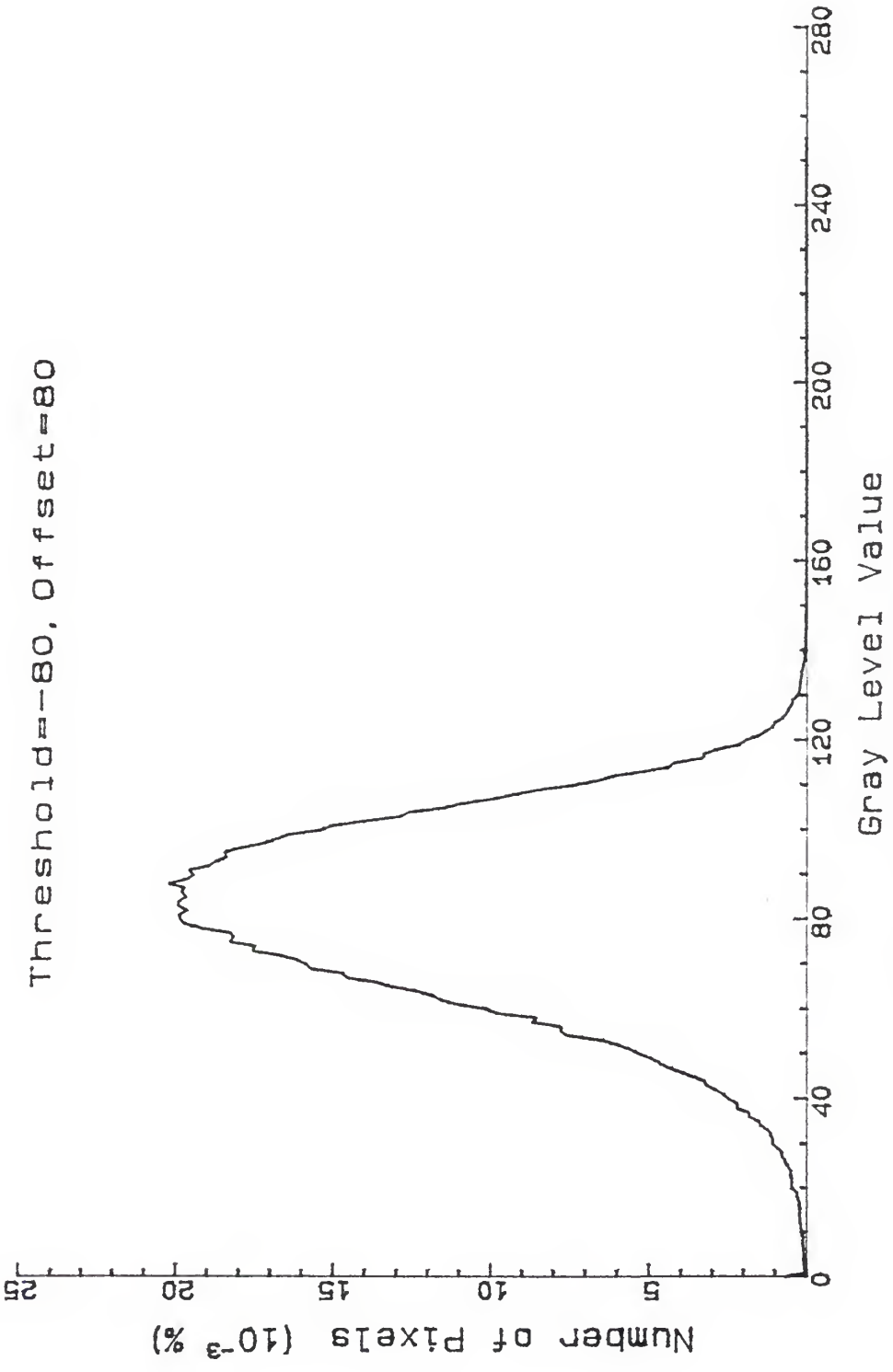


Figure 5.2-7 Histogram of image in Figure 5.2-6

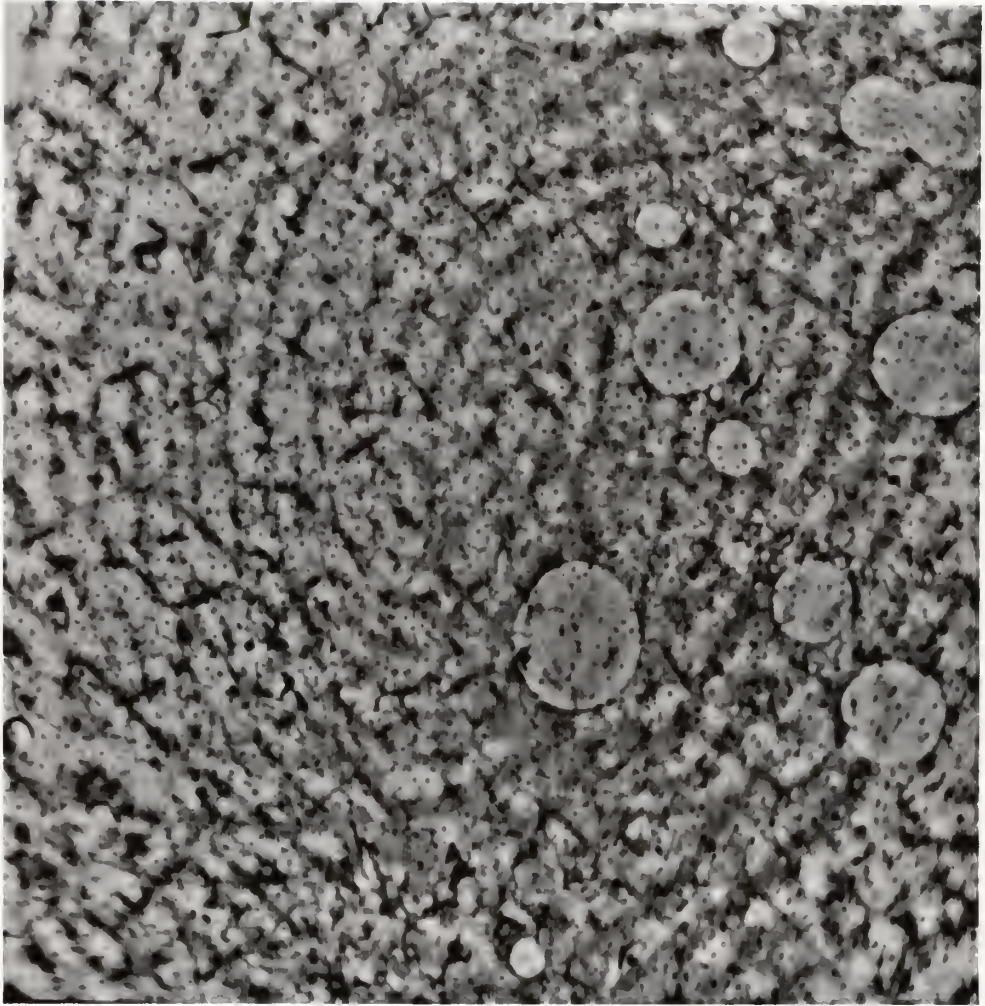


Figure 5.2-9 Result of parabolic contrast stretching applied to image in Figure 5.2-6

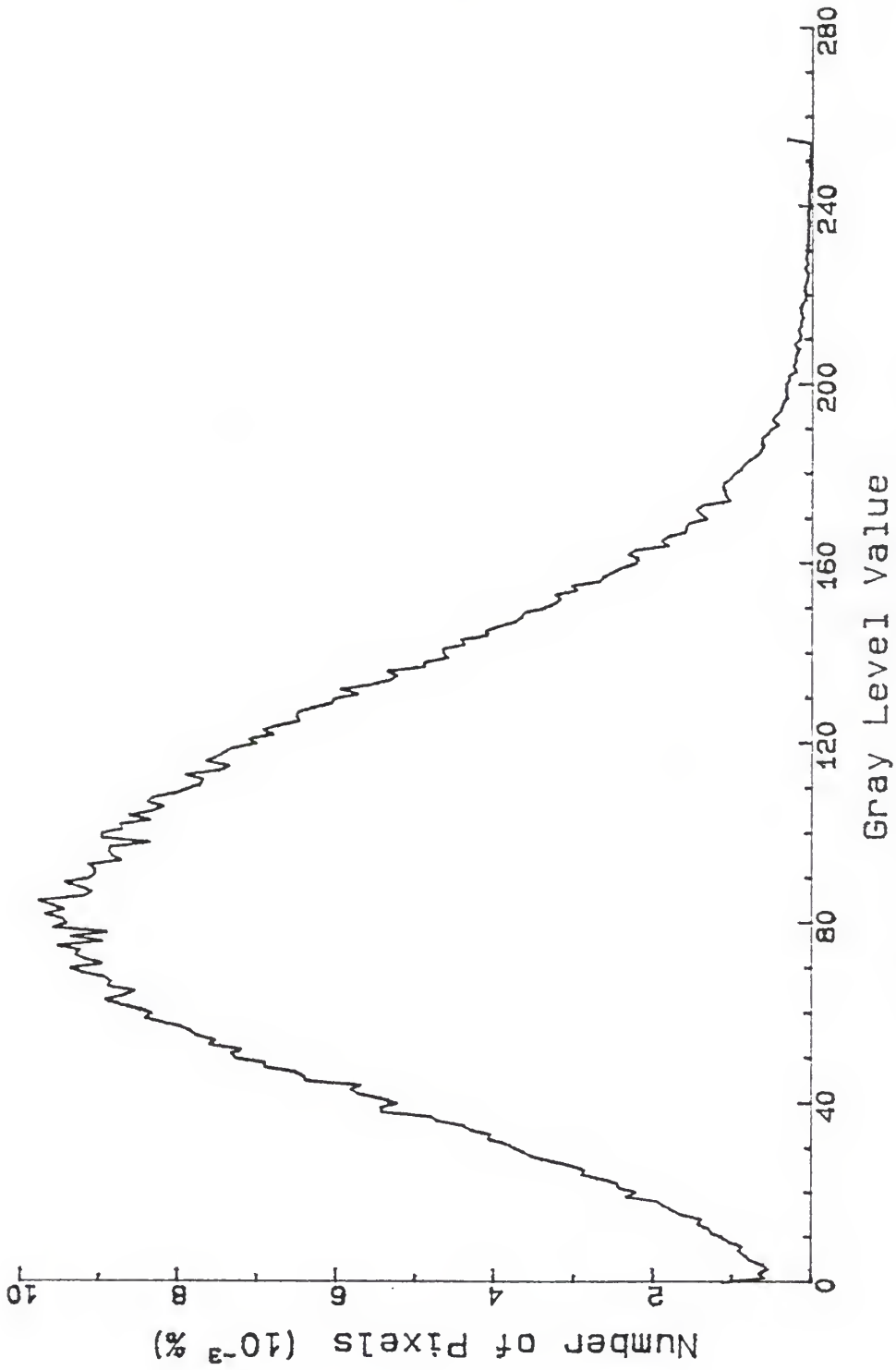


Figure 5.2-10 Histogram of image in Figure 5.2-9

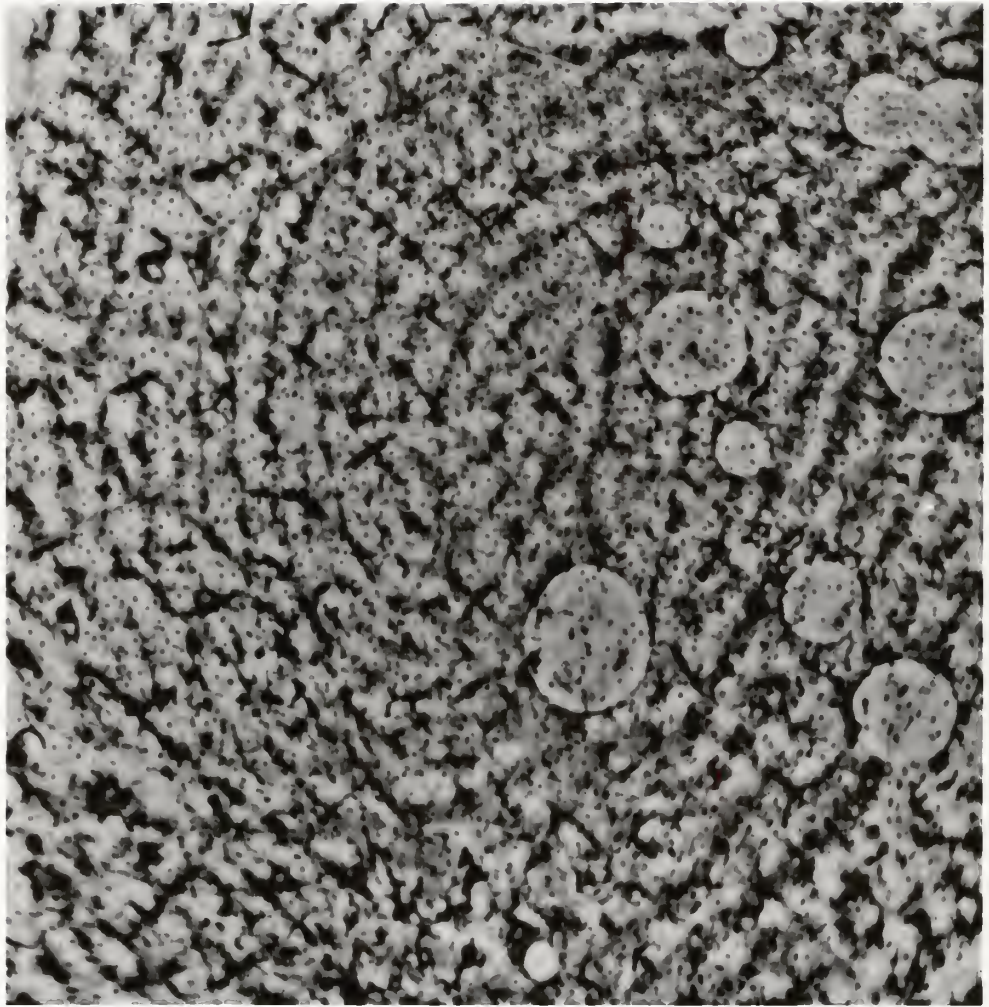


Figure 5.2-11 Result of histogram equalization applied to image in Figure 5.2-9

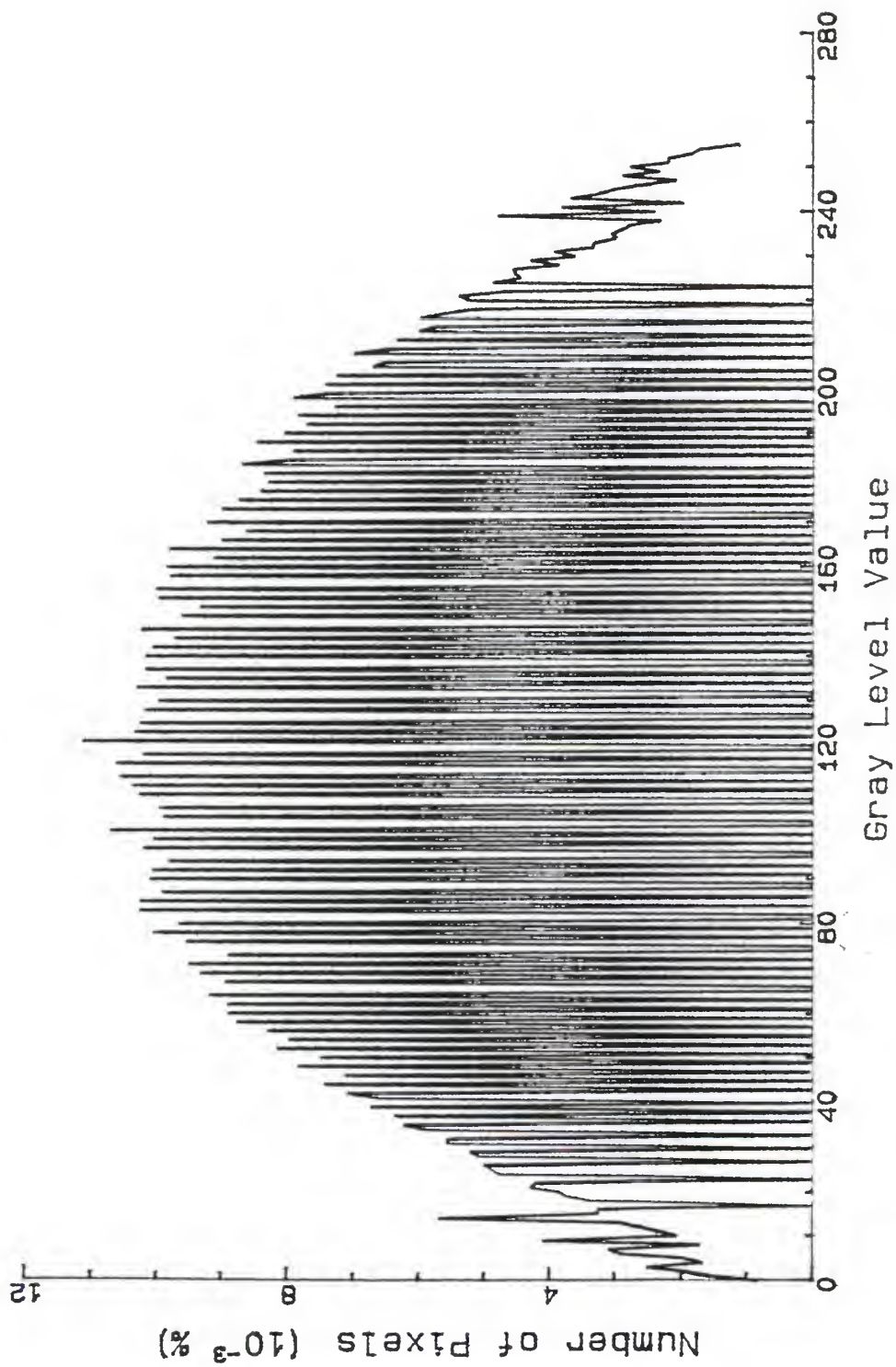


Figure 5.2-12 Histogram of image in Figure 5.2-11

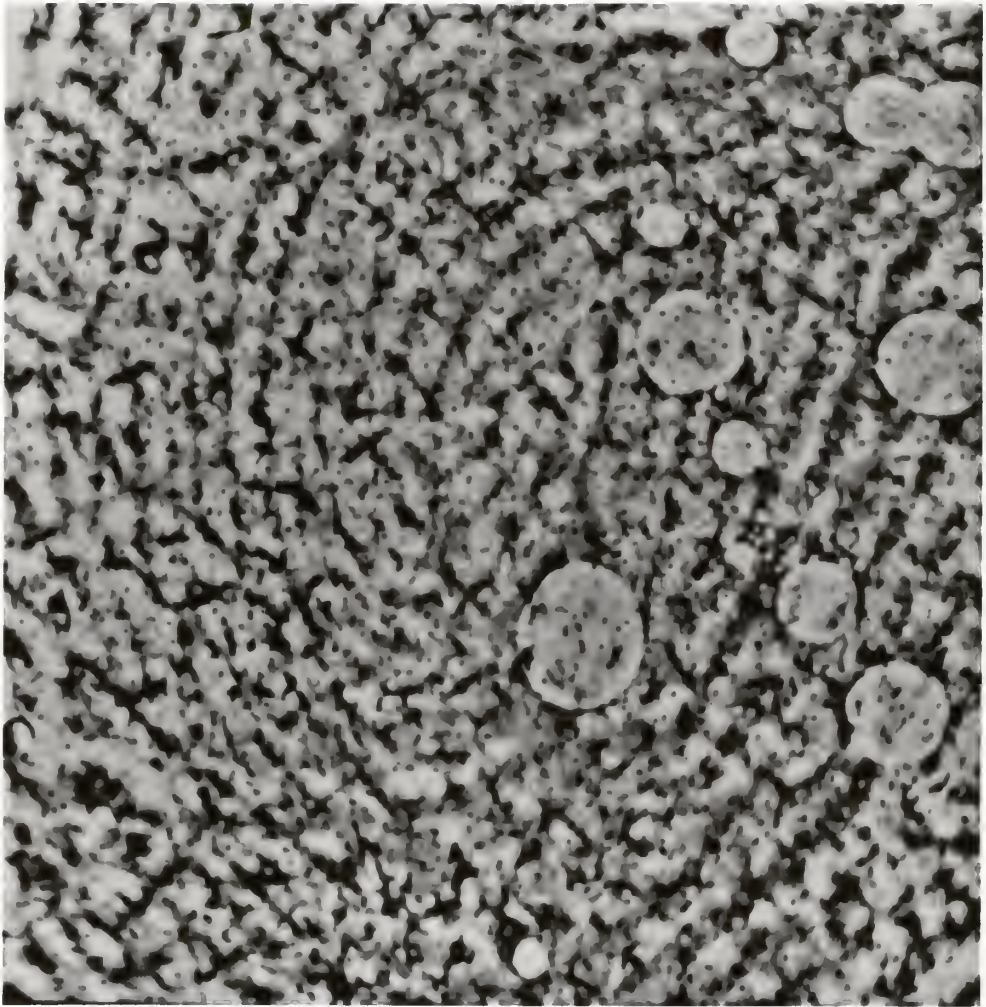


Figure 5.2-13 Result of neighborhood averaging applied to image in Figure 5.2-11

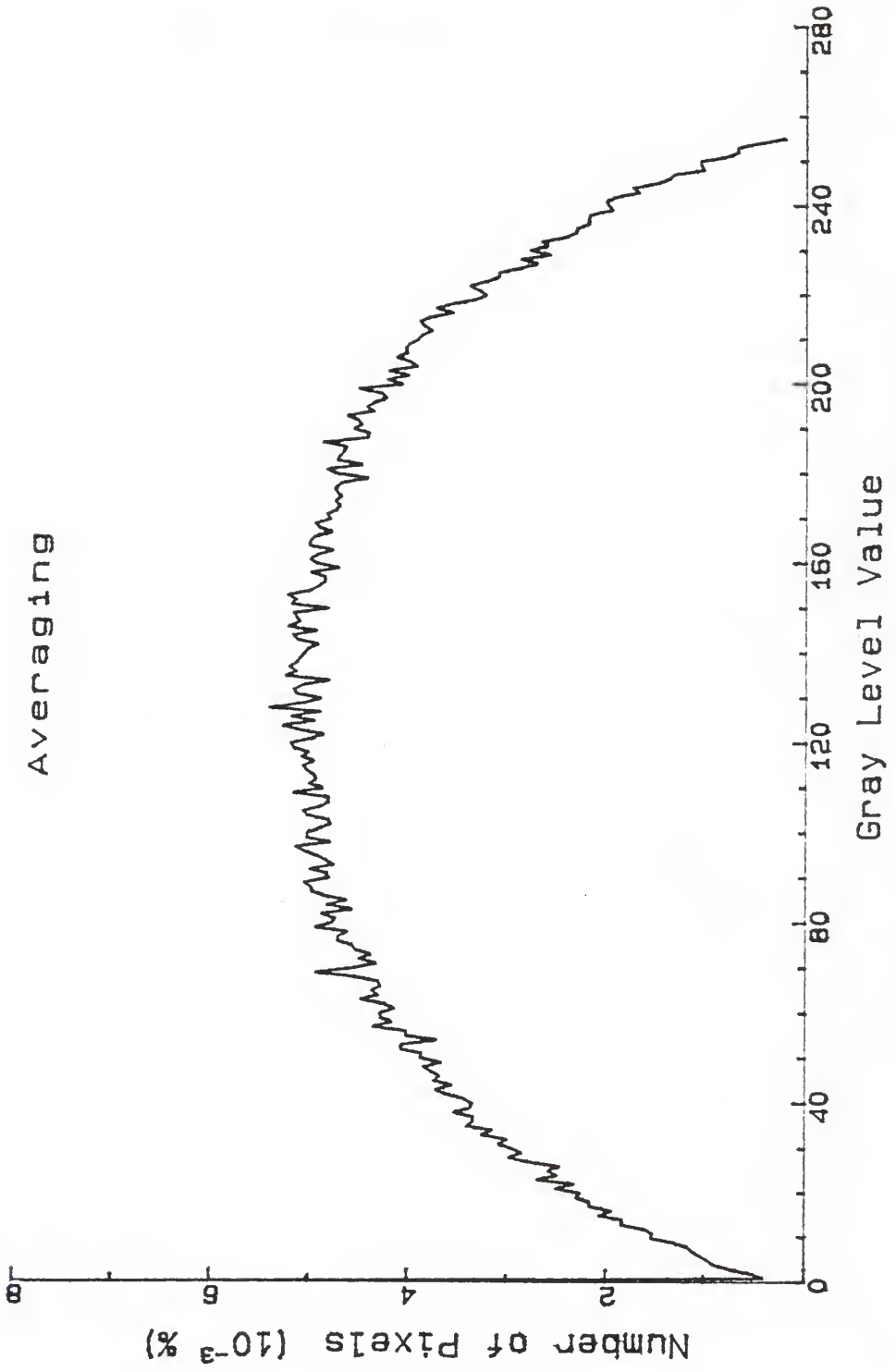


Figure 5.2-14 Histogram of image in Figure 5.2-13

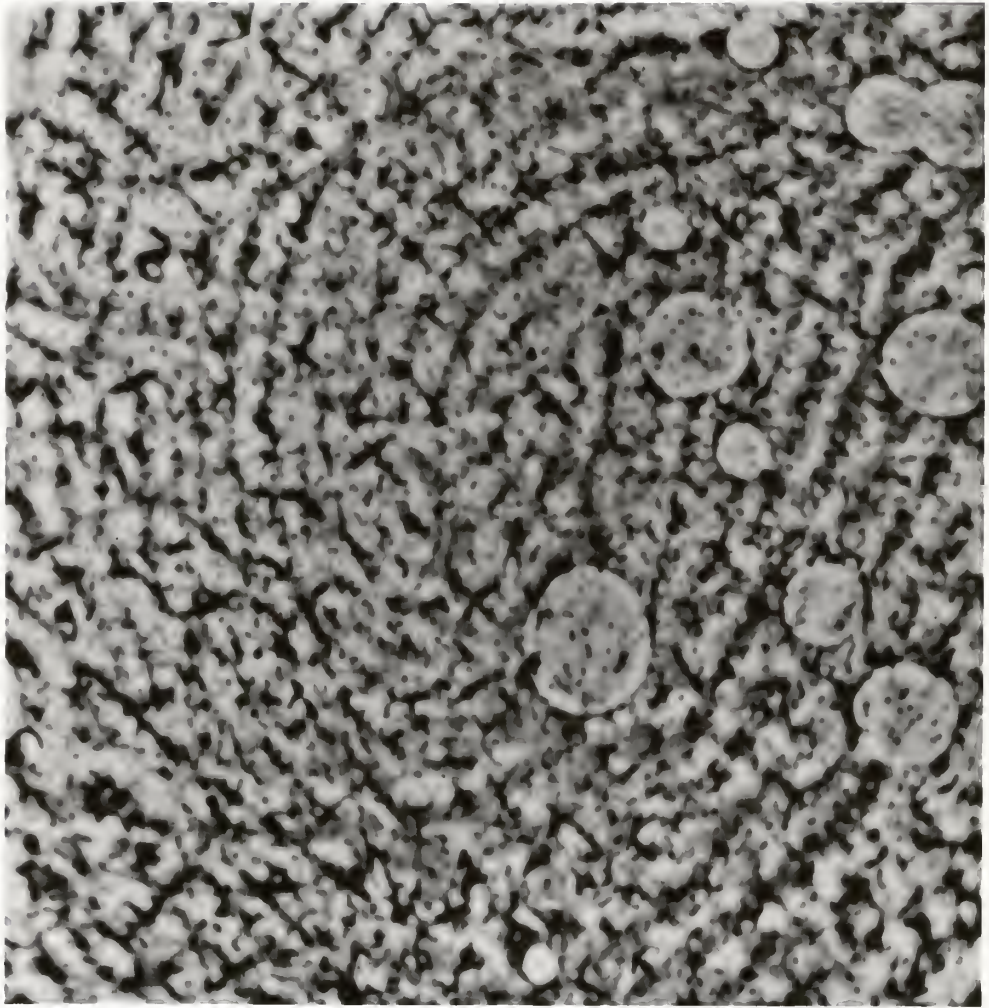


Figure 5.2-15 Result of applying 3 x 3 median filter to image in Figure 5.2-11

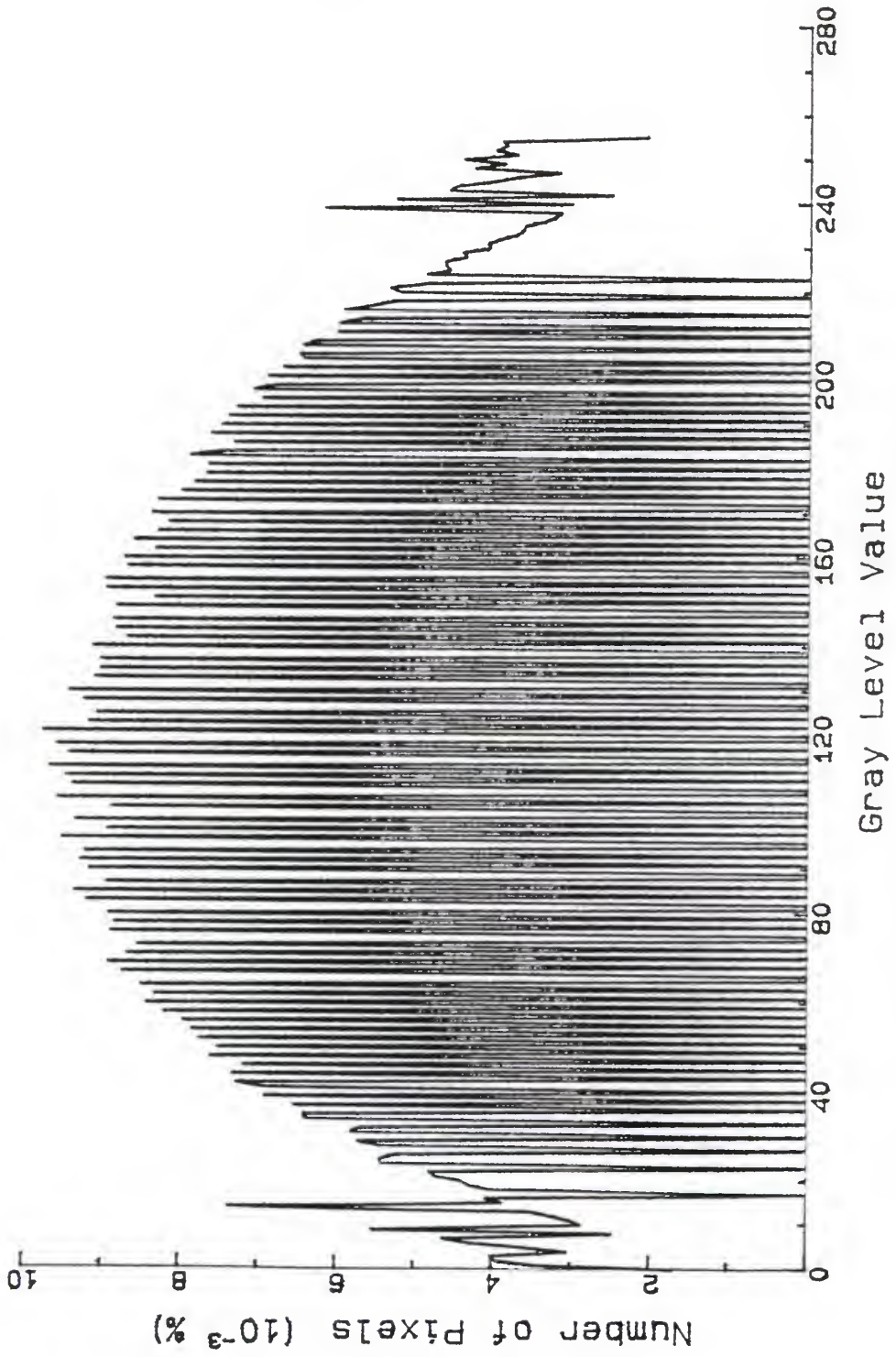


Figure 5.2-16 Histogram of image in Figure 5.2-15

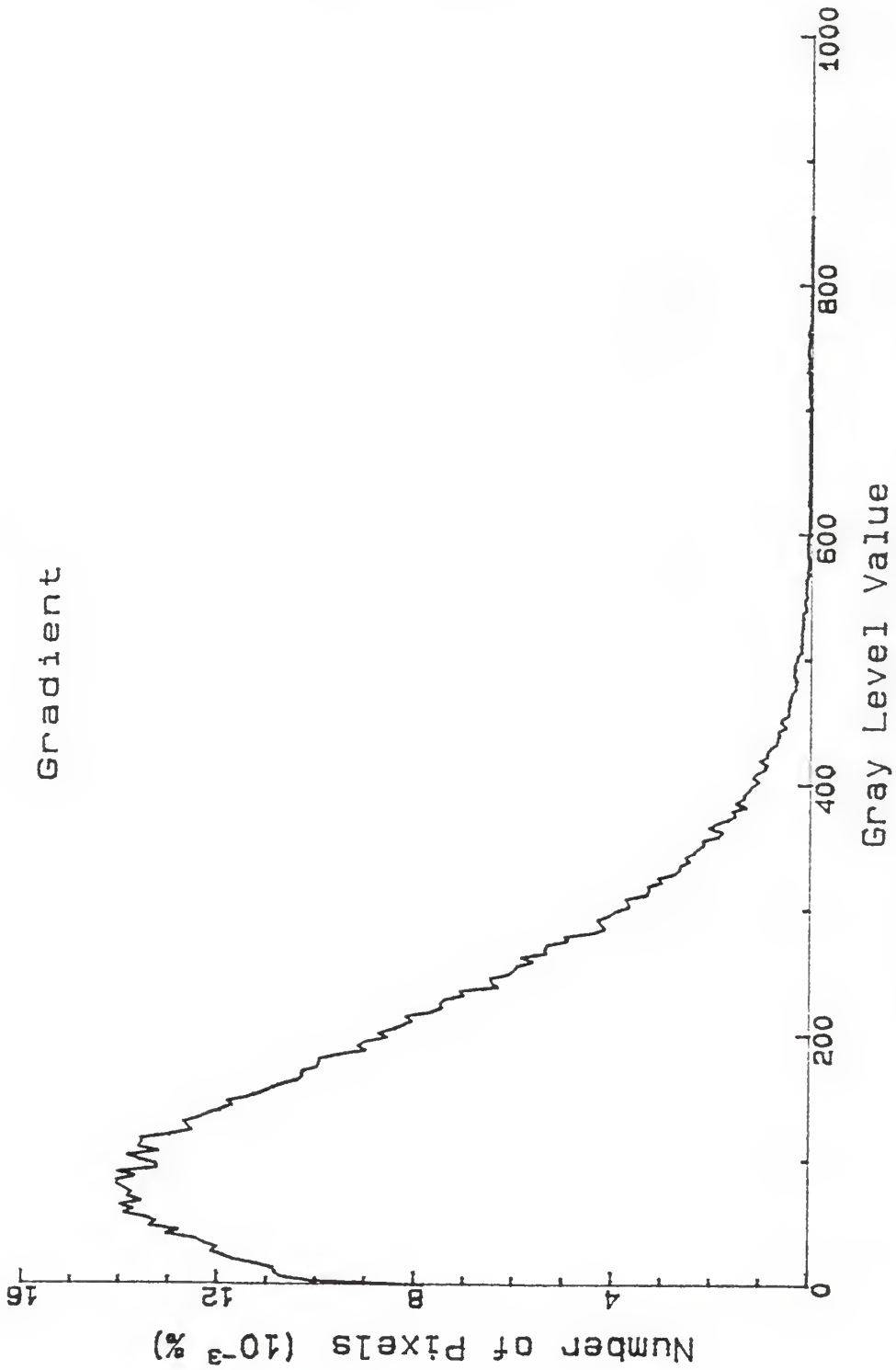


Figure 5.2-19 Histogram of gradient for image in Figure 5.2-13

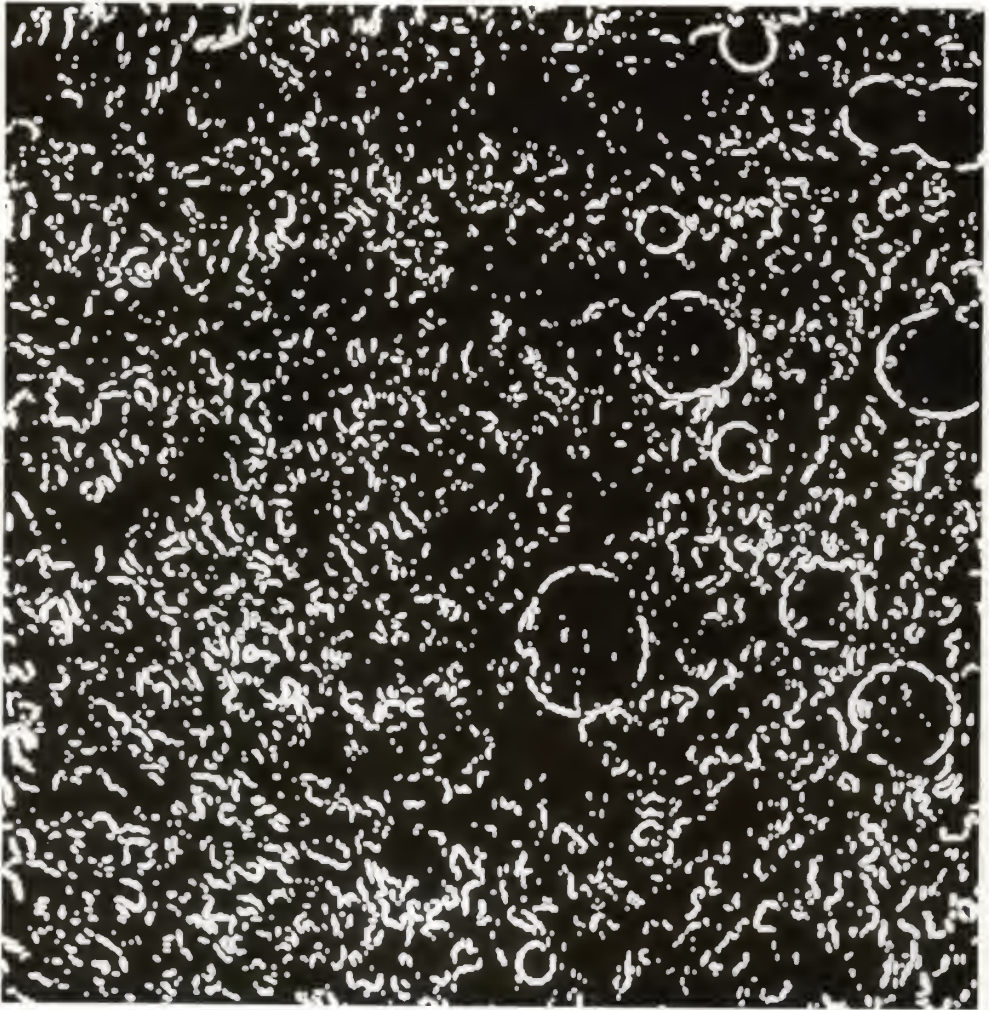


Figure 5.2-20 Gradient of image in Figure 5.2-13 after thresholding

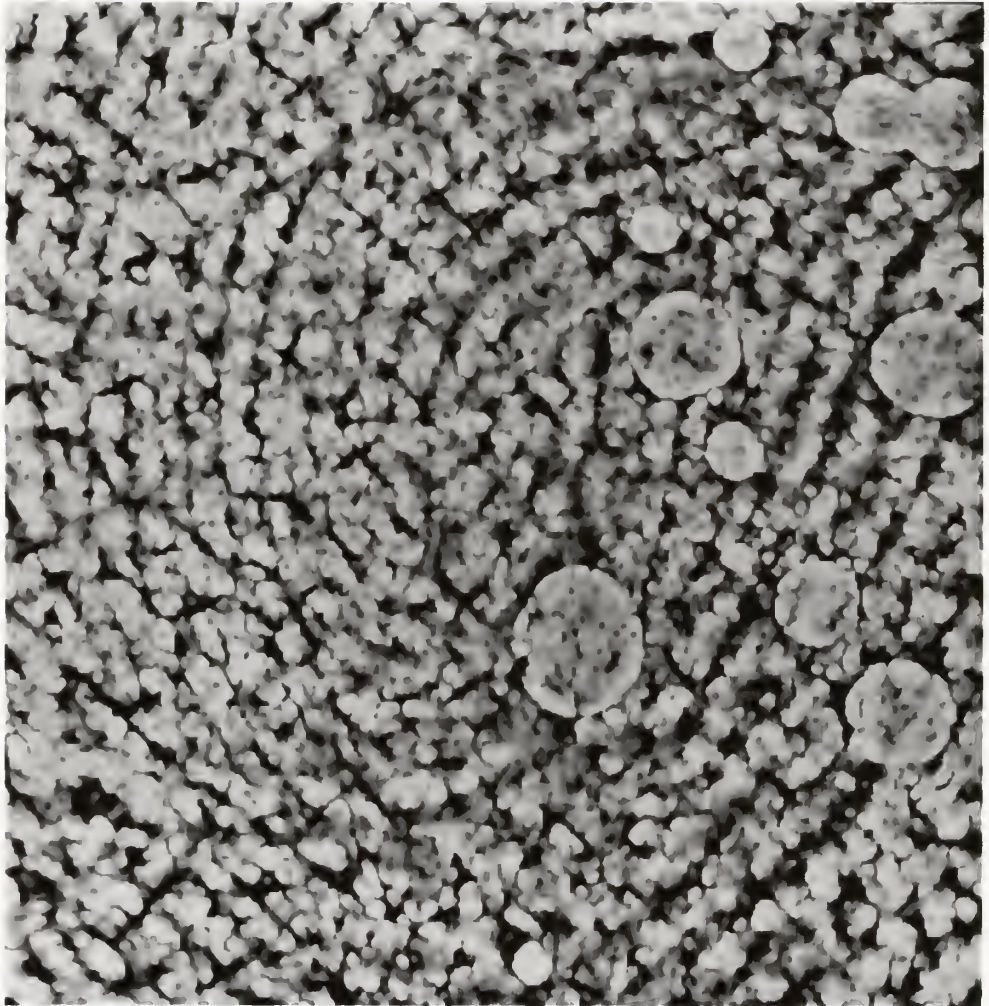


Figure 5.2-21 Result of edge enhancement through gradient thresholding applied to image in Figure 5.2-13

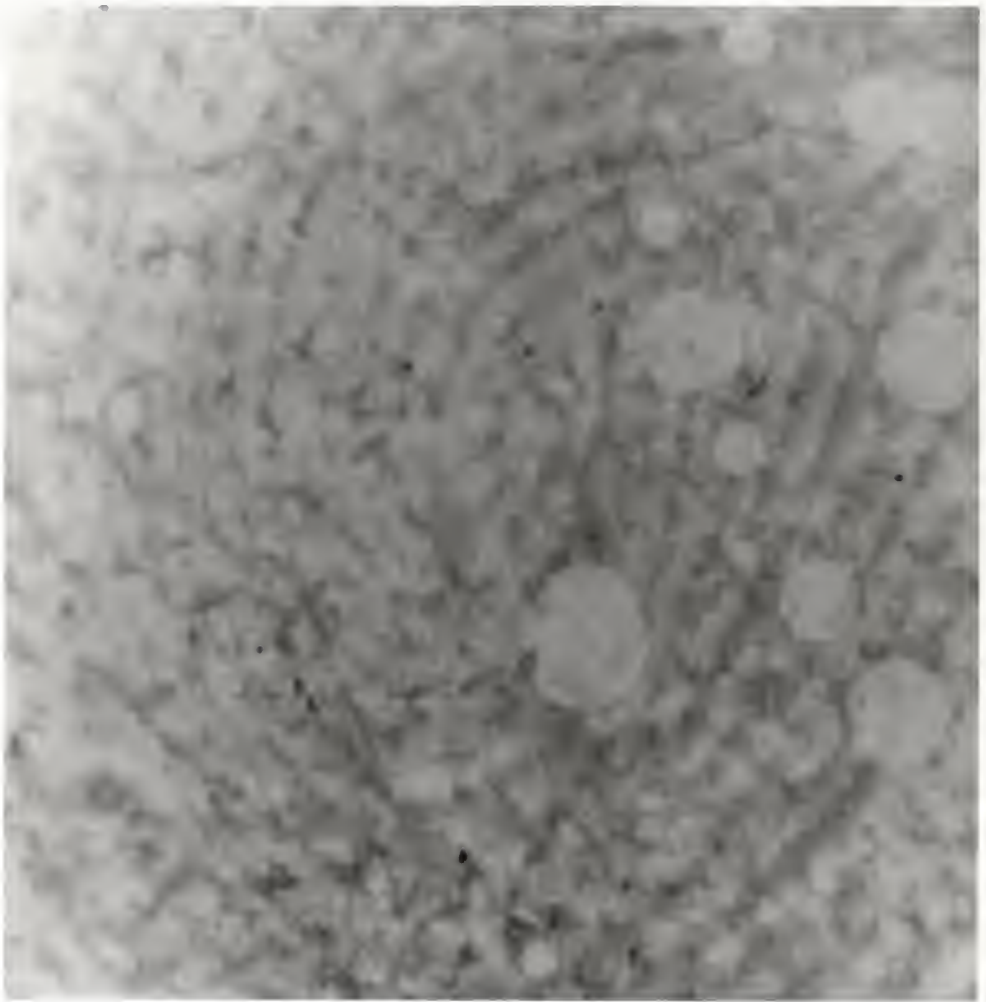


Figure 5.3-1 Result of smoothing through neighborhood averaging applied to image in Figure 5.1-1

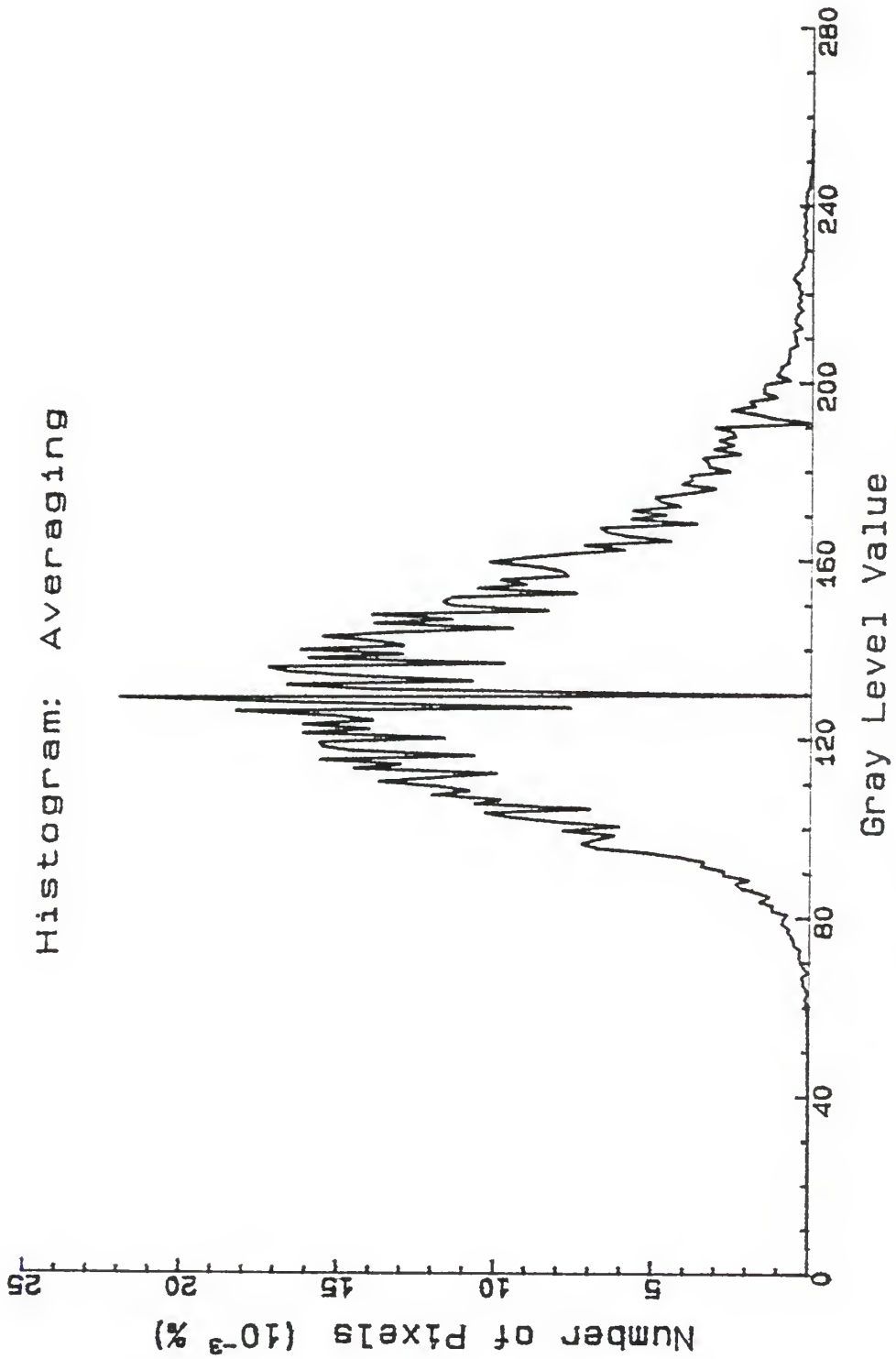


Figure 5.3-2 Histogram of image in Figure 5.3-1

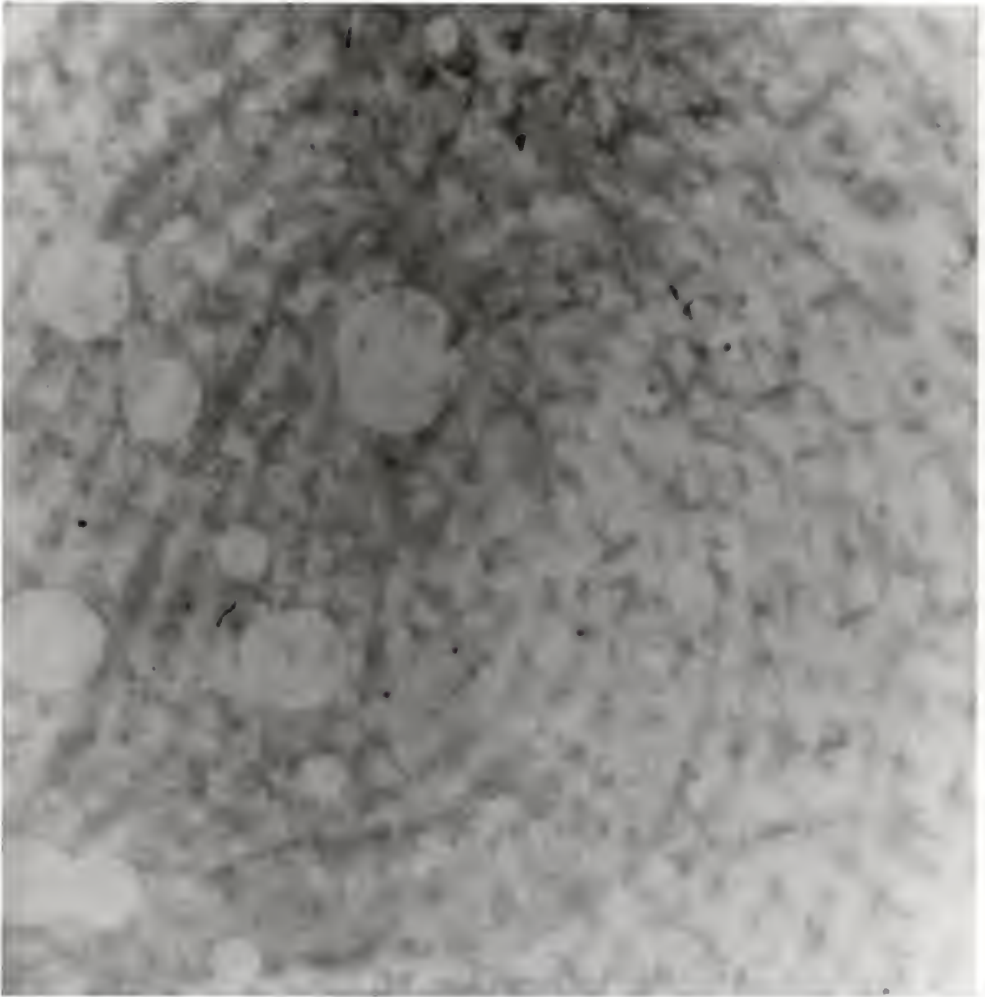


Figure 5.3-3 Result of 3×3 filtering applied to image
in Figure 5.1-1

3 X 3 Median Filter

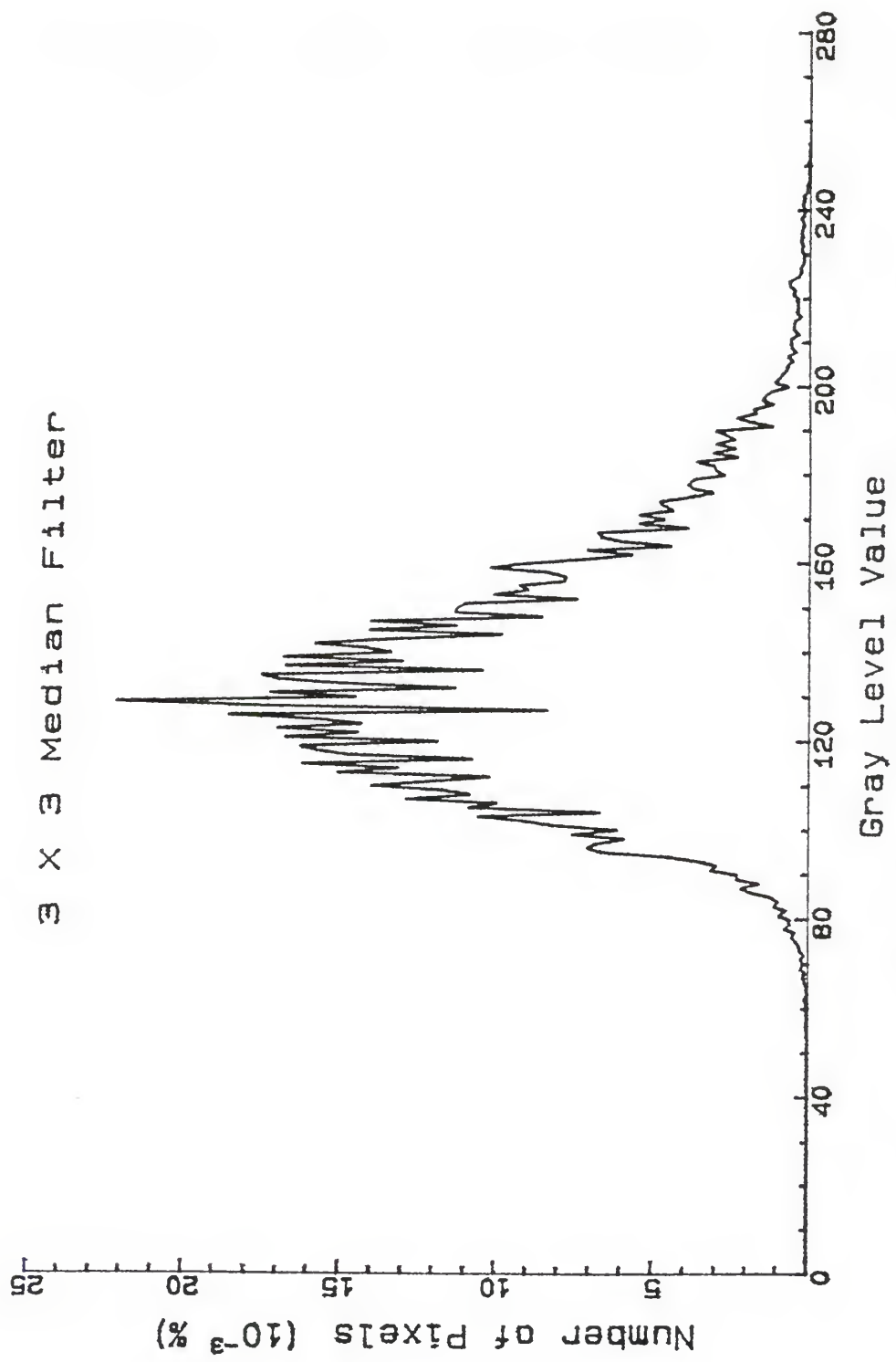


Figure 5.3-4 Histogram of image in Figure 5.3-3

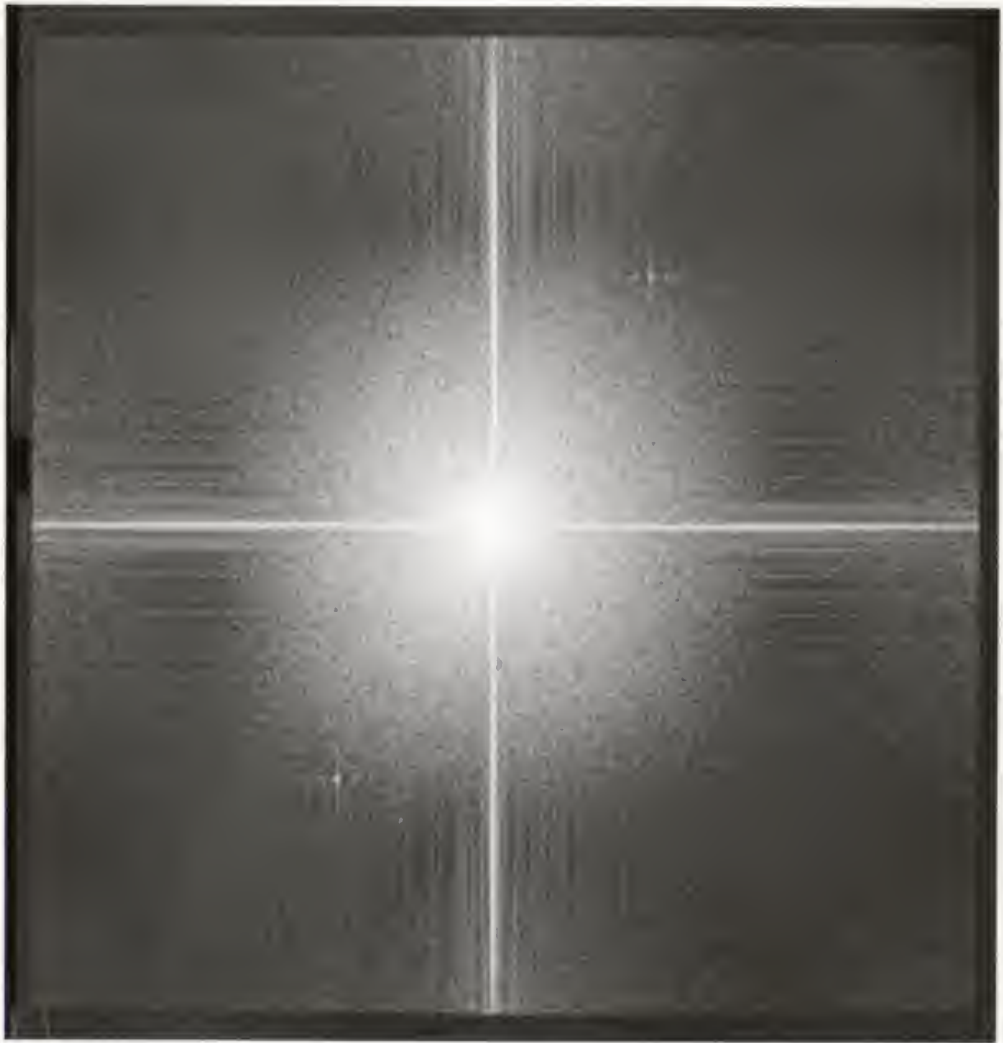


Figure 5.3-5 Result of transforming image in Figure 5.3-1 to the frequency domain and displaying using Equation (5.2-3)

Exponential Filter n=1, c=8

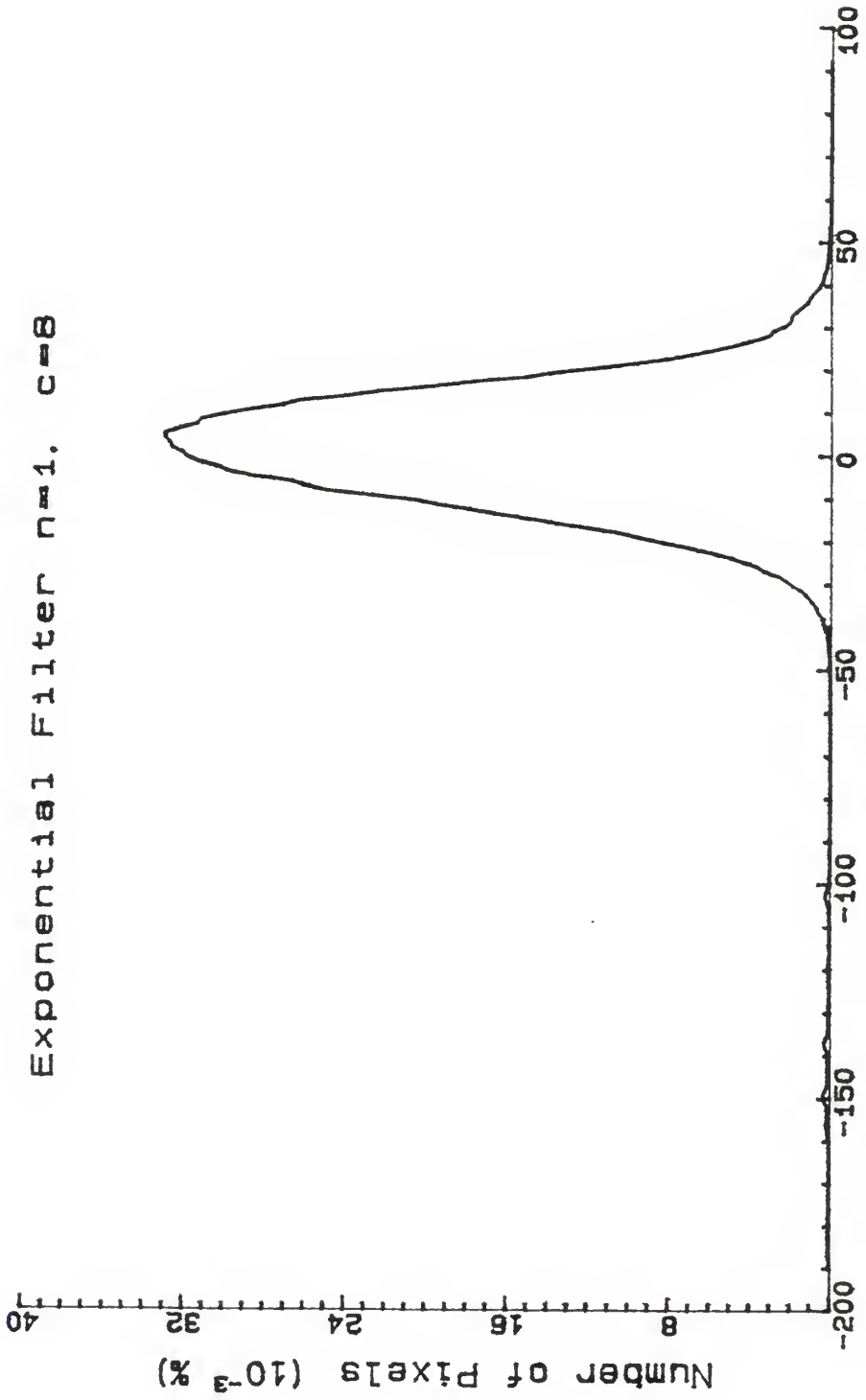


Figure 5.3-6 Histogram of filtered image before normalization

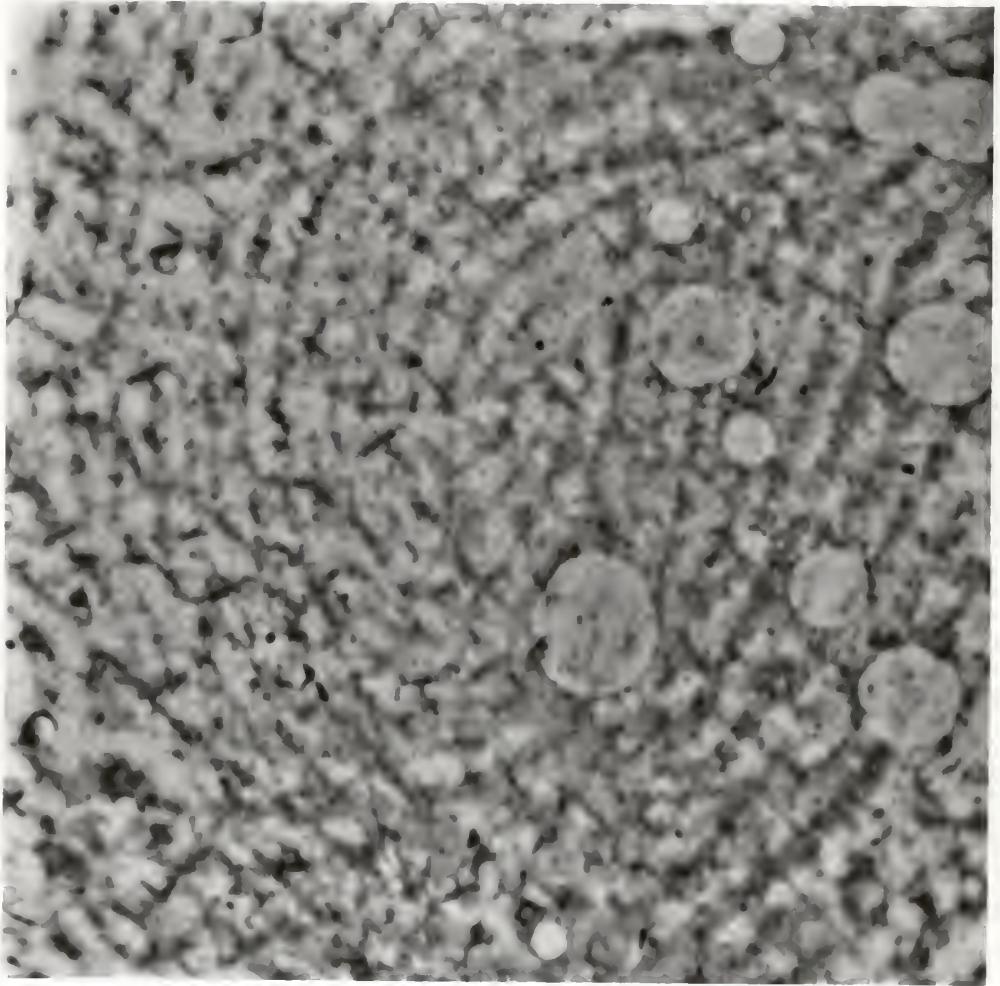


Figure 5.3-7 Result of highpass filtering and normalization applied to image in Figure 5.3-1

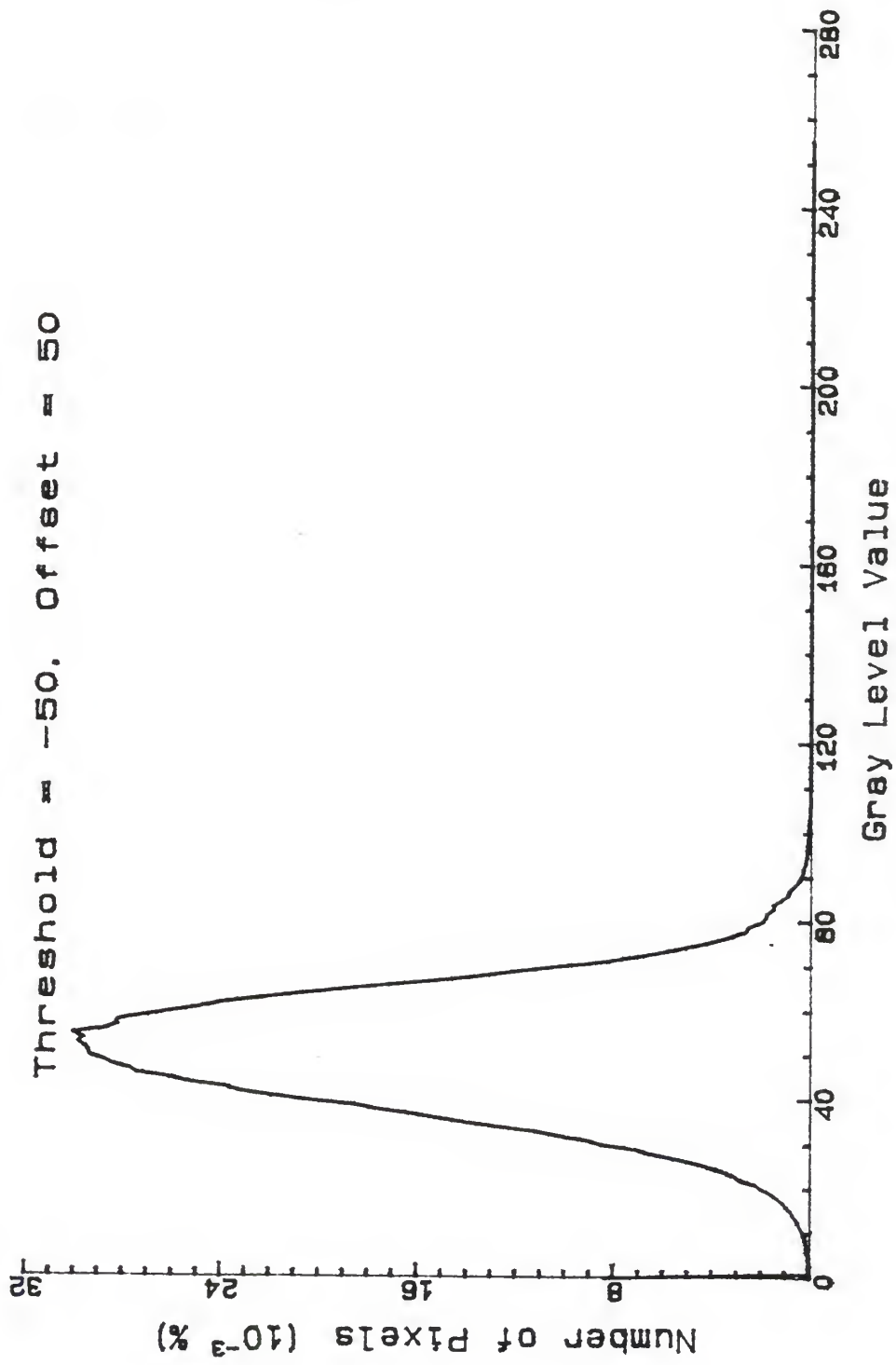


Figure 5.3-8 Histogram of image in Figure 5.3-7

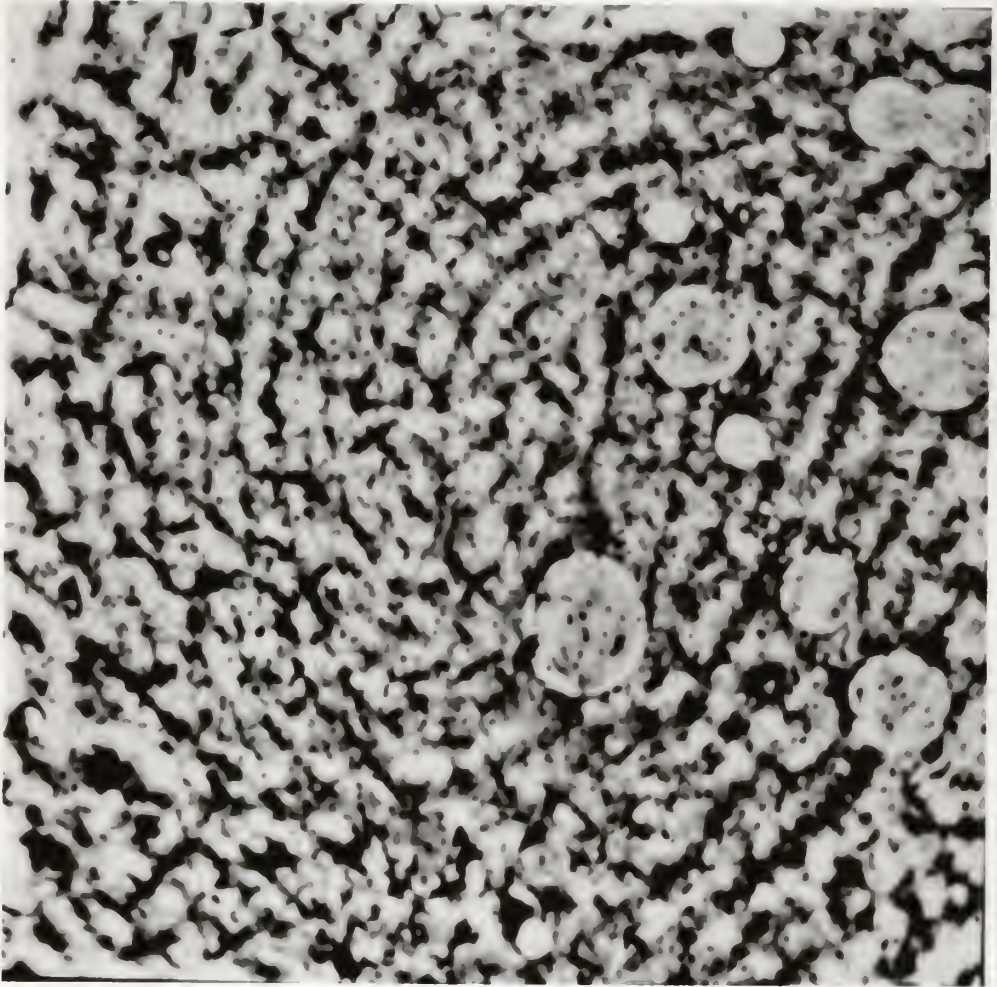


Figure 5.3-9 Result of histogram equalization applied to
image in Figure 5.3-7

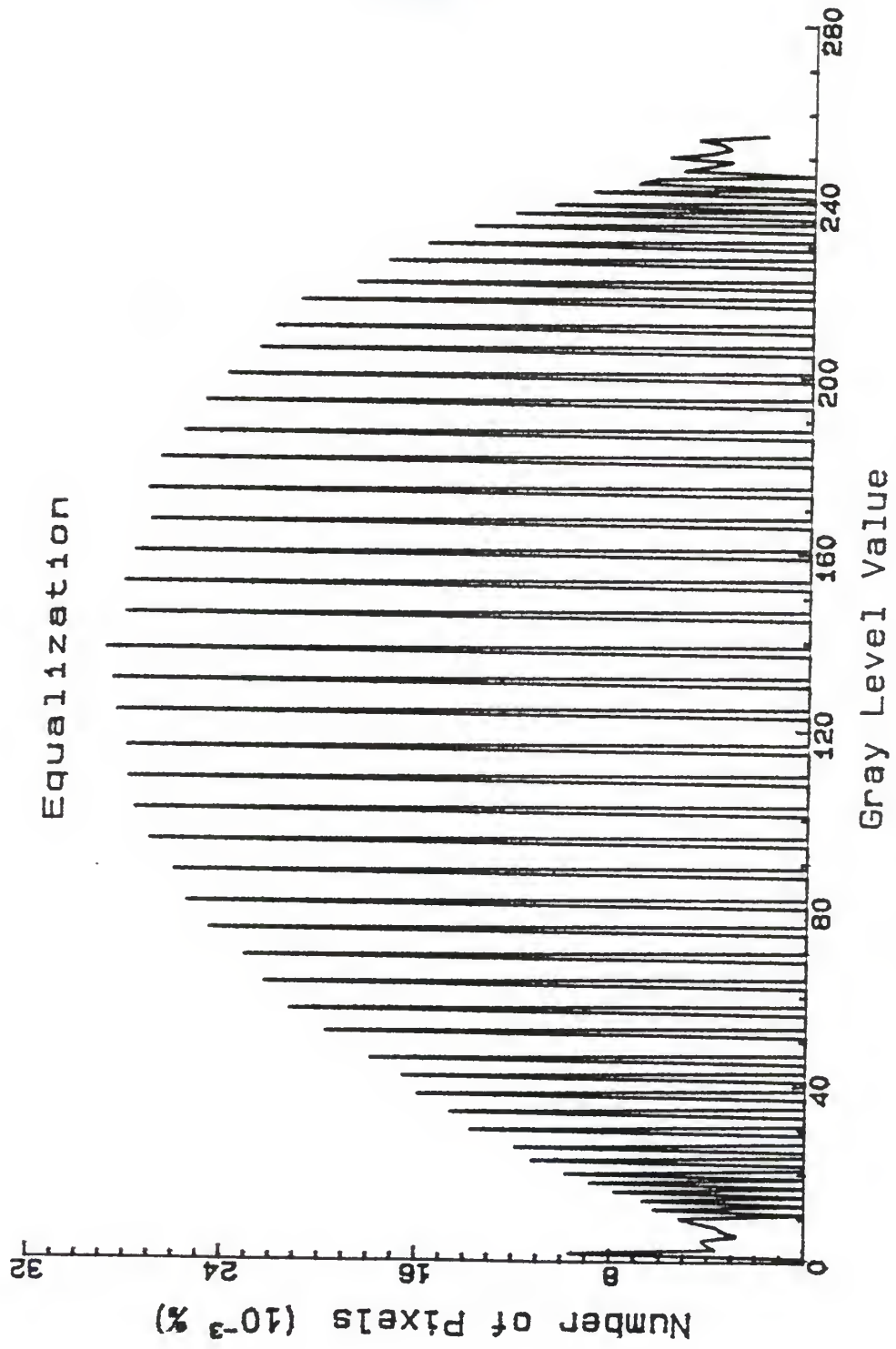


Figure 5.3-10 Histogram of image in Figure 5.3-9

CHAPTER 6: SUMMARY AND CONCLUSIONS

The objective of this project was to enhance degraded fingerprint images using digital image processing techniques. Improvement in image quality can be utilized by criminal investigators to identify unrecognizable fingerprints.

Prints for this project were recovered from human skin using the modified iodine-silver plate method. Considering that prints on human skin deteriorate after a short period of time, the recovery time of latent prints placed on human skin becomes critical. An increase in this recovery time is a consequence of the enhancement of degraded prints.

Ridges are the primary features to be enhanced in fingerprint images. The criteria for enhancement of ridge structure involves distinguishing individual ridges to the point where features, such as ridge endings and bifurcations, are recognizable. Ridges must be distinct from one another in order to precisely establish the relative positions of such features.

Combinations of spatial-domain and frequency-domain enhancement techniques were used to improve the print images.

Results show that frequency-domain techniques work well in enhancing ridge detail. Highpass filtering was quite successful at bringing out ridge detail from areas which were

either faint or obscured by degradation. Spatial-domain techniques are best suited to contrast enhancement. Histogram equalization and contrast stretching were both useful in sharpening the contrast between ridge and valley regions. Smoothing operations were utilized in order to remove noise and produce uniformity.

Two procedures were developed to enhance fingerprint images recovered through use of a modification of the iodine-silver plate method. Procedure One combines contrast stretching and highpass filtering in order to enhance ridge detail. Histogram equalization was added to improve contrast, followed by smoothing operations for additional uniformity in ridge areas. Edges were further enhanced through gradient thresholding. Procedure Two executes smoothing prior to highpass filtering. This is performed to remove noise and establish uniformity in the ridge areas for increased efficiency in the filtering operation. Filtering was followed by histogram equalization so that gray levels were redistributed throughout the normalized range for increased contrast. Both procedures showed significant enhancement of ridge detail. The most appreciable improvements occurred in areas where ridge detail was either faint or obscured by degradation.

One disadvantage of the procedures is the amount of trial

and error involved in choosing suitable parameters for the operations. It would prove beneficial to program the procedures so that parameters are chosen automatically. This could be accomplished, for instance, by calculating the cumulative distribution function from the probability density function to select proper threshold and cutoff values for spatial-domain techniques. Compass gradients could also be programmed to detect edges oriented in any direction with greater efficiency. In the frequency domain, the energy distribution provides insight into useful cutoff points. Automation of the procedures would allow technicians to implement the enhancement techniques in a practical environment. Further work along these lines would provide a useful application of this project.

CHAPTER 7: REFERENCES

1. Ahmed, N. and Natarajan, T. [1983]. Discrete Time Signals and Systems, Reston Publishing Co., Reston, Va.
2. Cooke, T.G. [1930]. Fingerprints, Secret Service, and Crime Detection, Fingerprint Publishing Co., Chicago.
3. Cummins, H. and Midlo, C. [1943]. Finger Prints, Palms, and Soles: An Introduction to Dermatoglyphics, The Blakiston Co., Philadelphia.
4. Eberlein, R.B. and Weszka, J.S. [1975]. "Mixtures of Derivative Operators as Edge Detectors." Computer Graphics and Image Processing, vol. 4, pp. 180-183
5. Feldman, M.A. [1981]. "New Applications of Chemistry to Criminalistics: Recovering Fingerprints from Skin [and] Evidence in Tire Slashings." Doctoral Dissertation, Kansas State University.
6. Gonzalez, R.C. and Wintz, P. [1987]. Digital Image Processing, Addison-Wesley Publishing Co., Reading, Ma.
7. Narendra, P.M. [1981]. "A Separable Median Filter for Image Noise Smoothing." I.E.E.E. Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-3, no. 1, pp. 20-29.
8. Olimb, H.E. [1986]. "Optical Enhancement of Degraded FingerPrints." Master's Thesis, Texas Tech University
9. Robinson, G.S. [1977]. "Edge Detection by Compass Gradient Masks." Computer Graphics and Image Processing, vol. 6, pp. 492-501

APPENDIX A: EQUIPMENT LIST

Digital Equipment Corporation VAX 11/750 digital computer

Grinnell GMR 270 Series High Performance Frame Buffers and Image Processing System

Mitsubishi Model #C3922 LPK High Resolution Color Television Monitor

Image Resource Videoprint 5000

Bencher, Incorporated Illuma System Quartz Light Table

Hewlett-Packard 7475A Plotter

Olympus 35 mm Camera

Ilford ASA 400 35 mm film

APPENDIX B: COMPUTER LISTINGS

Contained within this appendix are computer listings of some of the enhancement routines included in the programs COLOR and ENHANCE. The programs contain a considerable number of menu- and screen-drivers, which are omitted for the sake of brevity. Only routines which are relevant to the thesis are listed.

The routines require certain header files which contain variable and array declarations for the registers and working arrays. These header files are included in the routines at compile time. Most of the arrays are common-block arrays so that each routine affects the same memory locations.

The header files are listed first followed by the routines.

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: color.h
*
*****
*
*   ROUTINE:          color.h
*
*   DESCRIPTION:     This is a header file used to declare the
*                   common block of data which is used for
*                   integer*2 image files. This header
*                   declares the two-dimensional image array
*                   and the length of the rows and columns.
*
*   DOCUMENTATION
*   FILES:           None.
*
*   AUTHOR:          Ronald A. Gifford
*
*   DATE CREATED:    February 17, 1988
*
*   REVISIONS:
*
*****
*
*   integer*2 image, rd, sr, bl
*   dimension image(1:512,1:512), rd(1:512,1:512),
+   sr(1:512,1:512), bl(1:512,1:512)
*   integer*4 row, col, rrow, rcol, srow, scol,
+   brow, bcol
*
*   common image, rd, sr, bl, row, col, rrow, rcol, srow,
+   scol, brow, bcol
*
*   integer*4 MAX_PELS, MAX_LINES
*   parameter (MAX_PELS=512, MAX_LINES=512)

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: density.h
*
*****
*
*
*   ROUTINE:          density.h
*
*
*   DESCRIPTION:     This header defines the common block
*                   variables for the density slicing
*                   routines.
*
*
*   DOCUMENTATION
*   FILES:
*
*
*   AUTHOR:          Ronald A. Gifford
*
*
*   DATE CREATED:    February 26, 1988
*
*
*   REVISIONS:
*
*****
*
*
*   inteser*2 rlo, rhi, blo, bhi, slo, shi, ylo, yhi, mlo,
*   †           mhi, clo, chi, wlo, whi
*
*   common /density/ rlo, rhi, blo, bhi, slo, shi, ylo, yhi,
*   †           mlo, mhi, clo, chi, wlo, whi

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: hist.h
*
*****
*
*   ROUTINE:          hist
*
*   DESCRIPTION:     This is a header file which declares the
*                   memory used for the histogram parameters.
*
*   DOCUMENTATION
*   FILES:
*
*   AUTHOR:          Ronald A. Gifford
*
*   DATE CREATED:    September 26, 1987
*
*   REVISIONS:
*
*****
*
*   real*4 histogram, min, max, x_axis
*   dimension histogram(0:255), x_axis(0:255)
*
*   common /hist/ histogram, min, max, x_axis
*

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: mask.h
*
*****
*
*
*   ROUTINE:      mask.h
*
*
*   DESCRIPTION:  This is a header file for the spatial
*                 mask routines.
*
*
*   DOCUMENTATION
*   FILES:       None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: December 9, 1987
*
*
*   REVISIONS:
*
*****
*
*
integer*4 size
real*4 mask
dimension mask(1:7,1:7)
common /mask/mask, size

```

```

*****
*
*   Department of Electrical and Computer Engineering   *
*   Kansas State University                             *
*
*   VAX FORTRAN source filename:  math.h              *
*
*****
*
*   ROUTINE:      math.h
*
*   DESCRIPTION:  This is a header file which defines some
*                 commonly used mathematical parameters.
*
*   DOCUMENTATION
*   FILES:       None.
*
*   AUTHOR:      Ronald A. Gifford
*
*   DATE CREATED: August 23, 1987
*
*   REVISIONS:
*
*****
*
*   real*4 PI
*   parameter (PI = 3.1415926535897932)
*
*   real*8 DPI
*   parameter (DPI = 3.1415926535897932D00)

```

```

*****
*
*   Department of Electrical and Computer Engineering   *
*   Kansas State University                             *
*
*   VAX FORTRAN source filename: register.h           *
*
*****
*
*   ROUTINE:      register.h
*
*   DESCRIPTION:  This is a header file used to declare the
*                 common block of data which is used for
*                 complex image files. This header declares
*                 the two-dimensional image array and the
*                 length of the rows and columns.
*
*   DOCUMENTATION
*   FILES:        None.
*
*   AUTHOR:       Ronald A. Gifford
*
*   DATE CREATED: August 25, 1987
*
*   REVISIONS:
*
*****
*
*   complex*8 image, Y, storage, storage2
*   dimension image(1:512,1:512), Y(1:512,1:512),
+       storage(1:512,1:512), storage2(1:512,1:512)
*   integer*4 rows, columns, Yrows, Ycols, sto1rows, sto1cols,
+       sto2rows, sto2cols
*
*   common image, Y, storage, storage2, rows, columns, Yrows,
+       Ycols, sto1rows, sto1cols, sto2rows, sto2cols
*
*   integer*4 MAX_PELS, MAX_LINES
*   parameter (MAX_PELS=512, MAX_LINES=512)

```



```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: Butterworth_highpass.for
*
*****
*
*
*   ROUTINE:          subroutine Butterworth_highpass(cutoff,
*                   order,emphasis)
*
*
*   DESCRIPTION:     This routine filters a two-dimensional
*                   image array using an nth-order Butterworth
*                   highpass filter.
*
*
*   DOCUMENTATION
*   FILES:          None.
*
*
*   ARGUMENTS:      cutoff          real*4
*                   Cutoff frequency.
*
*                   order          real*4
*                   Order of the filter.
*
*                   emphasis       real*4
*                   High-frequency emphasis constant
*
*
*   INCLUDE:        resistor.h
*
*
*   RETURN:         image()          Filtered image
*
*
*   ROUTINES
*   CALLED:         None.
*
*
*   AUTHOR:         Ronald A. Gifford
*
*
*   DATE CREATED:   September 7, 1987
*
*
*   REVISIONS:
*
*****

```

```

*
      subroutine Butterworth_highpass(cutoff,order,emphasis)
*
*****
*   Variable declarations   *
*****
*
      implicit none
*
      include '[GIFFORD.HEADS]resister.h'
*
      real*4  cutoff, D, D0, H, emphasis
*
      integer*4 i, j, order, N2, arsl
*
*****
*   Set emphasis factor     *
*****
*
      if (flag.EQ. 0) then
*
          emphasis = 1.0
*
      else
*
          emphasis = 0.0
*
      endif
*
*****
*   Filter the image       *
*****
*
      D0 = sqrt(2.0*(cutoff**2))
      N2 = columns/2
      arsl = N2 - 1
      do j = 1, arsl
          do i = 1, rows
              D = sqrt(1.0*((i-N2)**2 + (j-N2)**2))
              H = 1.0/(1.0 + 0.414*((D0/D)**(2*order)))
              H = emphasis + H
              imase(i,j) = cmplx(H,0.0)*imase(i,j)
          enddo
      enddo
*
      do i = 1, arsl
          D = sqrt(1.0*((i-N2)**2))
          H = 1.0/(1.0 + 0.414*((D0/D)**(2*order)))
          H = emphasis + H
          imase(i,N2) = H*imase(i,N2)
      enddo
      imase(N2,N2) = emphasis*imase(N2,N2)
      arsl = arsl + 2
      do i = arsl, rows
          D = sqrt(1.0*((i-N2)**2))
          H = 1.0/(1.0 + 0.414*((D0/D)**(2*order)))
          H = emphasis + H
          imase(i,N2) = H*imase(i,N2)
      enddo

```

```

    enddo
*
do j = 1, columns
    do i = 1, rows
        D = sqrt(1.0*((i-N2)**2 + (j-N2)**2))
        H = 1.0/(1.0 + 0.414*((D0/D)**(2*order)))
        H = emphasis + H
        image(i,j) = H*image(i,j)
    enddo
enddo
*
return
end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: bright_trans.for
*
*****
*
*
*   ROUTINE:      subroutine bright_trans(cutoff,value)
*
*
*   DESCRIPTION:  This routine filters an image file with
*                 the cutoff point corresponding to a
*                 brightness level. The resulting image
*                 turns all gray levels greater than the
*                 cutoff to the transformation value.
*
*
*   DOCUMENTATION
*   FILES:       None.
*
*
*   ARGUMENTS:   cutoff      real*4
*                 The desired cutoff point for the
*                 filter routine.
*
*                 value      real*4
*                 The transformation value.
*
*   INCLUDE:     register.h
*
*
*   RETURN:      image()     Transformed image.
*
*
*   ROUTINES
*   CALLED:      None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: July 7, 1987
*
*
*   REVISIONS:
*
*****
*
*   subroutine bright_trans(cutoff,value)

```

```

*
*
*****
*      Variable declarations.      *
*****
*
*      implicit none
*
*      include '[GIFFORD.HEADS]resister.h'
*
*      integer*4 i, j
*
*      real*4 cutoff, value
*
*      do j = 1, columns
*          do i = 1, rows
*              if (real(imase(i,j)) .GE. cutoff) then
*                  imase(i,j) = cplx(value,0.0)
*              endif
*          enddo
*      enddo
*
*      return
*
*      end

```

```

*****
*
*   Department of Electrical and Computer Engineering   *
*   Kansas State University                             *
*
*   VAX FORTRAN source filename: dark_trans.for       *
*
*****
*
*
*   ROUTINE:      subroutine dark_trans(cutoff)
*
*
*   DESCRIPTION:  This routine filters an image file with
*                 the cutoff point corresponding to a
*                 brightness level. The resulting image
*                 turns the gray levels below the cutoff to
*                 the transformation value.
*
*
*   DOCUMENTATION
*   FILES:       None.
*
*
*   ARGUMENTS:   cutoff          real*4
*                 The desired cutoff point for the
*                 filter routine.
*
*                 value          real*4
*                 The transformation value.
*
*
*   INCLUDE:     register.h
*
*
*   RETURN:      image()         Transformed image.
*
*
*   ROUTINES
*   CALLED:      None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: July 7, 1987
*
*
*   REVISIONS:
*
*****
*
*

```

```

      subroutine dark_trans(cutoff,value)
*
*
*****
*   Variable declarations.   *
*****
*
*
      implicit none
*
      include '[GDIFFORD,HEADS]resister.h'
*
      integer*4 i, j
*
      real*4 cutoff, value
*
      do j = 1, columns
        do i = 1, rows
          if (real(imase(i,j)) .LT. cutoff) then
            imase(i,j) = cmplx(value,0.0)
          endif
        enddo
      enddo
*
      return
*
      end

```



```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: dft2D.for
*
*****
*
*
*   ROUTINE:      subroutine dft2D(flag)
*
*
*   DESCRIPTION:  This routine computes the DFT for a
*                 complex input sequence. The input
*                 sequence must be a two-dimensional square
*                 array with row and column lengths which
*                 are a power of two.
*
*
*   DOCUMENTATION
*   FILES:       None.
*
*
*   ARGUMENTS:   flag    INTEGER*2
*                 Flag to determine transform direction.
*                 0      Forward
*                 1      Inverse
*
*
*   INCLUDE:     resistor.h
*                 math.h
*
*
*   RETURN:      imase()      Transformed imase.
*
*
*   ROUTINES
*   CALLED:      None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: August 17, 1987
*
*
*   REVISIONS:
*
*****
*
*
*   subroutine dft2D(flag)

```

```

*
*****
*   Variable declarations   *
*****
*
    implicit none
*
    include '[GIFFORD.HEADS]resister.h'
    include '[GIFFORD.HEADS]math.h'
*
    INTEGER*2 flas
*
    INTEGER*4 i, j, k, iter, n, n2, m, m_N2, x, z
*
    REAL*4  sign, arg, mult
*
    COMPLEX*8 W, Temp, work
    dimension work(1:512)
*
*****
*   Calculate the number of iterations.   *
*****
*
    iter = 0
    n = rows/2
    do while (n .GE. 1)
        n = n/2
        iter = iter + 1
    enddo
*
*****
*   Check for DFT direction.   *
*****
*
    if (flas .EQ. 0) then
        sign = -1.0
    else
        sign = 1.0
    endif
*
*****
*   Column transform.   *
*****
*
    do x = 1, columns
*
*****
*   Resin iteration.   *
*****
*
        n2 = rows
*
        do i = 1, iter
            n = n2

```

```

        n2 = n/2
        mult = PI/(1.0*n2)
*
*****
*       Calculate the multiplier.       *
*****
*
        do j = 1, n2
            arg = (j-1)*mult
            W = cmplx(cos(arg), sign*sin(arg))
*
*****
*       Calculate partitions.         *
*****
*
            do k = n, rows, n
                m = k - n + j
                m_N2 = m + n2
                Temp = imase(m,x) - imase(m_N2,x)
                imase(m,x) = imase(m,x) + imase(m_N2,x)
                imase(m_N2,x) = W*Temp
            enddo
        enddo
    enddo
*
*****
*       Bit reversal.                 *
*****
*
        n2 = rows/2
        j = 1
        n = rows - 1
        do i = 1, n
            if (i .LT. j) then
                Temp = imase(j,x)
                imase(j,x) = imase(i,x)
                imase(i,x) = Temp
            endif
            k = n2
            do while (k .LT. j)
                j = j - k
                k = k/2
            enddo
            j = j + k
        enddo
    enddo
*
*****
*       Row transform.                 *
*****
*
        do x = 1, rows
            do z = 1, columns
                work(z) = imase(x,z)

```

```

        enddo
*
*****
*      Begin iteration,      *
*****
*
        n2 = columns
*
        do i = 1, iter
            n = n2
            n2 = n/2
            mult = PI/(1.0*n2)
*
*****
*      Calculate the multiplier,      *
*****
*
            do j = 1, n2
                arg = (j-1)*mult
                W = cmplx(cos(arg),sign*sin(arg))
*
*****
*      Calculate partitions,      *
*****
*
                do k = n, columns, n
                    m = k - n + j
                    m_N2 = m + n2
                    TEMP = work(m) - work(m_N2)
                    work(m) = work(m) + work(m_N2)
                    work(m_N2) = W*TEMP
                enddo
            enddo
        enddo
*
*****
*      Bit reversal,      *
*****
*
        n2 = columns/2
        j = 1
        n = columns - 1
        do i = 1, n
            if (i .LT. j) then
                TEMP = work(j)
                work(j) = work(i)
                work(i) = TEMP
            endif
            k = n2
            do while (k .LT. j)
                j = j - k
                k = k/2
            enddo
            j = j + k

```

```

        enddo
*
        do z = 1, columns
            image(x,z) = work(z)
        enddo
    enddo
*
*****
*   Scaling   *
*****
*
    Temp = cmplx(1.0/(1.0*rows),0.0)
    do j = 1, columns
        do i = 1, rows
            image(i,j) = image(i,j)*Temp
        enddo
    enddo
*
*
    return
end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: energy.for
*
*****
*
*
*   ROUTINE:      function energy(radius)
*
*
*   DESCRIPTION:  This routine calculates the energy
*                 distribution for a specific radius in the
*                 frequency domain.
*
*
*   DOCUMENTATION
*   FILES:
*
*
*   ARGUMENTS:   radius  real*4
*                Specified radius
*
*
*   INCLUDE:     resistor.h
*
*
*   RETURN:      real*4  Percentase of energy contained
*                    within radius.
*
*
*   ROUTINES
*   CALLED:      None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: February 28, 1988
*
*
*   REVISIONS:
*
*****
*
*
*   real*4 function energy(radius)
*
*****
*   Variable declarations *
*****

```

```

*
  implicit none
*
  include 'GIFFORD.HEADS]resister.h'
*
  integer*4 i, j
*
  real*4 radius, total, temp, r0
*
*****
*   Compute energy distribution   *
*****
*
  energy = 0.0
  total = 0.0
  do j = 1, columns
    do i = 1, rows
      temp = cabs(image(i,j))
      total = total + temp
      r0 = sqrt(real((256-i)**2 + (256-j)**2))
      if (r0 .LE. radius) then
        energy = energy + temp
      endif
    enddo
  enddo
  energy = (energy/total)*100.0
  return
end

```



```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: eau.for
*
*****
*
*   ROUTINE:          function eau()
*
*   DESCRIPTION:     This routine manipulates a 512 x 512 image
*                   file by equalizing according to the
*                   histogram of the original image.
*
*   DOCUMENTATION
*   FILES:          None.
*
*   ARGUMENTS:     None.
*
*   RETURN:        integer*2      Error flag
*                   0              SUCCESS
*                   1              Array not normalized
*
*                   image()        Equalized image.
*
*   INCLUDE:       register.h
*                   hist.h
*
*   ROUTINES
*   CALLED:        histnorm()
*
*   AUTHOR:        Ronald A. Gifford
*
*   DATE CREATED:  January 6, 1988
*
*   REVISIONS:
*
*****
*
*   integer*2 function eau()
*
*****

```

```

*      Variable declaration      *
*****
*
*      implicit none
*
*      include 'GIFFORD.HEADS]resister.h'
*      include 'GIFFORD.HEADS]hist.h'
*
*      real*4 ea(0:255), px
*
*      integer*2 err, histnorm
*      integer*4 i, j, gray
*
*****
*      Compute histogram      *
*****
*
*      err = histnorm()
*      if (err .EQ. 1) then
*          eau = 1
*          return
*      else
*          eau = 0
*      endif
*
*****
*      Compute uniform distribution      *
*****
*
*      px = 0.0
*      do i = 0, 255
*          px = px + histogram(i)
*          ea(i) = 255*px
*      enddo
*
*****
*      Equalize image array      *
*****
*
*      do j = 1, columns
*          do i = 1, rows
*              gray = nint(real(image(i,j)))
*              image(i,j) = cmplx(ea(gray),0.0)
*          enddo
*      enddo
*
*      return
*
*      end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: exponential_high.for
*
*****
*
*
*   ROUTINE:      subroutine exponential_high(cutoff,
*                order,emphasis)
*
*
*   DESCRIPTION:  This routine filters a two-dimensional
*                 image array using an nth-order exponential
*                 highpass filter.
*
*
*   DOCUMENTATION
*   FILES:       None.
*
*
*   ARGUMENTS:   cutoff          real*4
*                Cutoff frequency.
*
*                order          real*4
*                Order of the filter.
*
*                emphasis       real*4
*                High-frequency emphasis constant
*
*
*   INCLUDE:     register.h
*
*
*   RETURN:      image()         Filtered image
*
*
*   ROUTINES
*   CALLED:     None
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: September 7, 1987
*
*
*   REVISIONS:
*
*****
*

```

```

*
      subroutine exponential_high(cutoff,order,emphasis)
*
*****
*      Variable declarations      *
*****
*
      implicit none
*
      include 'IGIFFORD.HEADS]resister.h'
*
      real*4  cutoff, D, D0, H, emphasis
*
      integer*2 flag
*
      integer*4 i, j, order, N2, arsl
*
*****
*      Set emphasis factor      *
*****
*
      if (flag .EQ. 0) then
*
          emphasis = 1.0
*
      else
*
          emphasis = 0.0
*
      endif
*
*****
*      Filter the image      *
*****
*
      D0 = sqrt(2.0*(cutoff**2))
      N2 = columns/2
      arsl = N2 - 1
      do j = 1, arsl
          do i = 1, rows
              D = sqrt(1.0*((i-N2)**2 + (j-N2)**2))
              H = exp(-0.347*((D0/D)**(2*order)))
              H = emphasis + H
              image(i,j) = cmplx(H,0.0)*image(i,j)
          enddo
      enddo
*
      do i = 1, arsl
          D = sqrt(1.0*((i-N2)**2))
          H = exp(-0.347*((D0/D)**(2*order)))
          H = emphasis + H
          image(i,N2) = H*image(i,N2)
      enddo
      image(N2,N2) = emphasis*image(N2,N2)
      arsl = arsl + 2
      do i = arsl, rows
          D = sqrt(1.0*((i-N2)**2))
          H = exp(-0.347*((D0/D)**(2*order)))

```

```

        H = emphasis + H
        imase(i,N2) = H*imase(i,N2)
    enddo
*
do j = arg1, columns
    do i = 1, rows
        D = sqrt(1.0*((i-N2)**2 + (j-N2)**2))
        H = exp(-0.347*((D0/D)**(2*order)))
        H = emphasis + H
        imase(i,j) = H*imase(i,j)
    enddo
enddo
*
return
end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: histabs.for
*
*****
*
*
*   ROUTINE:          subroutine histabs()
*
*
*   DESCRIPTION:      This routine calculates the histogram for
*                     an image file. The range of values is
*                     divided into 256 bins.
*
*
*   DOCUMENTATION
*   FILES:           None.
*
*
*   ARGUMENTS:       None.
*
*
*   INCLUDE:         resistor.h
*                   hist.h
*
*
*   RETURN:          histogram()    Computed histogram.
*                   x_axis()       Gray level values.
*                   min and max    Minimum and maximum gray
*                                   level values.
*
*
*   ROUTINES
*   CALLED:          None.
*
*
*   AUTHOR:          Ronald A. Gifford
*
*
*   DATE CREATED:    January 28, 1988
*
*
*   REVISIONS:
*
*****
*
*   subroutine histabs()
*

```

```

*
*****
*      Variable declarations      *
*****
*
*      implicit none
*
*      include '[GIFORD.HEADS]register.h'
*      include '[GIFORD.HEADS]hist.h'
*
*      integer*4 i, j, k
*
*      real*4 total, bin
*      total = real(columns*rows)
*
*
*****
*      Zero histogram array      *
*****
*
*      do i = 0, 255
*          histogram(i) = 0.0
*      enddo
*
*****
*      Calculate bin width      *
*****
*
*      call range(min,max)
*      bin = (max-min)/250.0
*      x_axis(0) = min
*      do i = 1, 255
*          j = i - 1
*          x_axis(i) = x_axis(j) + bin
*      enddo
*
*****
*      Compute histogram      *
*****
*
*      do j = 1, columns
*          do i = 1, rows
*              k = 0
*              do while (x_axis(k) .LT. real(image(i,j)))
*                  k = k + 1
*              enddo
*              histogram(k) = histogram(k) + 1.0
*          enddo
*      enddo
*
*      do i = 0, 255
*          histogram(i) = histogram(i)/total

```



```
*      enddo  
      return  
end
```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: histnorm.for
*
*****
*
*
*   ROUTINE:          function histnorm()
*
*
*   DESCRIPTION:      This routine calculates the histogram for
*                     an image file. The image file is assumed
*                     to be normalized between the values 0 and
*                     255.
*
*
*   DOCUMENTATION
*   FILES:           None.
*
*
*   ARGUMENTS:       None.
*
*
*   INCLUDE:         register.h
*                   hist.h
*
*
*   RETURN:          integer*2          Error flag which
*                                     denotes acceptable range
*                                     of values.
*                                     0          Gray values within range
*                                     1          Gray values out of range.
*
*                   histogram()       Computed histogram.
*
*                   x_axis()         Gray level values.
*
*                   min and max       Minimum and maximum
*                                     gray level values.
*
*
*   ROUTINES
*   CALLED:
*
*
*   AUTHOR:          Ronald A. Gifford
*
*
*   DATE CREATED:   January 28, 1988
*
*
*   REVISIONS:

```

```

*
*
*****
*
*
      inteser*2 function histnorm()
*
*
*****
*   Variable declarations   *
*****
*
*
      implicit none
*
*
      include '[GIFFORD,HEADS]register.h'
      include '[GIFFORD,HEADS]hist.h'
*
*
      inteser*4 i, j, k
*
*
      real*4 total
      total = 1.0*(columns*rows)
*
*****
*   Test for normalization *
*****
*
      call range(min,max)
      if (min .LT. 0.0 .OR. max .GT. 255.0) then
          histnorm = 1
          return
      else
          histnorm = 0
      endif
*
*****
*   Scale x_axis   *
*****
*
      do i = 0, 255
          x_axis(i) = real(i)
      enddo
*
*
*****
*   Calculate histosram *
*****
*
*
      do i = 0, 255
          histosram(i) = 0.0
      enddo
*
*
      do j = 1, columns

```



```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: linear_stretch.for
*
*****
*
*   ROUTINE:      subroutine linear_stretch(low,high)
*
*
*   DESCRIPTION:  This routine performs a contrast stretch
*                 using a linear transformation.
*
*
*   DOCUMENTATION
*   FILES:        None.
*
*
*   ARGUMENTS:    low          real*4
*                 Low cutoff point.
*
*                 high         real*4
*                 High cutoff point.
*
*
*   INCLUDE:      register.h
*
*
*   RETURN:        image()      Transformed image.
*
*
*   ROUTINES
*   CALLED:        None.
*
*
*   AUTHOR:        Ronald A. Gifford
*
*
*   DATE CREATED:  October 4, 1987
*
*
*   REVISIONS:
*
*****
*
*   subroutine linear_stretch(low,high)
*
*****
*   Variable declaration *

```

```

*****
*
*      implicit none
*
*      include '[GIFFORD.HEADS]resister.h'
*
*      integer*4 i, j
*
*      real*4 low, high, m, b, temp
*
*****
*      Calculate slope and intercept *
*****
*
*      m = 255.0/(high-low)
*      b = -m*low
*
*****
*      Perform transformation *
*****
*
*      do j = 1, columns
*          do i = 1, rows
*              if (real(image(i,j)) .LT. low) then
*                  image(i,j) = cmplx(0.0,0.0)
*              else if (real(image(i,j)) .GT. high) then
*                  image(i,j) = cmplx(255.0,0.0)
*              else
*                  temp = m*real(image(i,j)) + b
*                  image(i,j) = cmplx(temp,0.0)
*              endif
*          enddo
*      enddo
*
*      return
*      end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: mask_compute.for
*
*****
*
*
*   ROUTINE:      subroutine mask_compute()
*
*
*   DESCRIPTION:  This routine computes the mask of a
*                 real-valued image. The mask size is
*                 given by the variable 'size' found in
*                 mask.h. The image size is given by the
*                 variables 'rows' and 'columns' found in
*                 register.h.
*
*
*   DOCUMENTATION
*   FILES:      None.
*
*
*   ARGUMENTS:  None.
*
*
*   INCLUDE:    register.h
*               mask.h
*
*
*   RETURN:     image()      Image after mask computation.
*
*
*   ROUTINES
*   CALLED:    c_switch()
*
*
*   AUTHOR:     Ronald A. Gifford
*
*
*   DATE CREATED:  December 9, 1987
*
*
*   REVISIONS:
*
*****
*
*   subroutine mask_compute()
*
*****

```



```

*      Variable declarations      *
*****
*
*      implicit none
*
*      include 'GIFORD.HEADS]register.h'
*      include 'GIFORD.HEADS]mask.h'
*
*      integer*4 i, j, k, l, ik, jl, edse, lower, upper
*
*      real*4 sum, temp
*
*****
*      Compute bounds      *
*****
*
*      edse = (size-1)/2
*      lower = edse + 1
*      upper = rows - edse
*
*****
*      Compute the mask and return value to the imaginary part.*
*****
*
*      do j = lower, upper
*          do i = lower, upper
*              sum = 0.0
*              jl = j - edse
*              do l = 1, size
*                  ik = i - edse
*                  do k = 1, size
*                      sum = sum + real(image(ik,jl))*mask(k,l)
*                      ik = ik + 1
*                  enddo
*                  jl = jl + 1
*              enddo
*              image(i,j) = cmplx(real(image(i,j)),sum)
*          enddo
*      enddo
*
*****
*      Switch real and imaginary      *
*****
*
*      call c_switch()
*
*      return
*
*      end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: mean_var.for
*
*****
*
*   ROUTINE:      subroutine mean_var(mean,var)
*
*
*   DESCRIPTION:  This routine calculates the mean and
*                 and variance for a histogram of the real
*                 parts of an image array.
*
*
*   DOCUMENTATION
*   FILES:
*
*
*   ARGUMENTS:   mean    real*4
*                 Computed mean returned by routine.
*
*                 var    real*4
*                 Computed variance returned by routine.
*
*   INCLUDE:     register.h
*                 hist.h
*
*   RETURN:      None.
*
*
*   ROUTINES
*   CALLED:      None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: December 8, 1987
*
*
*   REVISIONS:
*
*****
*
*   subroutine mean_var(mean,var)
*
*****
*   Variable declarations *

```

```

*****
*
*      implicit none
*
*      include 'GIFFORD,HEADS]hist.h'
*
*      integer*4 i
*
*      real*4 mean, var
*
*****
*      Calculate mean and variance.      *
*****
*
*      mean = 0.0
*      do i = 0, 255
*          mean = mean + x_axis(i)*histogram(i)
*      enddo
*
*      var = 0.0
*      do i = 0, 255
*          var = var + ((x_axis(i)-mean)**2)*histogram(i)
*      enddo
*
*      return
*
*      end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: median.for
*
*****
*
*
*   ROUTINE:          subroutine median(size)
*
*
*   DESCRIPTION:      This routine performs median filtering
*                     on two-dimensional image arrays.
*
*
*   DOCUMENTATION
*   FILES:           None.
*
*
*   ARGUMENTS:       size   integer*4
*                     Mask size.
*
*
*   INCLUDE:         resistor.h
*
*
*   RETURN:          image()   Filtered image.
*
*
*   ROUTINES
*   CALLED:          c_switch()
*
*
*   AUTHOR:          Ronald A. Gifford
*
*
*   DATE CREATED:    February 28, 1988
*
*
*   REVISIONS:
*
*****
*
*
*   subroutine median(size)
*
*****
*   Variable declaration *
*****
*
*   implicit none

```

```

*
      include 'GIFFORD.HEADS]register.h'
*
      integer*4 size, med, lower, upper, rup, cup, rl, ru, rll,
+         cl, cu, i, j, k, l, index, index1, index2,
+         index3
*
      real*4 medc, medr, temp
      dimension medc(1:5), medr(1:5)
*
*****
*   Initialize   *
*****
*
      med = (size-1)/2 + 1
      rup = rows - med + 1
      cup = columns - med + 1
*
*****
*   Filter columns *
*****
*
      do j = med, cup
        do i = med, rup
          rl = i - med + 1
          ru = i + med - 1
          cl = j - med + 1
          cu = j + med - 1
          index = 1
          do l = cl, cu
            medc(1) = real(imase(rl,l))
            rll = rl + 1
            index1 = 1
            do k = rll, ru
              index2 = index1
              do while (real(imase(k,l)) .LT.
+                 medc(index2) .AND. index2 .GE. 1)
                index3 = index2 + 1
                medc(index3) = medc(index2)
                index2 = index2 - 1
              enddo
              index2 = index2 + 1
              medc(index2) = real(imase(k,l))
              index1 = index1 + 1
            enddo
            medr(index) = medc(med)
            index = index + 1
          enddo
        enddo
*
*****
*   Filter rows   *
*****
*
      index = 1

```

```

        medc(1) = medr(1)
        do l = 2, size
            index1 = index
            do while (medr(l) .LT. medc(index1) .AND.
                +         index1 .GE. 1)
                index2 = index1 + 1
                medc(index2) = medc(index1)
                index1 = index1 - 1
            enddo
            index1 = index1 + 1
            medc(index1) = medr(l)
            index = index + 1
        enddo
        image(i,j) = cmplx(real(image(i,j)),medc(med))
    enddo
enddo
call c_switch()
*
*****
*   COPY border values   *
*****
*
    k = med - 1
    l = rup + 1
    do j = 1, k
        do i = 1, rows
            temp = zimag(image(i,j))
            image(i,j) = cmplx(temp,temp)
        enddo
    enddo
    do j = med, cup
        do i = 1, k
            temp = zimag(image(i,j))
            image(i,j) = cmplx(temp,temp)
        enddo
        do i = 1, rows
            temp = zimag(image(i,j))
            image(i,j) = cmplx(temp,temp)
        enddo
    enddo
    k = cup + 1
    do j = k, columns
        do i = 1, rows
            temp = zimag(image(i,j))
            image(i,j) = cmplx(temp,temp)
        enddo
    enddo
*
return
end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: range.for
*
*****
*
*
*   ROUTINE:          subroutine range(min,max)
*
*
*   DESCRIPTION:      This routine finds the range of values
*                     for the real part of a complex image
*                     array.
*
*
*   DOCUMENTATION
*   FILES:           None.
*
*
*   ARGUMENTS:       min      real*4
*                     Returned minimum value.
*
*                     max      real*4
*                     Returned maximum value.
*
*
*   INCLUDE:         register.h
*
*   RETURN:          None.
*
*
*   ROUTINES
*   CALLED:          None.
*
*
*   AUTHOR:          Ronald A. Gifford
*
*
*   DATE CREATED:    September 24, 1987
*
*
*   REVISIONS:       12/8/87          Changed minimum to range.
*
*****
*
*   subroutine range(min,max)
*
*****
*   Variable declarations  *

```



```

*****
*
*   implicit none
*
*   include '[GIFFORD.HEADS]register.h'
*
*   real*4 min, max, temp
*
*   integer*2 i, j
*
*****
*   Search for range.   *
*****
*
*       min = real(image(1,1))
*       max = min
*       do j = 1, columns
*           do i = 1, rows
*               temp = real(image(i,j))
*               if (temp .LT. min) then
*                   min = temp
*               else
*                   if (temp .GT. max) then
*                       max = temp
*                   endif
*               endif
*           enddo
*       enddo
*       return
*
*   end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: slice.for
*
*****
*
*
*   ROUTINE:      subroutine slice()
*
*
*   DESCRIPTION:  This routine performs density slicing
*                 using a total of eight colors.
*
*
*   DOCUMENTATION
*   FILES:       None.
*
*
*   ARGUMENTS:   None.
*
*
*   INCLUDE:     color.h
*                density.h
*
*
*   RETURN:      red()           Thresholded values
*                blue()         Thresholded values
*                green()        Thresholded values
*
*
*   ROUTINES
*   CALLED:      None.
*
*
*   AUTHOR:      Ronald A. Gifford
*
*
*   DATE CREATED: February 26, 1988
*
*
*   REVISIONS:
*
*****
*
*   subroutine slice()
*
*****

```

```

*      Variable declarations      *
*****
*
*      implicit none
*
*      include '[GIFFORD,HEADS]color.h'
*      include '[GIFFORD,HEADS]density.h'
*
*      integer*4 i, j
*
*      rrow = row
*      rcol = col
*      srow = row
*      scol = col
*      brow = row
*      bcol = col
*
*      do j = 1, col
*         do i = 1, row
*            rd(i,j) = 0
*            sr(i,j) = 0
*            bl(i,j) = 0
*         enddo
*      enddo
*
*****
*      Threshold      *
*****
*
*      do j = 1, col
*         do i = 1, row
*            if (image(i,j) .GE. rlo .AND.
+             image(i,j) .LE. rhi) then
*               rd(i,j) = 255
*            else if (image(i,j) .GE. blo .AND.
+             image(i,j) .LE. bhi) then
*               bl(i,j) = 255
*            else if (image(i,j) .GE. slo .AND.
+             image(i,j) .LE. shi) then
*               sr(i,j) = 255
*            else if (image(i,j) .GE. wlo .AND.
+             image(i,j) .LE. whi) then
*               sr(i,j) = 255
*               rd(i,j) = 255
*            else if (image(i,j) .GE. mlo .AND.
+             image(i,j) .LE. mhi) then
*               rd(i,j) = 255
*               bl(i,j) = 255
*            else if (image(i,j) .GE. clo .AND.
+             image(i,j) .LE. chi) then
*               sr(i,j) = 255
*               bl(i,j) = 255
*            else if (image(i,j) .GE. wlo .AND.
+             image(i,j) .LE. whi) then

```

```
        sr(i,j) = 255
        rd(i,j) = 255
        bl(i,j) = 255
    endif
enddo
enddo
return
end
```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: square_stretch.for
*
*****
*
*   ROUTINE:          subroutine square_stretch(x0,x1)
*
*   DESCRIPTION:      This routine performs a contrast stretch
*                     using a simple squared polynomial.
*
*   DOCUMENTATION
*   FILES:           None.
*
*   ARGUMENTS:       x0          real*4
*                     Lower cutoff point
*
*                     x1          real*4
*                     Upper cutoff point
*
*   INCLUDE:         resistor.h
*
*   RETURN:          image()      Transformed image.
*
*   ROUTINES
*   CALLED:          None.
*
*   AUTHOR:          Ronald A. Gifford
*
*   DATE CREATED:    October 4, 1987
*
*   REVISIONS:       February 29, 1988 Add cutoff points
*
*****
*
*   subroutine square_stretch(x0,x1)
*
*****
*   Variable declaration *

```

```

*****
*
*      implicit none
*
*      include '[GIFFORD,HEADS]register.h'
*
*      integer*4 i, j
*
*      real*4 x0, x1, temp, const
*
*****
*      Perform transformation *
*****
*
*      const = 255.0/((x1-x0)**2)
*      do j = 1, columns
*          do i = 1, rows
*              if (real(image(i,j)) .LT. x0) then
*                  image(i,j) = cmplx(0.0,0.0)
*              else if (real(image(i,j)) .GT. x1) then
*                  image(i,j) = cmplx(255.0,0.0)
*              else
*                  temp = const*((real(image(i,j))-x0)**2)
*                  image(i,j) = cmplx(temp,0.0)
*              endif
*          enddo
*      enddo
*
*      return
*      end

```

```

*****
*
*   Department of Electrical and Computer Engineering
*   Kansas State University
*
*   VAX FORTRAN source filename: translate.for
*
*****
*
*
*   ROUTINE:      subroutine translate()
*
*
*   DESCRIPTION:  This routine translates an image file in
*                 the spatial domain so that the Fourier
*                 representation is centered in the
*                 frequency square. The array must be a
*                 square array.
*
*
*   DOCUMENTATION
*   FILES:        None.
*
*
*   ARGUMENTS:    None.
*
*
*   INCLUDE:      register.h
*
*
*   RETURN:       image.h      Translated image.
*
*
*   ROUTINES
*   CALLED:       None.
*
*
*   AUTHOR:       Ronald A. Gifford
*
*
*   DATE CREATED: August 27, 1987
*
*
*   REVISIONS:
*
*****
*
*   subroutine translate()
*
*****
*   Variable declarations. *
*****

```



```

*
  implicit none
*
  include '[GIFFORD.HEADS]register.h'
*
  integer*4 i, j
*
  real*4 rem
*
*****
*   Translate image.   *
*****
*
  do j = 1, columns
    do i = 1, rows
      rem = ((i+j)/2.0) - ((i-j)/2)
      if (rem .NE. 0.0) then
        image(i,j) = cmplx(-1.0,0.0)*image(i,j)
      endif
    enddo
  enddo
*
  return
*
  end

```

IMAGE ENHANCEMENT TECHNIQUES FOR DEGRADED FINGERPRINTS

by

RONALD ALLEN GIFFORD

B.S., Kansas State University, 1986

AN ABSTRACT OF A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

ELECTRICAL ENGINEERING

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1988

ABSTRACT

This thesis presents the results of digital image enhancement techniques applied to degraded fingerprints. A brief discussion of fingerprint identification and recovery techniques is included for background.

Various image enhancement techniques which were employed are explained. Enhancement techniques are divided into spatial-domain and frequency-domain operations.

Two procedures which produced good results are outlined. These procedures consist of combinations of several enhancement operations. Details of the results obtained from applying these procedures to a fingerprint sample recovered using the modified iodine-silver plate method are incorporated into the discussion.