AEDC-TR-87-37
c.3

# Comparison of 2-D Fourier and Spatial Techniques for Removing Noise in Real-Time X-Ray Radiography

David L. Carson
University of Tennessee Space Institute

April 1988

Final Report for Period October 1, 1986 – December 31, 1987

Approved for public release; distribution is unlimited.

ARNOLD ENGINEERING DEVELOPMENT CENTER
ARNOLD AIR FORCE BASE, TENNESSEE
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE

## NOTICES

When U. S. Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, or in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Qualified users may obtain copies of this report from the Defense Technical Information Center.

References to named commercial products in this report are not to be considered in any sense as an endorsement of the product by the United States Air Force or the Government.

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

## APPROVAL STATEMENT

This report has been reviewed and approved.

THEODORE A. BAPTY, 1st Lt, USAF
Directorate of Technology
Deputy for Operations

Approved for publication:

FOR THE COMMANDER

Marion L. Laster

MARION L. LASTER
Technical Director
Directorate of Technology
Deputy for Operations

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

## REPORT DOCUMENTATION PAGE

*Form Approved OMB No. 0704-0188*

| | |
|---|---|
| 1a. REPORT SECURITY CLASSIFICATION<br>UNCLASSIFIED | 1b. RESTRICTIVE MARKINGS |
| 2a. SECURITY CLASSIFICATION AUTHORITY<br>2b. DECLASSIFICATION / DOWNGRADING SCHEDULE | 3. DISTRIBUTION / AVAILABILITY OF REPORT<br>Approved for public release; distribution is unlimited. |
| 4. PERFORMING ORGANIZATION REPORT NUMBER(S)<br>AEDC-TR-87-37 | 5. MONITORING ORGANIZATION REPORT NUMBER(S) |
| 6a. NAME OF PERFORMING ORGANIZATION<br>Arnold Engineering Development Center<br>6b. OFFICE SYMBOL *(If applicable)*<br>DOT | 7a. NAME OF MONITORING ORGANIZATION |
| 6c. ADDRESS *(City, State, and ZIP Code)*<br>Air Force Systems Command<br>Arnold Air Force Base, TN 37389-5000 | 7b. ADDRESS *(City, State, and ZIP Code)* |
| 8a. NAME OF FUNDING / SPONSORING ORGANIZATION<br>Arnold Engineering Development Center<br>8b. OFFICE SYMBOL *(If applicable)*<br>DO | 9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER |
| 8c. ADDRESS *(City, State, and ZIP Code)*<br>Air Force Systems Command<br>Arnold Air Force Base, TN 37389-5000 | 10. SOURCE OF FUNDING NUMBERS<br>PROGRAM ELEMENT NO. 65807F<br>PROJECT NO.<br>TASK NO<br>WORK UNIT ACCESSION NO |

11. TITLE *(Include Security Classification)*
Comparison of 2-D Fourier and Spatial Techniques for Removing Noise in Real-Time X-Ray Radiography

12. PERSONAL AUTHOR(S)
Carson, D. L., University of Tennessee Space Institute, Tullahoma, Tennessee

| 13a. TYPE OF REPORT | 13b TIME COVERED | 14. DATE OF REPORT *(Year, Month, Day)* | 15. PAGE COUNT |
|---|---|---|---|
| Final | FROM 10/1/86 TO 12/31/87 | April 1988 | 46 |

16. SUPPLEMENTARY NOTATION
Available in Defense Technical Information Center (DTIC).

17. COSATI CODES

| FIELD | GROUP | SUB-GROUP |
|---|---|---|
| 20 | 08 | |
| 21 | 08 | .2 |

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
X-ray inspection
nonintrusive diagnostic techniques
frequency domain filtering
solid-propellant rocket motors

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

When X-raying rocket propulsion systems during testing, periodic noise can corrupt the overall quality of an X-ray image. This periodic noise results from vibrations caused by the rocket firing and/or from the pulsing of the X-ray source. A method of removing band-limited noise for enhancement of real-time radiographic images of propulsion systems under test conditions is, therefore, desired to improve data analysis. This investigation compares the Fourier and spatial techniques relative to their ability to eliminate random and periodic noise. Fast Fourier transform (FFT) techniques are discussed with respect to efficiency and ease of implementation. Methods of removing low- and high-frequency and band-limited noise are demonstrated using both Fourier and spatial techniques. The radix-2 method of performing an FFT was selected for this investigation because of ease of implementation and reasonable computation time for a data transfer processor. The Fourier domain techniques were found to be better suited for removing band-limited periodic noise because extensive spatial filtering (Continued)

| | | |
|---|---|---|
| 20. DISTRIBUTION / AVAILABILITY OF ABSTRACT<br>□ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT □ DTIC USERS | 21. ABSTRACT SECURITY CLASSIFICATION<br>UNCLASSIFIED | |
| 22a. NAME OF RESPONSIBLE INDIVIDUAL<br>C. L. Garner | 22b. TELEPHONE *(Include Area Code)*<br>(615) 454-7813 | 22c. OFFICE SYMBOL<br>DOCS |

19. ABSTRACT (Concluded)

by means of frame averaging destroys the time resolution of the image.

# PREFACE

The work reported herein was conducted by the Arnold Engineering Development Center (AEDC), Air Force Systems Command (AFSC). These results were obtained by Sverdrup Technology, Inc., AEDC Group, operating contractor for the propulsion test facilities at the AEDC, AFSC, Arnold Air Force Base, Tennessee, under Project Number DB90EW, and under the University of Tennessee Space Institute/Sverdrup Technology, Inc. Subcontract Number A7B-012. The Air Force Project Manager was 1Lt. T. A. Bapty, DOTR. The research was conducted from October 1, 1986 to December 31, 1987, and the manuscript was submitted for publication on January 15, 1988.

## CONTENTS

## ILLUSTRATIONS

## APPENDIX

# 1.0 INTRODUCTION

At the Arnold Engineering Development Center (AEDC), real-time X-ray radiography and image enhancement techniques are currently being used to determine the structural integrity and performance characteristics of solid-propellant rocket motor (SRM) components such as rocket nozzles, propellant, and the insulator and case interface. In addition, propellant burn characteristics, slag formation, and other anomolies have been identified during motor firing (Ref. 1). These characteristics can affect the ballistic performance of a rocket during flight. However, one of the problems encountered during the recording of the radiographic images during test firing of the SRM's is that the quality of the recorded X-ray images is corrupted by noise. The noise appears as evenly spaced bars that are band-limited, periodic noise within the images. The periodic noise is produced by both the acoustical vibrations of the rocket motor firing and by the pulsing of the high-energy 6-Mev X-ray source. Random noise is produced in the images by the electronics of the image detector (camera), low light level, and X-ray scatter. To reduce the noise content in these images, various spatial techniques such as frame averaging and convolution were used. Because the image enhancement programs used with the image processing hardware were written for spatial domain processing, these processing techniques provided limited capability relative to noise. The resulting enhanced images still contained a certain amount of noise. Likewise, the extensive use of temporal techniques created blurred images with poor time resolution, and the spatial techniques could not completely eliminate the noise either.

The purpose of this study was to develop a means for minimizing the total noise content within recorded radiographic images without altering the time resolution of the image data. To achieve this objective, image enhancement techniques were applied in the frequency (Fourier) domain. This project used a fast Fourier transform (FFT) technique for generating a frequency domain representation of the image data and noise components of the recorded image. After visually determining the locations of the noise terms, a band-reject filter was used to remove them. An inverse fast Fourier transform (IFFT) was then applied to the resulting filtered data to provide a spatial domain representation of the image on the video display. If the image provided acceptable results, the analysis was complete. Otherwise, the process was repeated.

In Section 2.0, a literature survey is presented of the various image analysis techniques used for enhancing image data. Fast Fourier transform algorithm analysis, Fourier domain techniques, and spatial domain techniques are also discussed. The hardware and software used in the implementation of the frequency domain techniques for image enhancement are described in Section 3.0. In Section 4.0, the results obtained by the frequency domain processing technique are presented and compared to those in the spatial domain. In Section 5.0, a summary and conclusion of the study is presented.

# 2.0 TECHNIQUE ANALYSIS

## 2.1 FFT ALGORITHM ANALYSIS

In recent years, a number of new FFT algorithms have been developed that are more flexible, and in some instances faster, than the original Cooley-Tukey radix-2 FFT. In the research performed by M.A. Mehalic (Ref. 2) and J. D. Blaken (Ref. 3), the architecture and structure of the various FFT's are compared for execution speed, memory utilization, etc. Mehalic classified Von Neumann-type computer architectures into three categories in order to determine which algorithm performed the best on a specific system. These architectures are floating point processor, data transfer processor, and vector array processor. Mehalic demonstrated which algorithms performed optimally for each basic architecture. Blaken described an algorithm selection criterion based on algorithm operations count and memory requirements for the algorithm. By taking into consideration the guidelines established by Mehalic and Blaken in addition to the requirements of the work to be performed, a specific FFT was selected.

### 2.1.1 The Prime Factor Algorithm (PFA)

The Prime Factor Algorithm (PFA) is based on the Winograds' technique for computing prime length discrete Fourier transforms (DFT's) with circular convolution. A detailed theoretical development of the PFA can be found in Refs. 4 through 8 and will not be discussed here. Kolba and Parks (Ref. 6) developed a method of comparison of the prime factor FFT, and the more conventional Cooley-Tukey FFT's where the number of multiplications, m', for a length N Cooley-Tukey radix-2 FFT is

$$m' = 3\left(\frac{N}{2}\log N - 3\frac{N}{2} + 2\right) \tag{1}$$

where log ( ) refers to the base-2 algorithm. The number of additions is equal to

$$\alpha' = 2N\log N + 5(m') \tag{2}$$

The number of multiplications for the prime factor FFT is given by

$$m_p = 2\left(M_2M_3\mu_1 + M_1M_3\mu_2 + M_1M_2\mu_3\right) \tag{3}$$

where the $M_i$'s are the length of the short DFT's and $\mu_i$'s are the total number of multiplications for length M DFT's. For a prime factor FFT of a length of 504, the short-length DFT's $M_1$, $M_2$, and $M_3$ are equal to 9, 7, and 8, respectively. The numbers of multiplication $\mu_i$'s for each DFT are 10, 8, and 2, respectively, and the total multiplication count is equal to 2,524 (Ref. 4). The number of additions equals

$$\alpha_p = 2\left(M_2M_3\alpha_1 + M_1M_3\alpha_2 + M_1M_2\alpha_3\right) \tag{4}$$

where the $\alpha_i$'s are the number of additions for a length of $M_i$ DFT's. For a length of 504 FFT, the $\alpha_i$'s are 42, 36, and 26, respectively, and the total number of additions is equal to 13,164. The total operation count for the PFA is the combined total of the number of additions and multiplications, which is 15,688. By using Eqs. (1) and (2), the operations count for a roughly equivalent 512 length N Cooley-Tukey FFT is determined to be 36,900. It can be concluded from the total operations count and the number of multiplications for the PFA that a computer that performs multiplications efficiently is better suited for performing the FFT using the PFA.

From Eqs. (3) and (4), it was determined by Kolba and Parks that the PFA computed the DFT approximately 50 percent faster than Singleton's mixed radix FFT (Ref. 6). This indicated that the PFA is more efficient than the mixed radix FFT in performing the DFT for data sets with length equal to the product of prime number factors. In comparing the prime factor technique for performing the DFT with the techniques described in the following section, the PFA was superior. However, because of the complexity in implementing the PFA and the fact that execution speed was not a critical parameter in this work, the PFA was not used.

### 2.1.2 Mixed Radix

The mixed radix FFT is called a decimation in frequency algorithm since it decomposes the frequency parameter instead of the time or spatial parameter. The algorithm decomposes the data set into subsets of odd prime integers and even prime integers. The appropriate short transforms are performed. The Singleton (Ref. 9) mixed radix is based on the method proposed by Cooley and Tukey (Ref. 10). The sequence length N is factored into m different factors. The transform is then decomposed into m steps with $N/n_j$ transformations of size $n_j$. The transform is computed so that the output array is in a scrambled format, and a procedure is performed to rearrange the results in order. In Ref. 4, a comparison of the mixed radix and fixed radix FFT's was performed. It was concluded that the mixed radix is roughly equivalent in efficiency with the fixed radix algorithm.

Another type of mixed radix FFT is the Duhamel-Hollman split radix (Ref. 11). It has a total operations count that is the lowest known (Ref. 4). A complex four-butterfly split-radix FFT requires $N(M - 1) + 4$ real multiplications and $3N(M - 1) + 4$ real additions

(Ref. 11), where the terms N and M are determined from the equation $n = 2^M$ where N is the data sequence length. There are fewer operations with this technique than a five-butterfly radix-16 FFT and the same number of operations as obtained for the shorter length N DFT's using Winograds' Fourier Transform (Ref. 5). However, the main disadvantage in the implementation of the mixed radix FFT is the availability of enough memory to compute the small multivariate DFT's in addition to the fact that it is not as fast as the radix-2 on systems with floating point software (Ref. 2).

### 2.1.3 The Fixed Radix-2 FFT

The original Cooley-Tukey FFT, which is normally referred to as the radix-2 FFT or power of 2 transforms, uses a divide and conquer scheme to compute the DFT. The FFT uses the Chinese Remainder Theorem (CRT) to define its parameter on the time and frequency index. By using the CRT, the data sequence becomes cyclic (Refs. 4 and 10) and lends itself to a matrix factorization of the DFT. The matrix factorization is the basic principle behind the radix-2 FFT. By structuring the data in a factored matrix form as shown in the length $N = 8$ data flow graph of Fig. 1, the amount of complex computations is reduced from $N^2$ to n log n (Ref. 12). Because of its structure, the radix-2 is a relatively efficient algorithm for computer architectures that use floating point software to perform mathematical computations (Ref. 2). It is also a simple algorithm to implement, and it lends itself to a greater variety of data sequence lengths. Because of these features, the radix-2 FFT was selected as the 1-D transform technique to be used on the 2-D data processed in this project.

### 2.1.4 The Radix-4 FFT

The radix-4 FFT is very similar to the radix-2. The main difference between the radix-4 and radix-2 techniques is the reduction in the amount of multiplications performed. The radix-4 will require approximately 30-percent fewer multiplications than the radix-2 (Refs. 2, 3, and 4). Radix-4 does conform to the same basic divide and conquer scheme as the radix-2. However, the disadvantage in selecting a radix-4 algorithm over a radix-2 is in the selection of a length N where $N = 4^m$, and m is an integer. This restricts the number of lengths available for procesing data.

## 2.2 ROW-COLUMN 2-D FFT

Because of the increased interest in image processing where 2-D signals must be manipulated in the frequency domain, various techniques have been devised to extend 1-D algorithms to 2-D. A technique that has been used very successfully in performing the 2-D FFT is the row-column method.

The row-column approach to performing the FFT is the most common practice today for processing 2-D signals. This is the approach described in Gonzales (Ref. 13) and was used to generate frequency domain information for the images in this report.

The discrete Fourier transform for a square N × N 2-D signal can be expressed as

$$X(u,v) = \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} X(n,m) \exp\left(-j2\pi\frac{(un+vm)}{N}\right) \tag{5}$$

for u,v = 0.1,..., N − 1. Equation (5) can be expressed in separable form as

$$X(u,v) = \frac{1}{N} \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi un}{N}\right) \sum_{m=0}^{N-1} X(n,m) \exp\left(\frac{-j2\pi vm}{N}\right) \tag{6}$$

for u,v = 0.1,..., N − 1. The principle advantage in writing the DFT in separable form is that successive application of the 1-D transform will give X(u,v). This becomes evident if Eq. (6) is expressed as

$$X(u,v) = \frac{1}{N} \sum_{n=0}^{N-1} \hat{x}(n,\, v) \exp\left(\frac{-j2\pi un}{N}\right) \tag{7}$$

where

$$\hat{x}\,(n,v) = N\left[\frac{1}{N} \sum_{n=0}^{N-1} X(n,\, m) \exp\left(\frac{-j2\pi vm}{N}\right)\right] \tag{8}$$

For each value of N, the expression in the brackets is a 1-D transform with frequency values v = 0,1,..., N − 1. Therefore, the 2-D function X(n,v) is obtained by taking a transform along each row or column of X(n,m) and multiplying the result by N. The desired result is then obtained by taking a transform along each column or row of X(n,v) shown in Eq. (8). The procedure is summarized in Fig. 2. By using this technique and the radix-2 1-D FFT, 2-D transform of image data can be performed.

## 2.3 SPATIAL DOMAIN TECHNIQUES

### 2.3.1 Frame Averaging

The averaging of multiple images is used to reduce random noise in an image (Ref. 13). A noisy image g(x,y) is defined as the original image f(x,y) corrupted with additive zero-mean noise $\eta(x,y)$.

$$g(x,y) = f(x,y) + \eta(x,y) \tag{9}$$

The effect of adding a set of M noisy images $g_i(x,y)$ can be illustrated by noting that the averaged image $\bar{g}(x,y)$ can be represented by

$$\bar{g}(x,y) = \frac{1}{M} \sum_{i=1}^{M} g_i\,(x,y) \tag{10}$$

where the expected value of g(x,y), i.e. E[g(x,y)] is given by

$$E[g(x,y)] = f(x,y) \tag{11}$$

and

$$\sigma_{\bar{g}}{}^2(x,y) = \frac{1}{M}\,\sigma_n{}^2\,(x,y) \tag{12}$$

where $\sigma_{\bar{g}}^2/g(x,y)$ and $\sigma_n{}^2(x,y)$ are the variances of $\bar{g}$ and $\eta$, respectively, at coordinates (x,y). The standard deviation $\sigma_{\bar{g}}$ is given as

$$\sigma_{\bar{g}}\,(x,y) = \frac{1}{\sqrt{M}}\sigma_n\,(x,y) \tag{13}$$

Since the expected value of g(x,y) equals f(x,y), then $\bar{g}(x,y)$ will approach f(x,y) as the number of noisy images used in the averaging process increases.

### 2.3.2 Neighborhood Averaging

Image noise can also be reduced by performing a technique generally referred to as neighborhood averaging. This is a convolution procedure and a straight forward method of reducing noise in the spatial domain. Given an image, f(x,y), a smoothed image is obtained by replacing each pixel by the average of all the pixels in a predefined region, or kernel surrounding it. This is defined by

$$g(x,y) = \frac{1}{M} \sum_{(n,m)\epsilon S} f(n,m) \tag{14}$$

where S is the set of coordinate points in the neighborhood of the point (x,y), n and m are points in the set, and M is the total number of points defined by the coordinates in S. The median filter process is performed much like the average filter process, except that the median value is substituted into the center pixel of the kernel instead of the average value of the kernel.

### 2.3.3 Sobel Operators

The Sobel operation is an edge-enhancement technique that is based on taking the gradient of the image. Given some three-by-three region as in Fig. 3, the pixel at e with a new value is defined as

$$e \leftarrow G = \sqrt{G_x^2 + G_y^2} \tag{15}$$

where $G_y$ is the gradient along the y direction given by

$$G_y = (g + 2h + i) - (a + 2b + c) \tag{16}$$

and $G_x$ is the gradient along x,

$$G_x = (c + 2f + i) - (a + 2d + g) \tag{17}$$

It is evident in Eq. (16) that $G_y$ is the difference between the first and third rows, where the elements closer to "e" are weighted twice as much as the corner values. Thus, $G_y$ represents an estimate of the derivative in the y direction. Similiarly, $G_x$ is an estimate of the derivative in the x direction. Equations (15), (16), and (17) can be implemented by using the two templates shown in Fig. 4, often referred to as Sobel operators. Implementation of Eq. (15) requires the square root of the sum of the squared responses of the two templates. Discussion of the gradient and templates can be found in Ref. 13.

## 2.4 FREQUENCY DOMAIN TECHNIQUES

In this section, the different types of Fourier domain techniques for smoothing images and enhancing edges are discussed. The relationship of spatial domain convolution to Fourier domain filtering is also presented.

The convolution of a spatial domain image with a filter function h(x,y) is represented by

$$g(x,y) = f(x,y) * h(x,y) \tag{18}$$

where g(x,y) is the filtered image and * represents the convolution process. The Fourier transform of this equation is

$$G(u,v) = F(u,v)H(u,v) \tag{19}$$

This can be written succinctly as

$$f(x,y) * g(x,y) \longleftrightarrow F(u,v)H(u,v) \tag{20}$$

which states that convolution in the spatial domain is equivalent to a multiplication in the frequency domain. The H(u,v) term in Eqs. (19) and (20) is typically referred to as the filter function in Fourier domain filtering applications. The H(u,v) term in a Fourier domain filter is selected so that specific frequencies in the transform of the original image are attenuated while certain other frequencies of the image are enhanced.

### 2.4.1 Low-Pass Filtering

Low-pass filtering of data in the frequency domain causes edge suppression in the spatial domain image. In the spatial domain, noise that is characterized by high-frequency components can be removed with a low-pass filter. If the high-frequency components of the original image are attenuated, then a smoothed image will be the result.

There are a number of filters available for low-pass filtering, one of which is the ideal low pass. The 2-D ideal low-pass filter has a transfer function that satisfies the following relationship.

$$H(u,v) = \begin{cases} 1, \text{ if } D(u,v) \leq D_0 \\ 0, \text{ if } D(u,v) > D_0 \end{cases} \tag{21}$$

where $D_0$ is the specified positive quantity, and D(u,v) is the radial distance from the center of the frequency plane to the point (u,v) (Ref. 13) and is given by Eq. (22),

$$D(u,v) = (u^2 + v^2)^{1/2} \tag{22}$$

An image that has been processed with an ideal low-pass filter typically has a substantial amount of ringing and is somewhat blurred. The ringing effect is caused by the sharp cutoff of the ideal filter. This can be explained by referring to Eq. (19), which is the Fourier transform of Eq. (18). The spatial extent of h(x,y) in Eq. (18) depends on the radius of the filter function in the frequency domain. By computing the inverse transform of H(u,v) for the ideal low pass, it can be shown that the extent of h(x,y) is inversely proportional to the radius selected in the frequency domain. For example, in one dimension the Fourier transform of a rectangularly shaped "filter" is the sinc function. The equation for a 1-D sinc function is

$$\text{sinc}(u) = AX \left| \frac{\sin \pi uX}{\pi uX} \right| \tag{23}$$

By enlarging the bandwidth of the filter, the extent of the sinc fuction will become less. Therefore, the larger the "filter bandwidth" size, the less pronounced the ringing will be. The 2-D circular ideal low-pass filter of Eq. (21) corresponds to a circularly symmetric Bessel function of the first kind of order 1 (Ref. 14) in the spatial domain. This will appear as radial symmetric ringing in the resulting image. Consequently, the results of the circular ideal low pass is similar to that of the rectangular window low pass in that as the radial distance of the circular filter is enlarged, the radial distance of the Bessel function will decrease.

A method of performing a low-pass filtering operating without the ringing is with an exponential low-pass filter (ELPF). The ELPF with a cutoff frequency at a radial distance $D_0$ from the center of the frequency domain is given by Eq. (24), where n effects the rate of attentuation in the filter H(u,v),

$$H(u,v) = \exp^{-[D(u,v)/D_0]^n} \tag{24}$$

The radius $D_0$ is selected such that all terms outside of the region where $D(u,v) = D_0$ are highly attenuated. The exponential low-pass filter function is continuous, and therefore, the sharp cutoff prevalent in the ideal filter is avoided, thus avoiding the ringing effect.

There are a number of other filters available for low-pass filtering of frequency domain data. However, they will not be presented here because they were not considered for this project. A more thorough discussion of the various types of low-pass filters that are available can be found in (Refs. 13 and 14).

### 2.4.2 High-Pass Filtering

It was shown in the previous section that an image can be blurred by attenuating the high-frequency content of an image. Because edges and other abrupt changes in gray levels are associated with high-frequency components, these features will be enhanced after high-pass filtering is performed. Image sharpening is the main reason for high-pass filtering in the frequency domain. An ideal high-pass filter is one whose transfer function satisfies the relation

$$H(u,v) = \begin{cases} 0, \text{ if } D(u,v) \leq D_0 \\ 1, \text{ if } D(u,v) > D_0 \end{cases} \tag{25}$$

where the function D(u,v) is the same as that defined in Section 2.5.1.

### 2.4.3 Band-Reject Filtering

The band-reject filter was the primary filter used in this research. An ideal band-reject filter is given by the relation

$$H(u,v) = \begin{cases} 0, \text{ if } D(u,v) \le D_0 \\ 1, \text{ if } D(u,v) > D_0 \end{cases} \tag{26}$$

where the definition of D(u,v) has changed and is defined as

$$D(u,v) = [(u - u_0)^2 + (v - v_0)^2]^{1/2} \tag{27}$$

The ideal band-reject filter suppresses all frequencies in the neighborhood $D_0$ about the point $(u_0, v_0)$ and $(-u_0, -v_0)$ when negative frequencies are considered. The band-reject filtering is performed in pairs about the frequency domain axes to account for the symmetric nature of Fourier transformed real data. The use of a band-reject filter offers a unique advantage over spatial domain filters in that it can remove band-limited noise in an image without destroying the time resolution of the data. The band-reject filter does not affect the original image in the same manner as the low-pass filter because the sharp cutoff is normally in a region of higher frequencies, and the ringing effect will not be as evident. By filtering an image with this filter, a great deal of detail will remain in the image in both the high- and low-frequency regions while band-limited noise is eliminated.

## 3.0 IMAGE PROCESSING HARDWARE AND SOFTWARE

### 3.1 IMAGE PROCESSOR HARDWARE

The image processor used in this effort is one of three basic types of image processors with a basic configuration as shown in Fig. 5. The first type is the off-line image processor. This type of system is configured with a frame grabber that is a high-speed, flash, analog-to-digital (A/D) converter and is typically interfaced to a host computer such as a DEC VAX or PC. A number of software support routines, such as 2-D FFT's, histogram equalization, and N × N convolution kernel filters are usually made available for this type of system. The off-line image processor is used to analyze visual data by first capturing a frame of data with the frame grabber and storing the digitized data in image memory. Specialized software is then executed by the host computer to perform image analysis on the digitized data.

The second type of image processor, which is the type used in this work, has a pipeline processor to perform real-time operations on images. The capabilities of real-time image processors vary for different systems. The Quantex which is a real-time system with a pipeline processor (See Appendix A for detailed information), is able to perform real-time frame averaging, frame summing, high- and low-pass spatial filtering operations and pseudocoloring. Another system, the Matrox® MVP-AT, is also capable of real-time operations similar to that of the Quantex except for spatial filtering. Real-time systems also use a flash analog-to-digital converter for digitizing video data. However, unlike the off-line systems that store data in image memory, the digitized data is ported to the pipeline processor. The image data is then processed in real-time or stored in memory. If the data is stored in image memory, customized software can then be used for off-line enhancement just as in the previous image processing system.

The third type of image processor, such as the Datacube® MaxVision system, is one that integrates more enhancement techniques in a real-time environment using custom hardware. One-dimensional finite impulse response (FIR) filters, segmentation operations, and various other functions, such as 2-D FFT's, are generally added by software or with add-on array processors. Customized image processors are the most expensive.

## 3.2 SOFTWARE FOR FREQUENCY DOMAIN PROCESSING

The program that performs the filtering operation uses an ideal high-pass, low-pass, and band-reject filter. The filtering program was written to allow the user to interactively select, through computer terminal input, which areas of the Fourier transformed data to filter out. After filtering, the program displays the modified Fourier data and allows the user to retain the modified data or filter the original Fourier data set over again. Upon obtaining a satisfactory data set, it is then transformed back to the spatial domain.

The initial set in processing image data was to truncate the 640-by-480 image into a 480-by-480 image array, after which the data array was decimated to a 240-by-240 array and then zero padded to a 256-by-256 array that is a standard length for a power of 2 FFT. This procedure reduced the system memory requirement and computation time of the FFT. The decimation of the image was essential because a full image required in excess of 4 megabytes of RAM for real, complex, and temporary data arrays. The decimated image arrays reduced the computation time of the FFT program, since the data arrays could be maintained in the system memory instead of the computer disk. The decimation was performed by a program called "Half2," which read the image data by calling the Quantex subroutines DVPOPN and DVPREA (described in Appendix A), which opens the procesor memory board and allows image data to be read into computer memory. The decimation is performed by reading every other horizontal line and storing it into a temporary array, after which every other pixel of each line in the temporary array is placed into an output array and transferred to the memory/processor board using the Quantex DVPWRI subroutine. The data array that results from the decimation process is shown in Fig. 6.

The DVPWRI subroutine enables the program to place data anywhere in the memory/processor board. The data are input to system memory in integer*1 or byte data format for each pixel value in the image buffer (array). Prior to performing the FFT, the data are scaled to values between 0 and 255 by placing each pixel into a temporary integer*2 variable and offsetting the pixel value by adding it to 256 if it is less than zero. This is necessary because the Fortran compiler interprets an 8-bit byte as signed magnitude values ranging from − 127 to 128. The representation of the data in the image processor, however, extends from 0 to 255 with no sign bit. After offsetting, data are then converted to floating point format and multiplied by $(-1)^{i+j}$, which centers the DC value of the Fourier transform in the frequency plane. The row-column method of performing the 2-D FFT is then implemented by first calling a radix-2 FFT subroutine, which performs a 1-D FFT on each of the rows of a 2-D data array. The rows and columns are then transposed, and the subroutine is called again to complete the 2-D process.

The floating point data are then scaled to values between 0 and 255, which will allow the image processor to display the Fourier transform of the data. At the option of the user, the floating point data can also be log-scaled to allow the wide dynamic range of the frequency domain data to be displayed.

After completing the scaling process, the data are converted to integer*1 (byte) format and transferred to the memory/processor board using the DVPWRI subroutine. The coordinates of the Fourier transformed image are input to DVPWRI to display the image directly above its spatial domain representation. The approximate location of the noise in the frequency domain image is determined by the user through a subroutine called "Encl" and by using the equation

$$\Delta u = \frac{1}{N\Delta x} \tag{28}$$

and the equation

$$f_N = \frac{N_p}{N\Delta x} \tag{29}$$

Where Δu is the sample interval in the Fourier domain in units of inverse length, Δx is the sample interval in the spatial domain in units of length, N is the number of data samples, $f_N$ is the frequency of the noise in cycles/pixel, and $N_p$ is the periodicity of the noise in cycles/picture. The number of pixels in the frequency domain where the noise will be located can be determined by dividing Δu into $f_N$. From Eqs. (28) and (29), yields

$$n_p = \frac{f_N}{\Delta u} = N_p \tag{30}$$

Therefore, it is seen that the location of the noise frequency components (pixels from the origin) is the same as the number of cycles of noise in the spatial domain image. The $n_p$ value allows the user to determine the distance the noise term is located from the center of the Fourier domain image. The noise components removed with this method were always on the coordinate axis of the frequency domain; therefore, no other technique was necessary for locating the noisy data. The coordinates of the pixel location of the noise frequencies are calculated by adding and subtracting $n_p$ from the center coordinate of the vertical axis in the frequency domain and using a fixed center coordinate along the horizontal axis. The coordinates and the radius of a circle are then input to the computer after the program prompts the user. The circle is then written to the memory/processor board and displayed on the monitor. The circles are used to verify the regions in the frequency domain that are to be filtered out of the image data. After determining the regions in the frequency domain that are to be removed, a subroutine called "Filtr1" is called, and the data locations that correspond to the identified regions are filled with zeros. The filtered frequency domain representation is then transferred back to the memory processor board for viewing. At this point, the inverse FFT can be performed on the filtered data or the data can be discarded, and the process of selecting the filter location and size is repeated. The "Filtr1" program retains a copy of the original frequency domain data set so that the data can be processed interatively. The program continually prompts the user about retaining the filtered Fourier data each time the filtering process is performed. Once the user has obtained a filtered Fourier image that satisfies his requirements, the program destroys the original frequency domain data set by writing the filtered data over the original data. The inverse FFT is then performed on the filtered data. Upon completing the inverse FFT, the filtered image is placed to the right of the original image and directly below the filtered frequency domain representation using DVPWRI.

A summary of the entire process of decimating and frequency domain processing of an image is illustrated in Fig. 7. The FFT program is designed to allow sequential processing with the ICOS program (Ref. 15) using the CSOS environment (Ref. 16). By using the LNK command on the image processor, the user's program is linked to the ICOS program, thereby allowing the user to run the program without exiting ICOS. The ICOS program will temporarily suspend operation until the user program is complete, at which time ICOS will resume. The FFT program is also capable of processing data independent of the ICOS program, in which case the LNK command is not needed.

## 4.0 RESULTS

Fourier domain programs were written to evaluate the advantages of frequency domain processing as opposed to spatial domain processing. The FFT program was used to obtain Fourier domain data that could be either low-pass filtered, high-pass filtered, or band-reject filtered as discussed in the previous section. The primary advantages of using the FFT approach is that images can be filtered without harming time resolution and the ease with which periodic noise is removed. One of the disadvantages of using the FFT approach is the amount of time required to generate a frequency domain image. Because of the long data processing time, in addition to the amount of system memory necessary for storing a 640-by-480 image, it was necessary to decimate the image. In order to obtain a reasonable estimate of the time required to process a full image, the data were decimated into 256-by-256 arrays. The FFT program was then used on each image array. The amount of time necessary for processing the data approximately doubled for each power of two increase in the image size. The 256-by-256 array was processed in approximately 15 min. This processing time is exclusive of the time necessary for the program to search for the first and second largest values in the frequency domain with which the entire data set is scaled. The 15-min processing time of the 256-by-256 array is because of the slow floating point software used by the system. The entire procedure of transforming the data from the spatial domain to the frequency domain, filtering, and performing the inverse 2-D Fourier transform took approximately 60 min. The 60-min length for the procedure was primarily because of the interactive nature of the programs used to perform the filtering.

The results of the frequency domain processed images will be compared to images processed by spatial techniques to show the utility of frequency domain methods. The frame averaging spatial domain technique is examined first. A radiographic image of a small solid-propellant rocket motor (SRM) that was corrupted with noise is illustrated in Fig. 8. The result of frame averaging is illustrated in Fig. 9. In this figure, the image is slightly blurred because of the loss in time resolution in the image caused by averaging over 4 frames.

The results of median and average filtering are illustrated in Figs. 10 and 11, respectively. The median filter in Fig. 10 was performed with a one-by-three kernel, and the average filter in Fig. 11 was performed with a three-by-three kernel. By using a one-by-three kernel size in the median filter process, the edge detail of the image and the noise in Fig. 10 are enhanced. The three-by-three kernel in the average filter, however, created a blurring effect on the image as illustrated in Fig. 11, and in neither case was the noise eliminated. The four images in Fig. 12 are used to illustrate the effect of low-pass filtering in the frequency domain. This picture is a single video frame from a real-time radiographic analysis of an end-burning solid-propellant rocket motor. The low-pass filtered image shown in Fig. 12d was created using a rectangular window function as the filter function as shown in Fig. 12c. The image illustrates the smoothing effect that can be obtained with a frequency domain low-pass filter. In Fig. 12c, the noise term evident in the original image is removed. However, because an ideal low pass was used to perform the filtering, ringing is present in the processed image.

The edge region in each image was zero padded in the spatial domain after decimating the original image to 240 by 240. The zero padding in the image extended the image length in the x and y directions to 256 by 256, which is a standard length for a radix-2 FFT. The zero padding eliminates wrap around error that is associated with the convolution process.

The results of applying an ideal high-pass filter with a circular filter function is illustrated in Figs. 13c and d. This image illustrates how a high-pass filter brings out edge detail. Most of the information in the image is removed, and the picture appears to be black and white exclusively.

The ideal band-reject filter provides a means for removing band-limited periodic noise in the frequency domain. This is illustrated in Fig. 14. The noise identified in the frequency domain was eliminated by a circular filter with a rejection band with a radii of 10 pixels. The resultant spatial domain image, however, still had residual noise present. This noise was caused by second and third harmonics in the frequency domain image. In the images of the SRM shown in Figs. 15 and 16, this noise was removed by filtering out the second and third harmonics. In Figs. 15c and d, it is evident that the fundamental terms and the second harmonic are the primary noise terms in the image, since the resultant spatial domain image is relatively noise free when the third harmonic was left unfiltered in the frequency domain image. The multiple band-reject filters used in Figs. 15 and 16 were implemented with rejection bands of radii 10, 5, and 3 pixels for the fundamental, second, and third harmonics, respectively. This type of filter cannot be implemented in the spatial domain, without an excessive computational effort; consequently, images with narrow periodic low-frequency noise bands are more efficiently filtered in the frequency domain.

## 5.0 SUMMARY AND CONCLUSIONS

In this report, a comparison of the Fourier and spatial domain techniques was presented to provide insight for selecting the appropriate techniques for reducing noise in recorded radiographic images. A method for processing X-ray radiographic images using a 2-D row-column FFT and ideal filters were demonstrated. An efficient technique for computer memory utilization by using image decimation was also developed.

In this research, the various FFT techniques were analyzed for speed of computation and memory utilization. The radix-2 FFT with table look-up and three butterflies was selected for this work because of its ease of implementation, flexibility, and low operations count relative to other radix-2 FFT's, such as a radix-2 with no table look-up or one that does not take advantage of the butterfly structure. The radix-2 FFT implementation was found to be a lot simpler than that of the prime factor algorithm (PFA). It is for this reason that the PFA was not used. However, it was found through the literature survey that the PFA is the optimum FFT for a computer system with a central processing unit (CPU) that performs multiplications efficiently.

The FFT processing techniques could be performed much more efficiently than performed in this work by using computers with floating point coprocessors with at least four megabytes of memory for 512-by-512 image data processing. The resident memory will allow the data to be stored and retrieved from within the computer and, therefore, will improve the processing time of the FFT. For a computer not equipped with a coprocessor but an array processor, the WFTA or PFA are the optimum algorithms to improve the system processing time.

The image decimation technique was used to reduce the data processing time of the FFT by creating a 256-by-256 pixel array. After performing the FFT, the frequency domain data were then visually examined to determine the characteristics of the noise such as the locations and magnitudes of the key frequency components of the image data. Although both magnitude and phase information were calculated for the frequency domain, the phase provided no information that would have contributed to the enhancement of the images and, therefore, was not a concern in the research. After analyzing the data, they were then filtered, and the inverse FFT was performed.

The spatial domain images processed in the frequency domain were compared with images processed using spatial domain techniques. Of the two methods, the spatial domain approach was found to be the most useful for real-time applications and for removing random noise from images because of the speed with which the data could be processed. Frame averaging was beneficial in removing time-varying noise from spatial domain images. The average and median convolution techniques were effective in removing high-frequency noise from spatial images thereby eliminating a need for Fourier domain low-pass filtering.

The primary benefit of Fourier domain processing is the elimination of band-limited, periodic noise. Frequency domain techniques provided a means for determining that band-limited noise had harmonics associated with it. This type of noise could be eliminated in the frequency domain by appropriate placement of the filtering window. By using Fourier domain filters, the noise and the associated harmonics were removed from the images without affecting the overall image resolution.

## REFERENCES

1. Belz, R. A. et al. Radiographic Diagnostics For Tactical Missile Development." 1981 JANNAF Propulsion Meeting, San Diego, California, December 15–17, 1981.

2. Mehalic, M. A. *Effects of Computer Architecture on Fast Fourier Transforms*. M. S. Thesis, Air Force Institute of Technology, Dayton, Ohio, December 1983.

3. Blaken, J. D. *Efficient Computer Implementation of Fast Fourier Transform*. M. S. Thesis. Air Force Institute of Technology, Dayton, Ohio, December 1980.

4. Burrus, C. S. and Parks, T. W. *DFT/FFT Convolution Algorithms*. New York, John Wiley & Sons, Inc., 1985.

5. McCellan, J. H. and Rader, C. M. *Number Theory in Digital Signal Processing*. Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1979.

6. Kolba, Dean P. and Parks, Thomas W. "A Prime Factor FFT Algorithm Using High-Speed Convolution." *IEEE Transactions ASSP*, August 1977.

7. Burrus, Sidney and Eschenbacher, Peter W. "An In-Place In-Order Prime Factor Algorithm." *IEEE Transactions ASSP*, August 1981.

8. Burrus, Sidney and Johnson, Howard W. "On the Structure of Efficient DFT Algorithms." *IEEE Transactions ASSP*, February 1985.

9. Singleton, R. C., "Mixed Radix FFT." *Programs for Digital Signal Processing*. New York, John Wiley & Sons, Inc., 1979.

10. Cooley, J. W. and Tukey, J. W. "An Algorithm for the Machine Computation of Fourier Series." *Mathematics of Computation*, April 1965.

11. Duhamel P. and Hollman H. "Split Radix FFT Algorithm." *Electronic Letters*. Vol. 20, No. 1, January 1984.

12. Brigham, E. Oran. *The Fast Fourier Transform*. Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1974.

13. Gonzales, Rafael C. and Wintz, Paul. *Digital Image Processing*. Sixth Edition. Reading, Massachusetts, Addison-Wesly Publishing Co., 1983.

14. Mersereau, Russel M., and Dudgeon, Dan E. *Multidimensional Digital Signal Processing*. Englewood Cliffs, New Jersey, Prentice-Hall Inc., 1984.

15. Quantex Corporation. *Quantex QX-9000 Users Guide*. October 1984.

16. IBM Corporation. *CSOS Users Guide*. IBM Instruments, 1983.

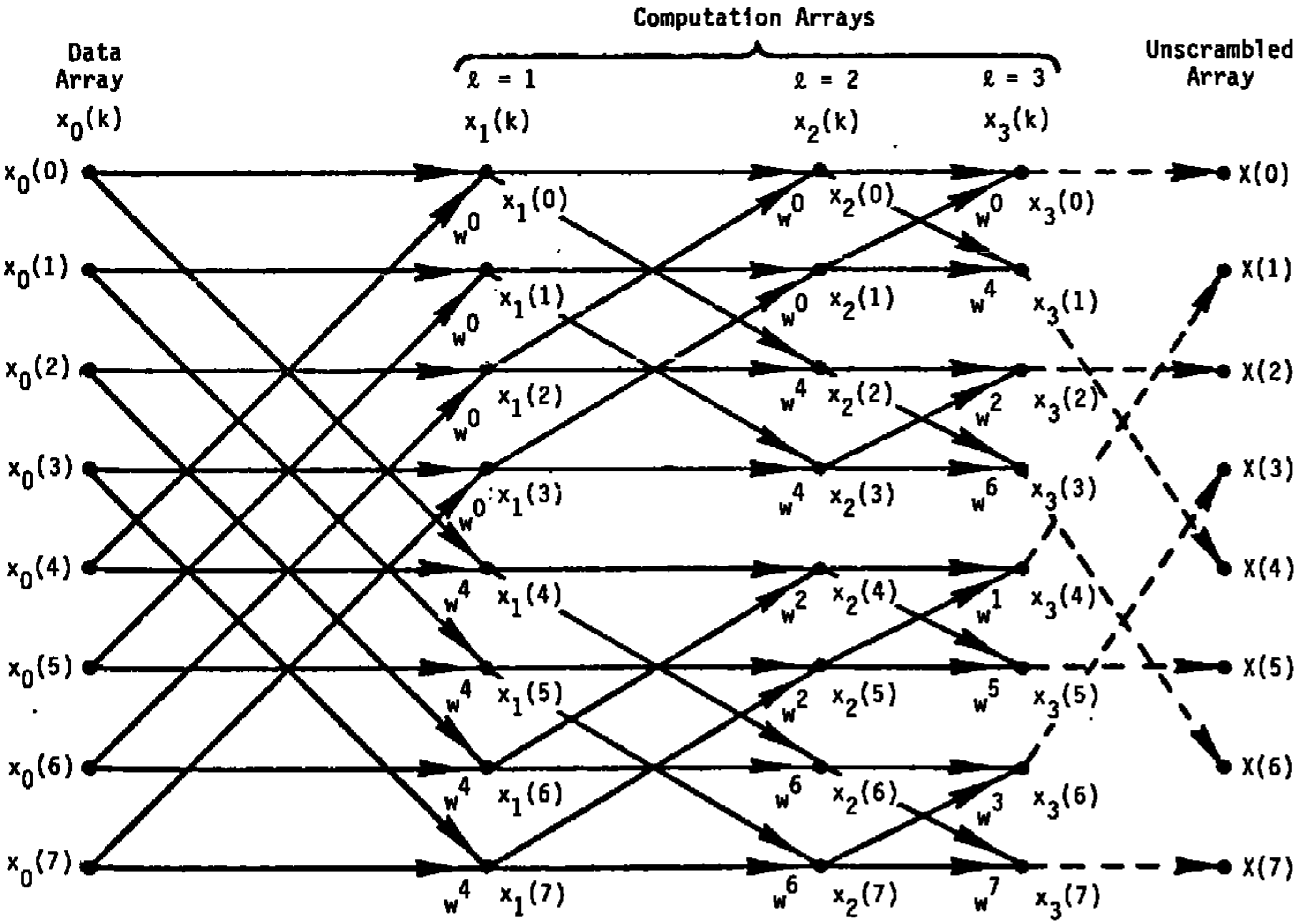


**Figure 1. FFT signal flow graph for N = 8.**

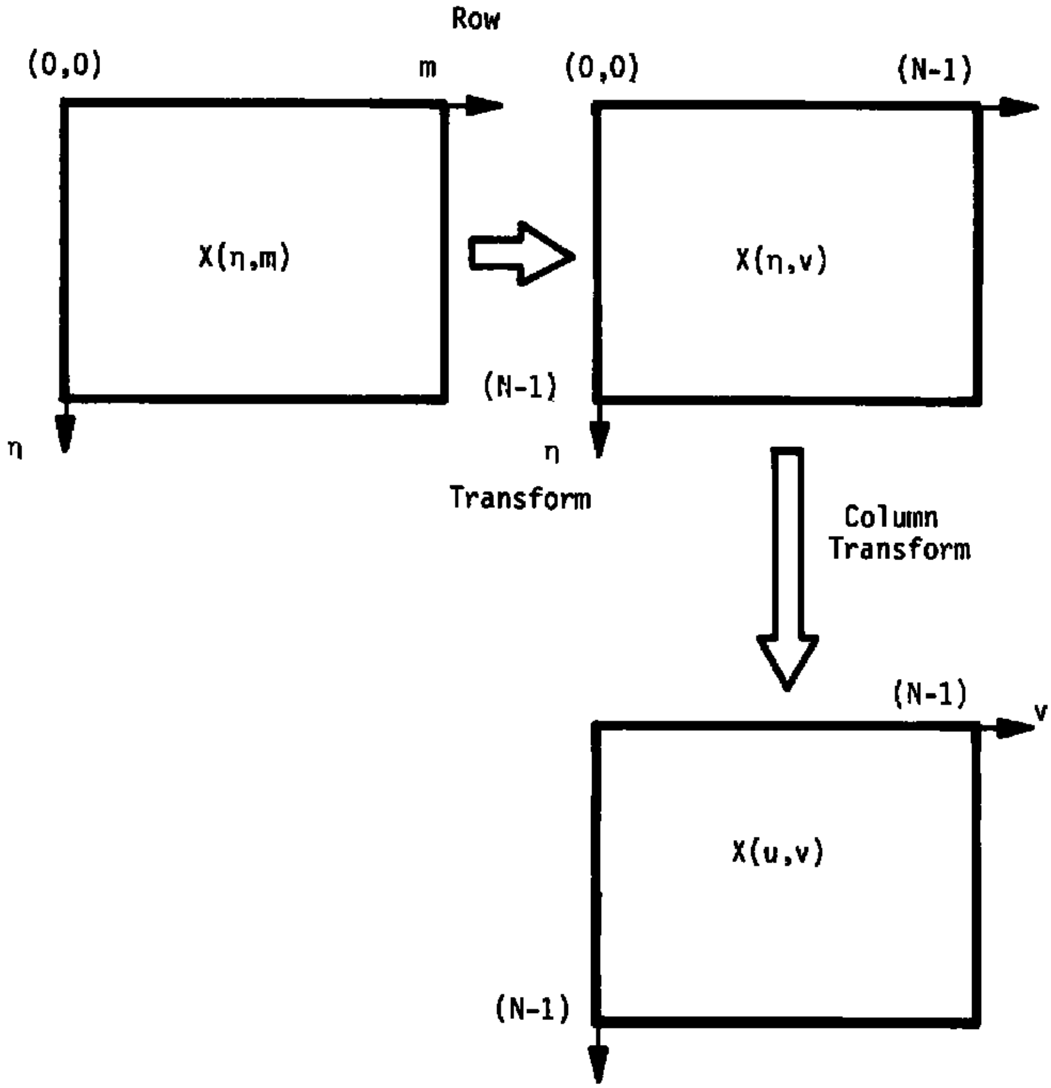
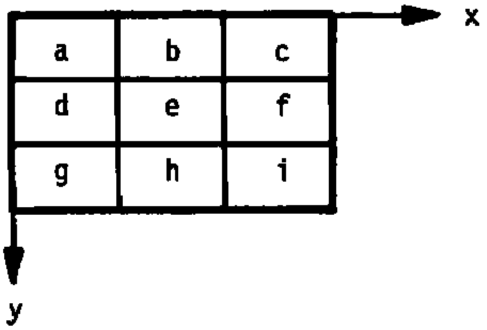


Figure 2. Diagram summarizing row-column method of performing 2-D FFT.



| | | |
|---|---|---|
| a | b | c |
| d | e | f |
| g | h | i |

**Figure 3. Three-by-three image region used for gradient operations.**

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

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

**Figure 4. Gradient (Sobel) templates.**

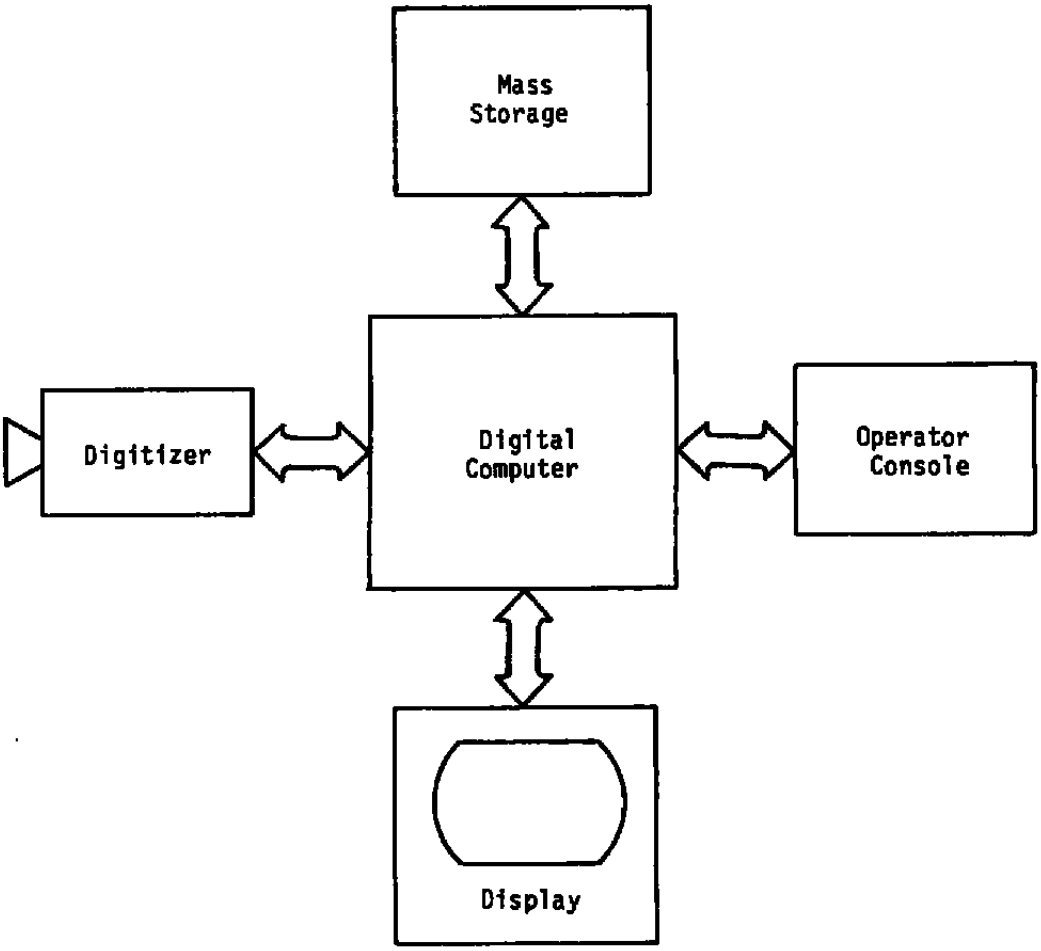


**Figure 5. Layout of basic image processing system.**

| | | | |
|---|---|---|---|
| X(1,479) | X(3,479) | ... | X(255,479) |
| X(1,477) | X(3,477) | ... | X(255,477) |
| ⋮ | ⋮ | ... | |
| X(1,1) | X(3,1) | ... | X(255,1) |

**Figure 6. Data array resulting from the decimation procedure.**

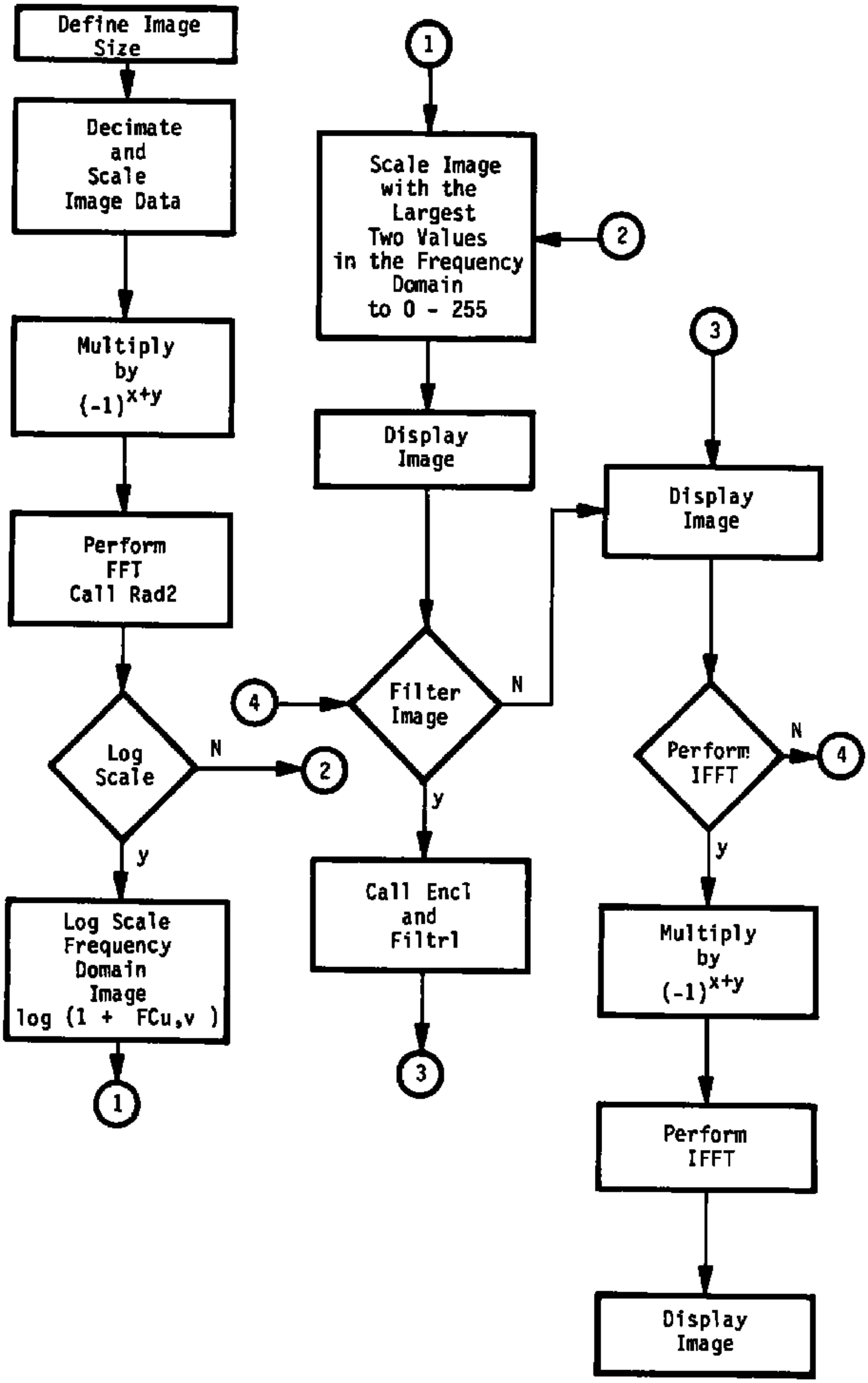


Figure 7. Flowchart of image enhancement procedure.

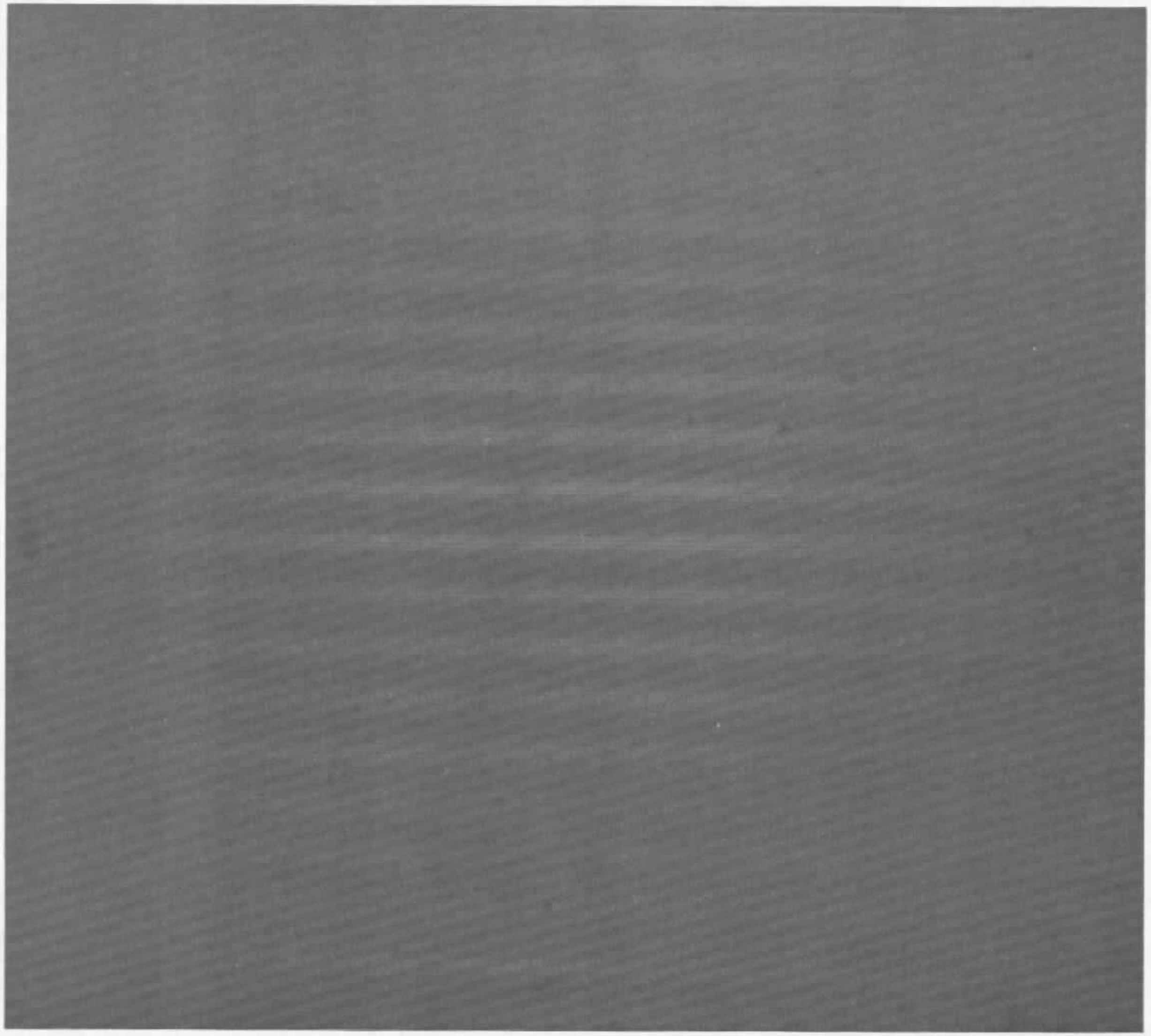


Figure 8. Noise-corrupted image prior to frame averaging.

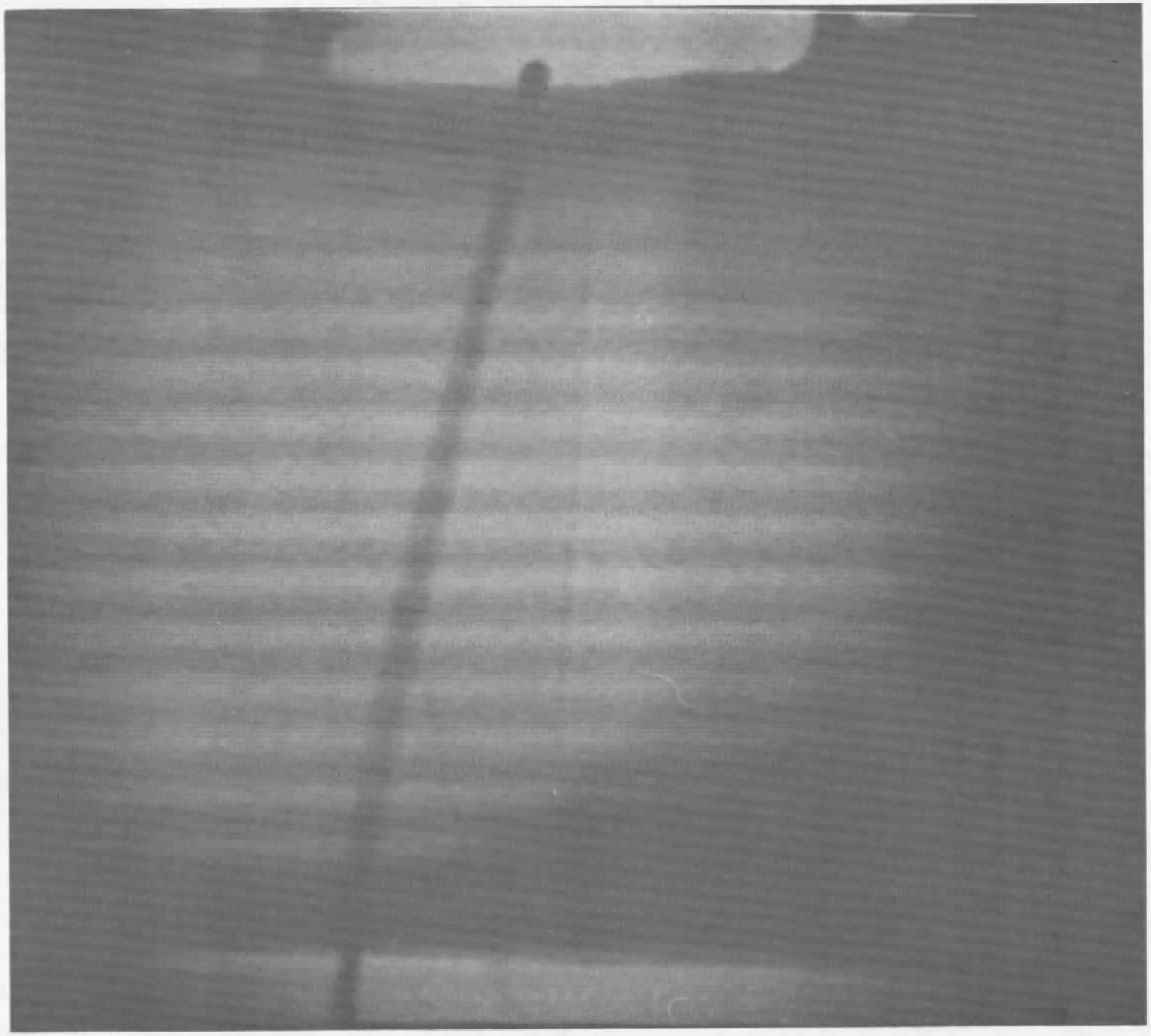


Figure 9. Image after frame averaging.

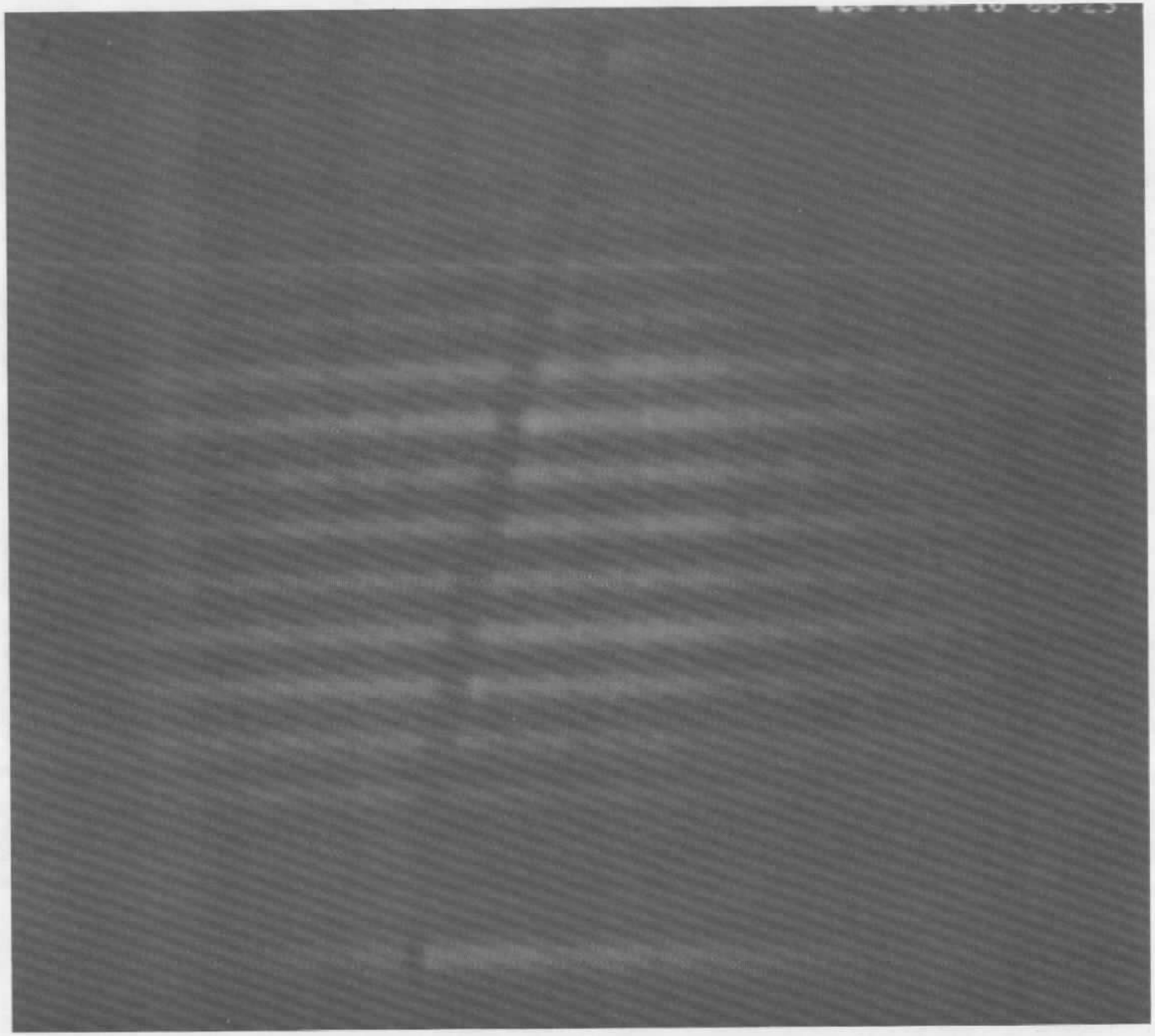


Figure 10. Image after average filtering.

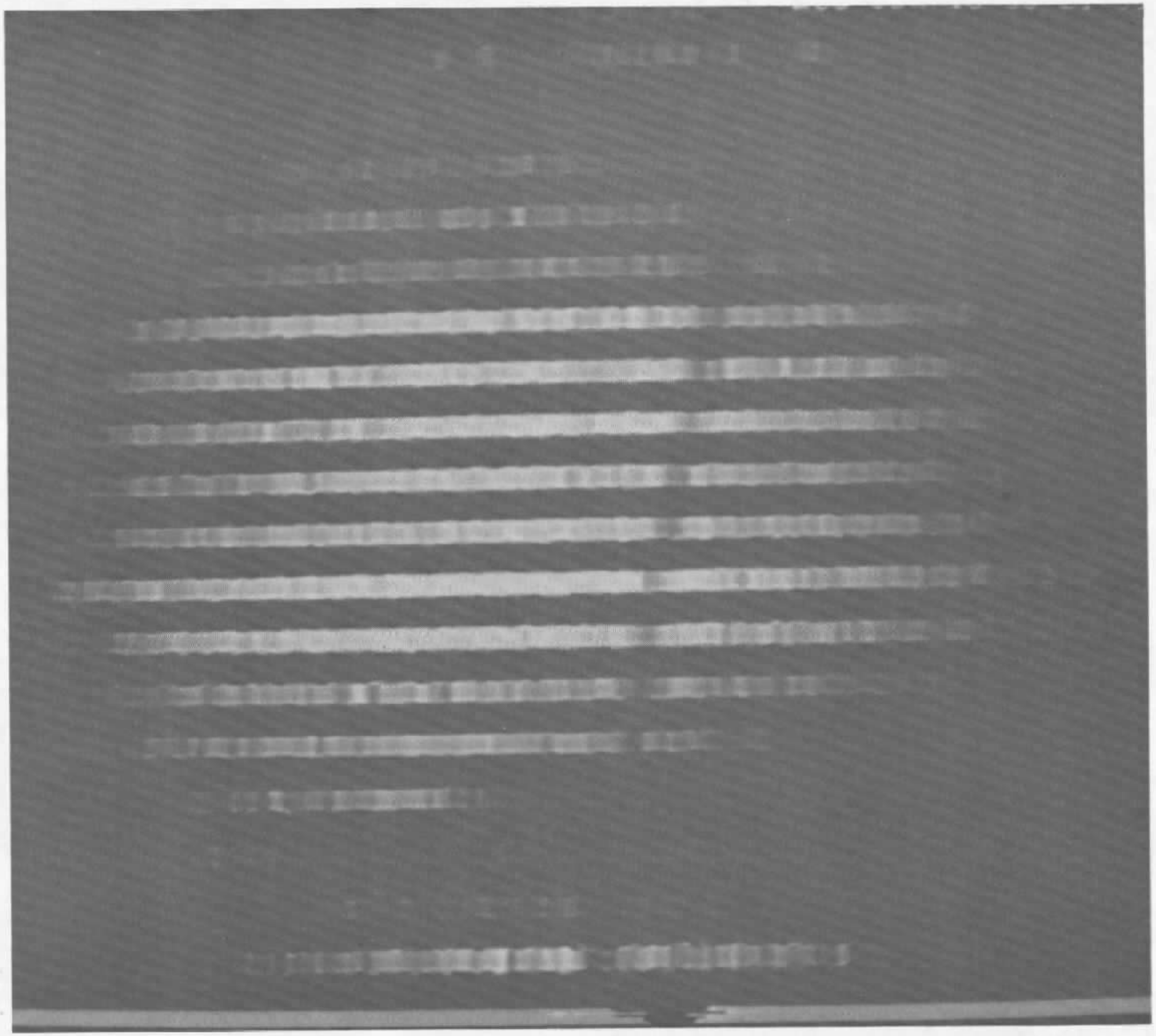
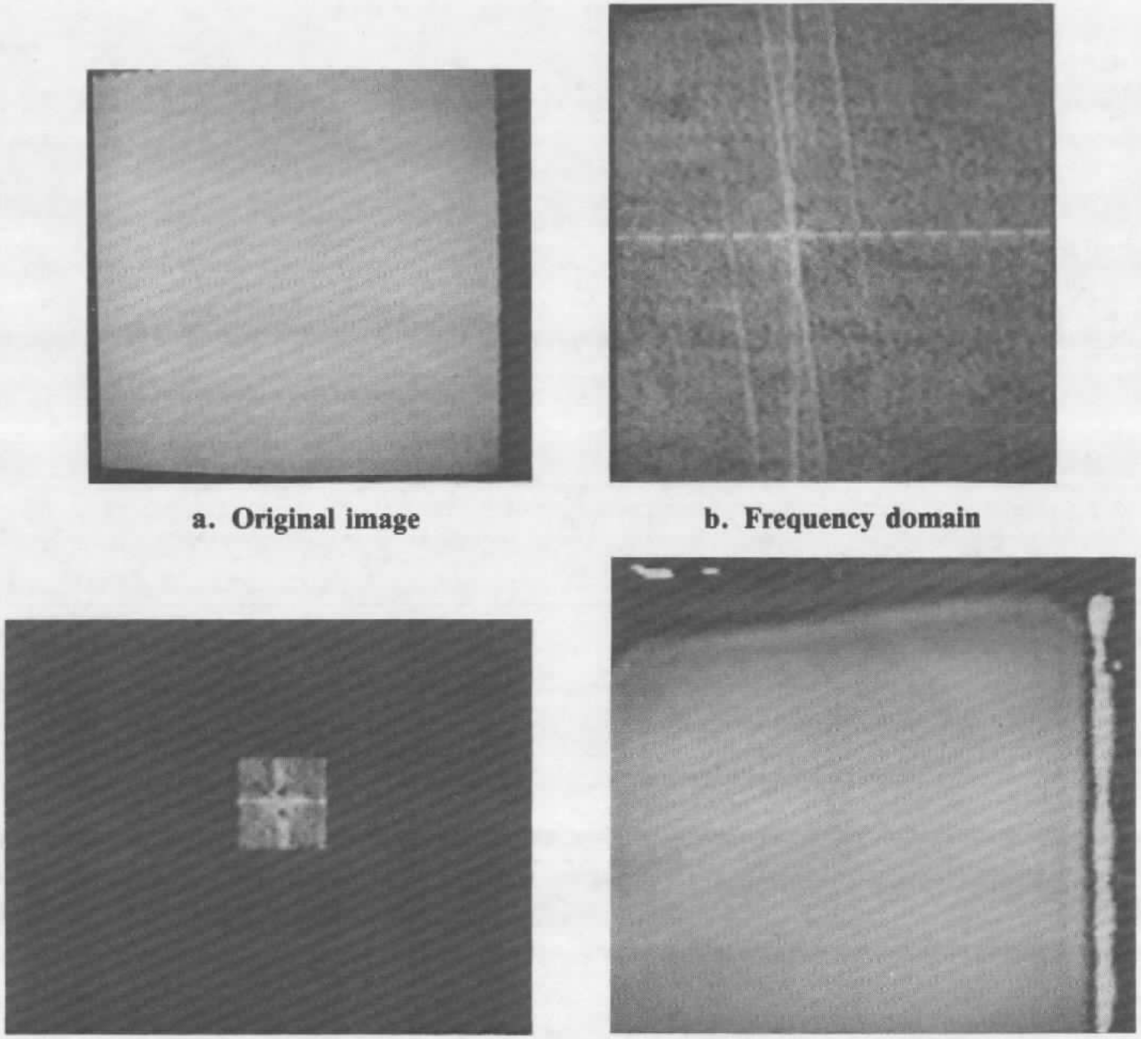


Figure 11. Image after median filtering.



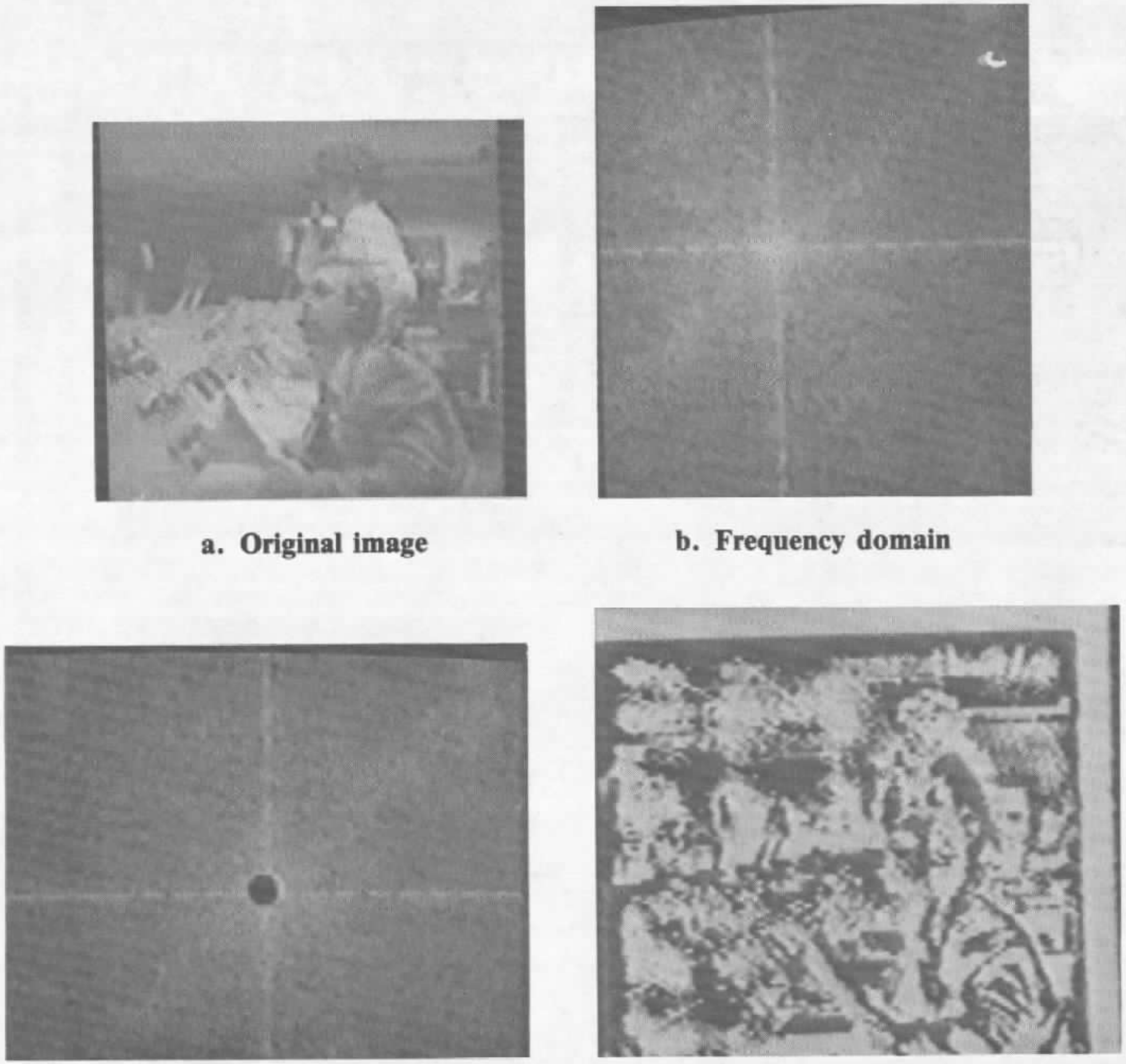
a. Original image

b. Frequency domain

c. Filtered frequency domain

d. Processed image

Figure 12. Image illustrating ideal low-pass filter technique.



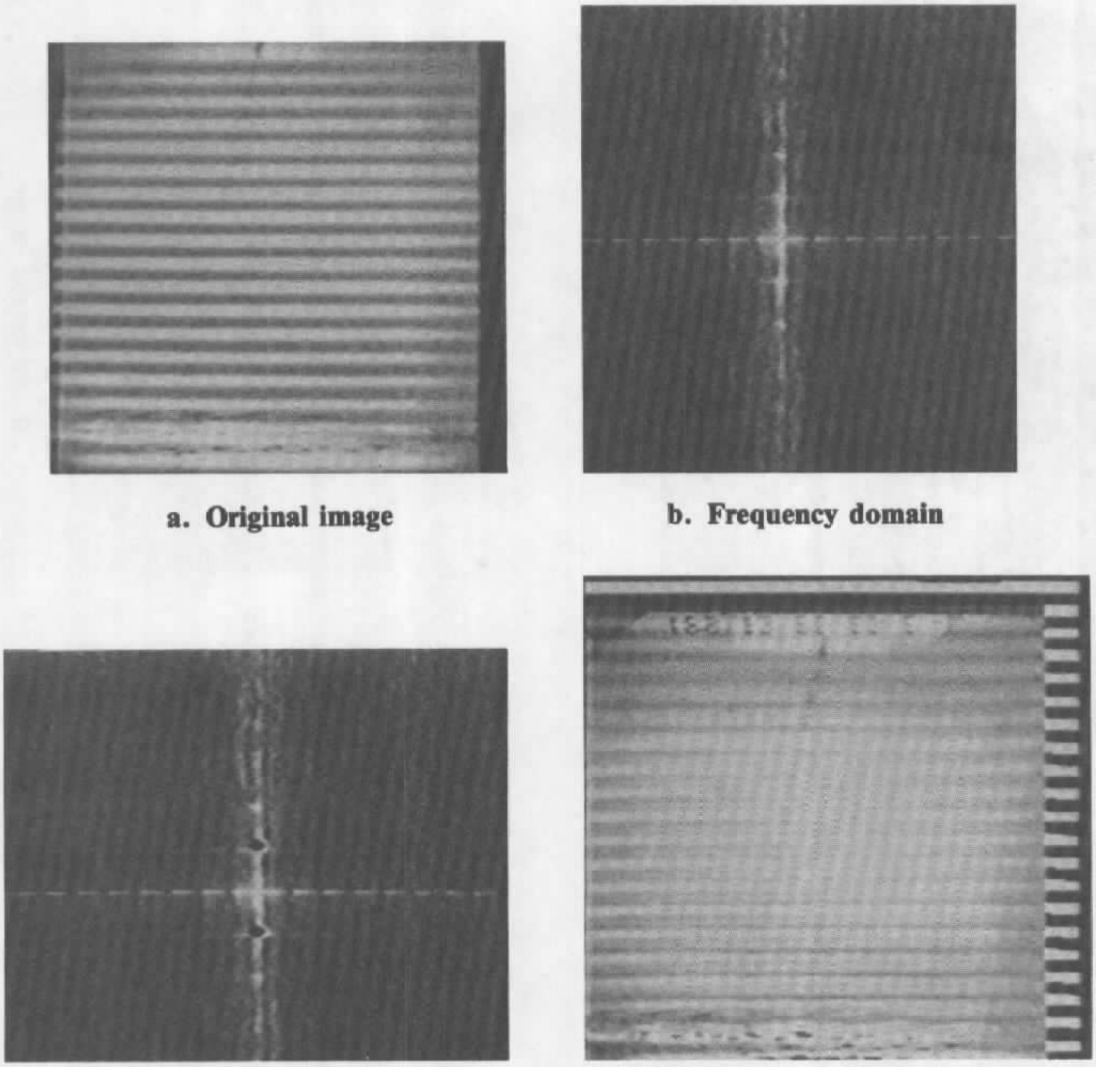
a. Original image

b. Frequency domain

c. Filtered frequency domain

d. Processed image

**Figure 13. Image illustrating ideal high-pass filtering technique.**



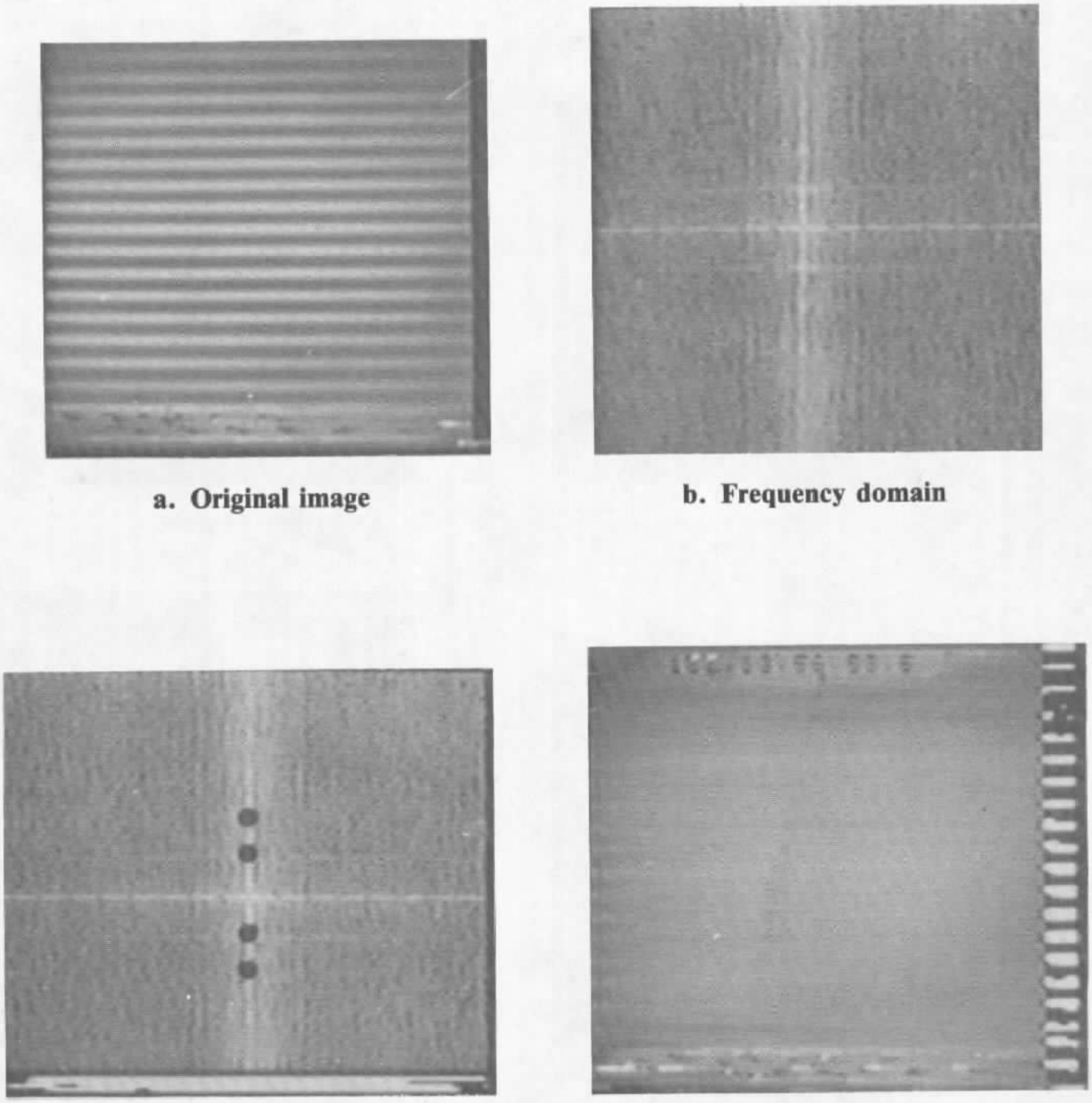
a. Original image

b. Frequency domain

d. Processed image

c. Filtered frequency domain

**Figure 14. Images illustrating the band-reject filtering technique and residual noise in resulting image caused by noise harmonics.**



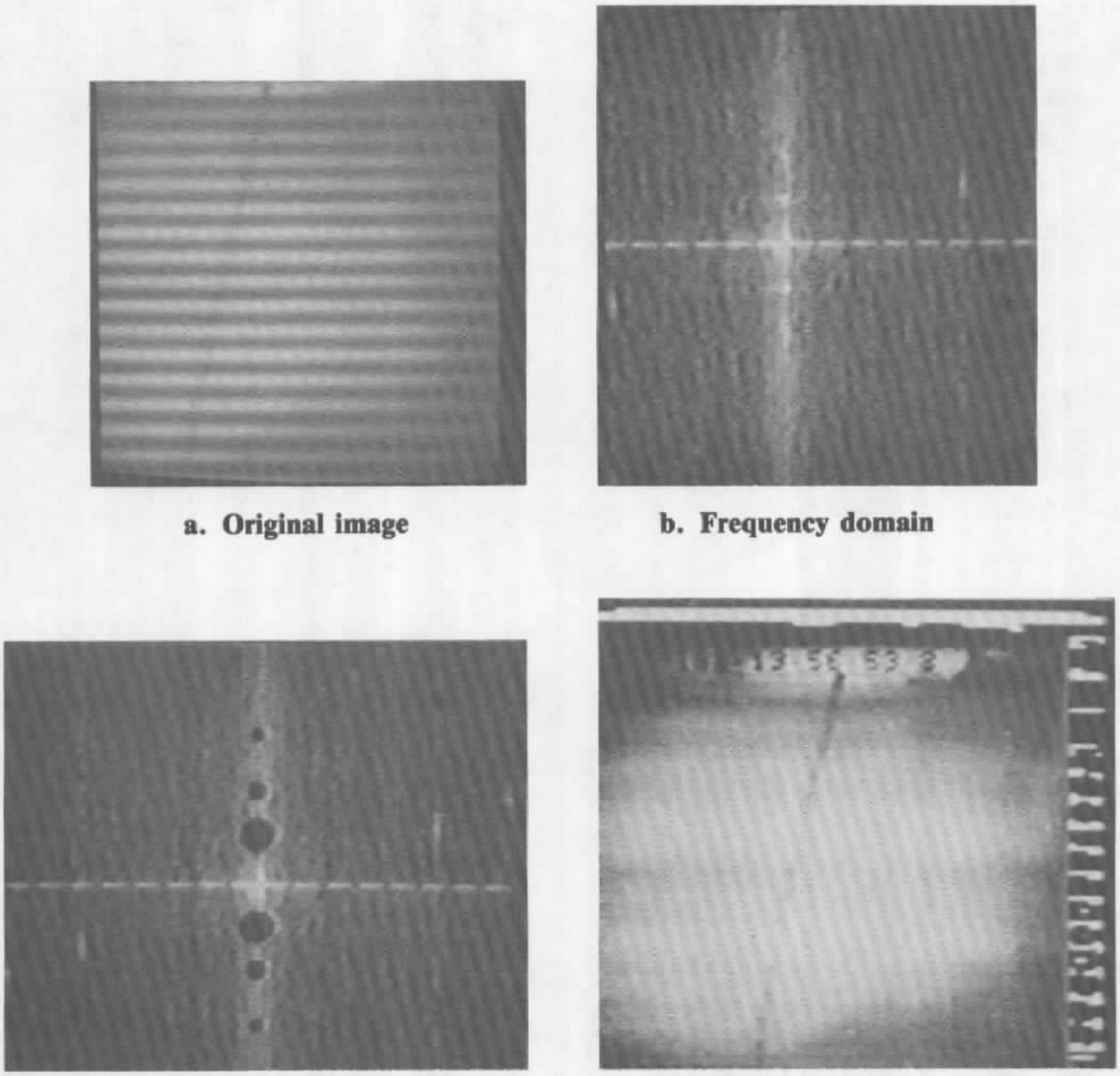
a. Original image

b. Frequency domain

c. Filtered frequency domain

d. Processed image

Figure 15. Images illustrating the band-reject filtering technique with second noise harmonics removed from frequency domain.



a. Original image

b. Frequency domain

c. Filtered frequency domain

d. Processed image

Figure 16. Images illustrating the band-reject filtering technique with second and third noise harmonics removed from frequency domain.

# APPENDIX A
# QUANTEX 9210 IMAGE PROCESSOR

The Quantex 9210 Image Processor at the AEDC was selected over other systems because it is a turn-key real-time imaging system with a sufficient amount of software support. The Quantex Image Processor was interfaced by the manufacturer to an IBM 9000 desktop computer. The host processor, a Motorola 68000 16/32-bit central processing unit, has a 16-bit data bus and a 32-bit internal architecture. The AEDC system was equipped with 4-Mbytes of RAM with expansion capabilities up to 16 Mbytes.

The image data and control signals for the imaging electronics are transmitted to system hardware via a Versabus. The Versabus and the systems internal structure are illustrated by the block diagram in Fig. A-1. The main components of the imaging system are the analog processor board and the memory/pipeline point processor board. The analog board consists of a flash A/D converter that converts an incoming video signal to an 8-bit word at a 30-Mhz sampling rate. This rate is sufficient to provide 640 samples per horizontal line, which, for the Quantex, equates to a 640-by-480 image.

The digitized A/D output is routed to the memory/processor board, which is a three-function device. It consists of a pipeline processor that can combine data in the onboard memory with new data arriving from the video input. In addition, it can modify data from either the video pipeline or memory, and it can also transmit the data to the video pipeline or memory unmodified. The memory on the memory/processor board is organized in a 640-by-480 picture element (pixel) array with each pixel address capable of storing a 12-bit value to prevent overflow when summing image arrays. The memory has three ports that can access the Versabus. One port is used for reading data in synchronism with the analog board. The second port reads the memory asynchronously. The third port writes data back to memory from the pipeline processor at the same rate as the data arriving from the video pipeline. The memory/processor board also serves as a memory controller that controls the various read and write operations performed by the system. The memory controller supports additional functions of the image processing system, such as scrolling data, roam, and zoom. When special analysis, such as Sobel transforms or median filtering of image data, is to be performed, the sampled image stored in the video memory is transferred in block form over the Versabus to a separate memory board directly addressable by the CPU.

In addition to average, median filtering, and Sobel transforms, the image processor can also perform the following processes in real-time:

1. store output

2. subtraction and differencing

3. look-up table and operations

The Quantex 9210 is also equipped with a second memory/processor board that allows the user to perform subtraction or differencing operations on a image in the video pipeline and a spatial board that enhances the system overall analytical capabilities.

Recently, the Quantex 9210 was upgraded to an IBM PC/AT desk top computer as a host computer and a Quantex Coprocessor (QCP) image processing system (modified 9210 image processors hardware). The system is designed to support the QCP image processing operations and IBM PC/AT operations. This is desirable because the QX-7 image processor uses a Motorola 68020, a true 32-bit 16-Mhz CPU, for intensive computational analysis, where the IBM PC/AT uses an Intel 80286 CPU with an internal configuration designed around the PC bus. Although a less powerful processor, the PC/AT 16-bit system is capable of supporting the Unix operating system as well as MS-DOS.

The upgraded system is capable of supporting four image memory boards in conjunction with the existing two memory/processor boards. The upgraded system is also capable of supporting the Motorola 68881 floating point coprocessor for image processing applications if required.

The IBM 9000-based image processor operates under the IBM CSOS operating system. The real-time and off-line functions of the image processor were designed by the Quantex Corp. using the CSOS utilities. The operating system is a real-time, multitasking facility. It can support the following languages: Basic, Fortran, Pascal, and MC68000 macroassembler. The Quantex Corp. used the CSOS operating system and Fortran programming language to design special on- and off-line image processing routines into a program called ICOS. The program was written so that external Fortran programs could run sequentially with the image processing functions. A special link utility, called LNK, was provided to merge a user's program to the ICOS program.

The Quantex Corp. provided a number of interfaces to the image processor hardware through a Fortran object code file called DVPIFAC. The DVPIFAC is a subroutine library that can be merged with a user's program. Subroutines that are available for integrating Fortran programs with the image processor hardware include

DVPOPN(UNIT,STATUS)

DVPCLO(UNIT,STATUS)

DVPFUN(UNIT,FUNCTION LIST,STATUS)

DVPREA(UNIT,X1,Y1,X2,Y2,BUFFER,STATUS)

DVPWRI(UNIT,X1,Y1,X2,Y2,BUFFER,STATUS)

DVPLUT(UNIT,OFFSET,SIZE,BUFFER,STATUS)

DVPLUW(UNIT,OFFSET,SIZE,BUFFER,STATUS)

The "unit" specification in the subroutine call statements is used to identify whether the memory/processor board, analog board, or spatial board is being addressed by the Fortran program. The "status" specification in the subroutine call statements is an integer *4 variable that is defined in the user's program for passing error codes. The other items that are unique to certain subroutines willl be described as each subroutine is discussed.

The first and second subroutine calls DVPOPN and DVPCLO are similar to basic Fortran OPEN and CLOSE statements. These two subroutines are used to open and close the specific device being addressed by the user. The DVPFUN subroutine allows any type of function used by the ICOS program to be used by the user's program. By using the FUNCTION LIST variable of DVPFUN, a variety of operations normally performed by ICOS such as setting the analog or digital ramp, or initiating the "zoom" function, can be implemented in the user's program. The DVPREA and DVPWRI subroutines are designed to allow user access to image memory. The X1,Y1,X2,Y2 variables in the variable list are designed to pass user-specified coordinates to the subroutines, so that only a certain predetermined region of image data need be read or written into system memory. The variable BUFFER is an integer array defined by the user to pass image data. The DVPLUT subroutine allows the user to write to the look-up table (LUT) on a specified memory/processor board. The LUT is setup as 4,096 words that are 12 bits long. This subroutine loads the LUT as if it were only 256 words, 8-bits long. The OFFSET variable defines the location to start writing to the LUT. The SIZE variable defines the number of LUT locations that are to be written. The look-up table in the image processor is set up as a write-only device and, therefore, DVPREA cannot be used with the LUT. The DVPLUW is similar to the DVPLUT except that DVPLUW enables the user to write to all 4,096 locations of the memory/processor board look-up table.

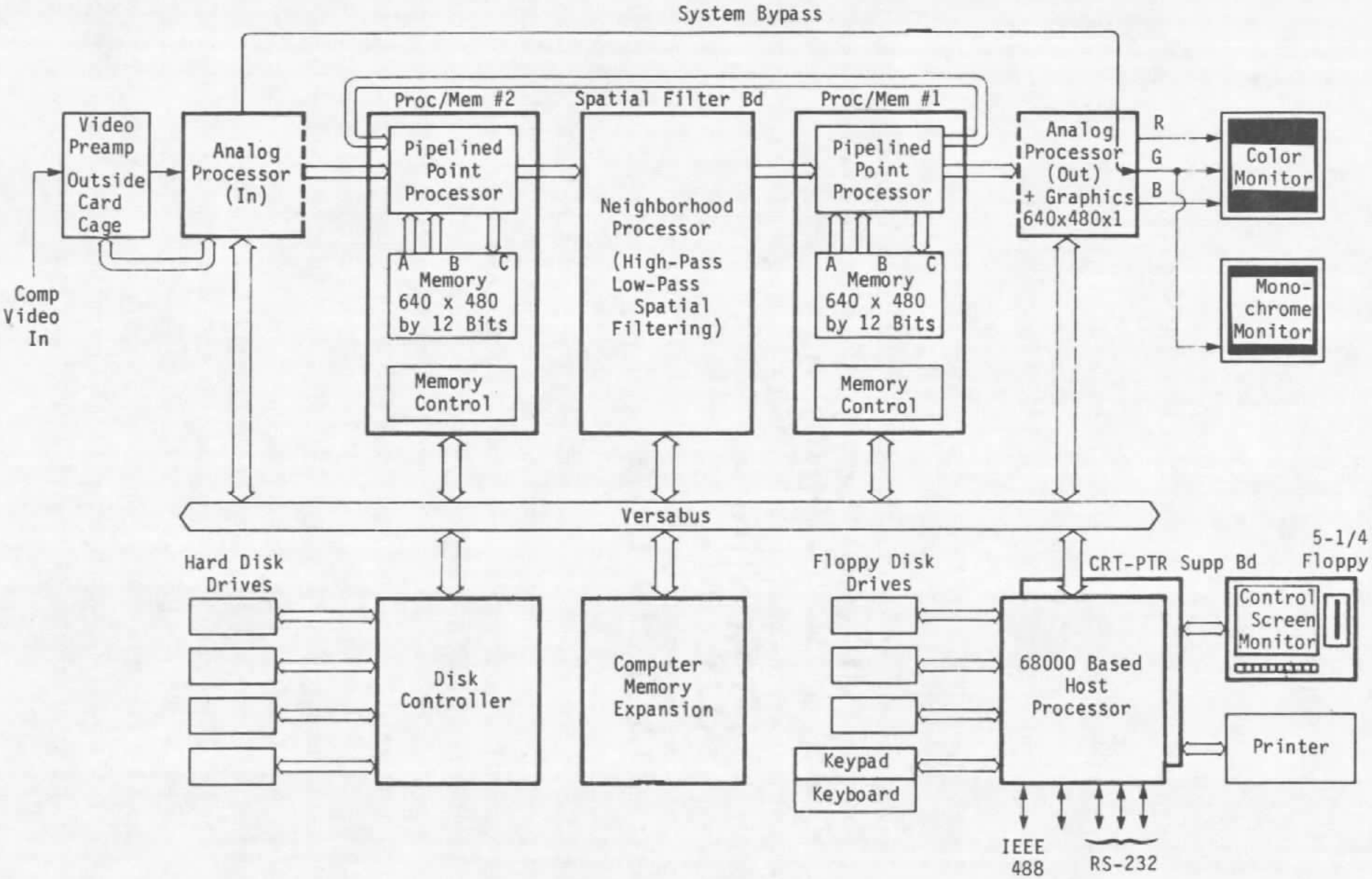


Figure A-1. QX-9210 system block diagram.

## NOMENCLATURE

| | |
|---|---|
| $D_0$ | Fourier domain filter cutoff |
| $D(u,v)$ | Fourier domain filter parameter |
| $E[x]$ | Expected value |
| $f(x,y)$ | 2-D image |
| $F(u,v)$ | Fourier transform of $f(x,y)$ |
| $f_N$ | Frequency of the noise |
| $G_x$ | Gradient in the x direction |
| $G_y$ | Gradient in the y direction |
| $g(x,y)$ | Noisy or processed 2-D image |
| $G(u,v)$ | Fourier transform of $g(x,y)$ |
| $\bar{g}(x,y)$ | Averaged image |
| $h(x,y)$ | Spatial domain filter kernel |
| $H(u,v)$ | Fourier transform of $h(x,y)$ |
| $m'$ | The number of multiplications in a Cooley-Tukey radix-2FFT |
| $M_i$ | Short-length discrete Fourier transforms used in the prime factor FFT |
| $m_p$ | The number of multiplications in a prime factor FFT |
| $N$ | The data sequence length for an FFT |
| $N_p$ | Periodicity of the noise |
| $n_p$ | The distance of the noise from the center of the Fourier domain image |

| | |
|---|---|
| $(n,m)$ | Coordinates n and m in the spatial domain |
| $X(n,m)$ | 2-D spatial domain signal |
| $X(u,v)$ | Fourier transform of a 2-D signal |
| $\hat{x}(n,v)$ | Fourier transform along column data |
| $\alpha_i$ | Number of additions in a short-length discrete Fourier transform |
| $\alpha'$ | The number of additions in a Cooley-Tukey radix-2 FFT |
| $\alpha_p$ | The number of additions in a prime factor FFT |
| $\Delta x$ | Sample interval in the spatial domain |
| $\Delta u$ | Sample interval in the Fourier domain |
| $\Sigma$ | Summation |
| $\epsilon$ | In the set of |
| $\eta(x,y)$ | Additive zero mean noise |
| sinc(u) | The sinc function |
| $\mu_i$ | Number of multiplications in a short-length discrete Fourier transform |
| $\sigma_{\bar{g}}\ (x,y)$ | Standard deviation of the averaged image |
| $\sigma_\eta^2(x,y)$ | Variance of the noise |
| $\sigma_{\bar{g}}^2(x,y)$ | Variance of the averaged image |
| $\sigma_\eta(x,y)$ | Standard deviation of the noise |