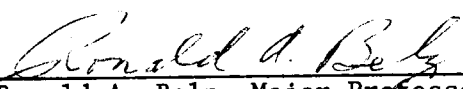


To the Graduate Council:

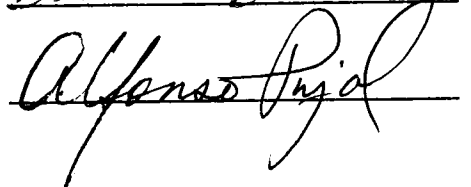
I am submitting herewith a thesis written by David L. Carson entitled "A Comparison of 2D Fourier and Spatial Techniques For Removing Noise in Real Time X-Ray Radiography." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.



Ronald A. Belz, Major Professor

We have read this thesis and
recommend its acceptance:





Accepted for the Council:



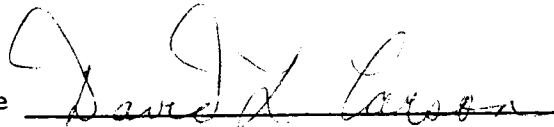
Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature



Date

A COMPARISON OF 2D FOURIER AND SPATIAL TECHNIQUES
FOR REMOVING NOISE IN REAL TIME X-RAY
RADIOGRAPHY

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

David L. Carson
December 1987

ACKNOWLEDGMENTS

The work reported herein was supported by the Arnold Engineering Development Center.

The author wishes to express his sincere gratitude and appreciation to the many people who have offered assistance during the course of this thesis. Special thanks are due to the major advisor, Dr. Ronald Belz, for his invaluable suggestions, guidance and support. Appreciation is also expressed to the committee members, Dr. Alfonso Pujol and Dr. Bruce Bomar, for their assistance and suggestions.

Finally, the author expresses his gratitude and appreciation to his parents, Mr and Mrs Phillip L. Carson and the rest of the Carson family who supported the author during the preparation of this thesis.

ABSTRACT

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 vibration due to the rocket motor 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 by means of frame averaging destroys the time resolution of the image.

TABLE OF CONTENTS

CHAPTER	PAGE
I. INTRODUCTION	1
II. TECHNIQUE ANALYSIS	4
2.1 FFT Algorithm Analysis	4
2.1.1 The Prime Factor Algorithm	5
2.1.2 Mixed Radix	7
2.1.3 The Fixed Radix-2 FFT	8
2.1.4 The Radix-4 FFT	10
2.2 2D FFT's	10
2.2.1 Row-Column 2D FFT	10
2.3 Spatial Domain Techniques	12
2.3.1 Frame Averaging	12
2.3.2 Neighborhood Averaging	14
2.3.3 Sobel Operators	15
2.4 Frequency Domain Techniques	17
2.4.1 Low-Pass Filtering	19
2.4.2 High-Pass Filtering	21
2.4.3 Band-Reject Filtering	22
III. IMAGING PROCESSING HARDWARE AND SOFTWARE	24
3.1 Image Processor Hardware	24
3.2 Software for Frequency Domain Processing	26
IV. RESULTS	34
V. SUMMARY AND CONCLUSIONS	47
REFERENCES	50
APPENDIX	53
VITA	60

LIST OF FIGURES

FIGURE	PAGE
1. FFT signal flow graph for $N = 8$	9
2. Diagram summarizing row-column method of performing 2D FFT	13
3. 3x3 Image region used for gradient operations	16
4. Gradient (Sobel) Templates	18
5. Layout of basic image processing system	25
6. Data array resulting from the Decimation Procedure	28
7. Flowchart of image enhancement procedure	33
8. Noise corrupted image prior to frame averaging	36
9. Image after frame averaging	37
10. Image after average filtering	38
11. Image after median filtering	39
12. Image illustrating ideal low-pass filtering technique	41
13. Image illustrating ideal high-pass filtering technique	42
14. Images illustrating the band-reject filtering technique and residual noise in resulting image due to noise harmonics	43
15. Images illustrating the band-reject filtering technique with second noise harmonics removed from frequency domain	45
16. Images illustrating the band-reject filtering technique with second and third noise harmonics from frequency domain	46
A.1. QX-9210 System Block Diagram	55

LIST OF SYMBOLS

D_0	Fourier domain filter cutoff
$D(u,v)$	Fourier domain filter parameter
$E[x]$	Expected value
$f(x,y)$	2D 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 2D 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 multiplies in a Cooley-Tukey radix-2 FFT
M_i	Short Length Discrete Fourier Transforms used in the Prime Factor FFT
m_p	The number of multiplies in a Prime Factor FFT
N	The data sequence length for a 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
$X(n,m)$	2D spatial domain signal
$X(u,v)$	Fourier transform of a 2D signal
$\hat{x}(n,v)$	Fourier transform along column data

α_i	Number of additions in a short length Discrete Fourier Transform
α'	The number of additions in a Cooley-Tukey radix-2 FFT
α_p	The number of additions in a Prime Factor FFT
Δx	Sample interval in the spatial domain
Δu	Sample interval in the Fourier domain
Σ	Summation
ϵ	In the set of
$\eta(x,y)$	Additive zero mean noise
$\text{sinc}(u)$	The sinc function
μ_i	Number of multiplies in a short length Discrete Fourier Transform
$\sigma_g(x,y)$	Standard deviation of the averaged image
$\sigma_\eta^2(x,y)$	Variance of the noise
$\sigma_g^2(x,y)$	Variance of the averaged image
$\sigma_\eta(x,y)$	Standard deviation of the noise

CHAPTER I

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 rocket motor (SRM)'s components such as rocket nozzles, propellant, and the insulator and case interface. In addition propellant burn characteristics and slag formation and other anomalies have been identified during motor firing (1)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 which is 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 Million electron volt (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

¹Numbers in parentheses refer to similarly numbered references in the list of references.

processing hardware were written for spatial domain processing, these processing techniques provided limited capability relative to noise suppression. Although the resulting images were enhanced they still contained a certain amount of noise. Thus the spatial techniques could not completely eliminate the noise. Likewise, the extensive use of spatial techniques such as frame averaging created blurred images with poor time resolution.

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. The study 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 Chapter II, a literature survey is presented of the various image analysis techniques used for enhancing image data. FFT algorithm analysis, Fourier domain techniques, and spatial domain techniques are discussed. Chapter III describes the hardware and software used in the implementation of the frequency domain techniques for image enhancement. In Chapter IV, the results obtained by the frequency domain processing

technique are presented and compared to those in the spatial domain. In Chapter V, a summary and conclusion of the study is presented.

CHAPTER II

TECHNIQUE ANALYSIS

This chapter presents a survey of the literature pertaining to various FFT algorithms available for implementation of Fourier domain processing. The advantages and disadvantages of the various one-dimensional (1D) FFT's are discussed. A discussion of the two-dimensional (2D) FFT is presented with emphasis on implementing the row-column method with a radix-2 1D FFT. The Fourier domain filters analyzed in this paper are presented with special emphasis given to the band-reject filter. Finally, a discussion on the spatial domain techniques used for image enhancement is given.

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 (2), and J.D. Blaken (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 is based on the on 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 reference (4) and references (5)-(8) and will not be discussed here. Kolba and Parks (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

$$m' = 3 \left[\frac{N}{2} \log N - 3 \frac{N}{2} + 2 \right] \quad (2.1)$$

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

$$\alpha' = 2N \log N + 5(m') \quad (2.2)$$

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

$$m_p = 2 \left[M_2 M_3 \mu_1 + M_1 M_3 \mu_2 + M_1 M_2 \mu_3 \right] \quad (2.3)$$

where the M_i 's are the length of the short Discrete Fourier Transforms (DFT's) and μ_i 's are the total number of multiplies for length M_i DFT's. For a Prime Factor FFT of length 504, the short length DFT's M_1 , M_2 , and M_3 , are equal to 9, 7, and 8, respectively the number of multiplies μ_i 's for each DFT are 10, 8, and 2, respectively and the total multiplication count is equal to 2524 Ref. (4). The number of additions equals

$$\alpha_p = 2 \left[M_2 M_3 \alpha_1 + M_1 M_3 \alpha_2 + M_1 M_2 \alpha_3 \right] \quad (2.4)$$

where the α_i 's are the number of additions for length M_i DFT's. For a length 504 FFT the α_i 's are 42, 36, and 26 respectively and the total number of additions is equal to 13164. The total operation count for the PFA is the combined total of the number of additions and multiplications, which is 15688. By using equations 2.1 and 2.2 the operations count for a roughly equivalent 512 length N Cooley-Tukey FFT is determined to be 36900. It can be concluded from the total operations count and the number of multiplies for the PFA that a computer that performs multiplications efficiently is better suited for performing the FFT using the PFA.

From equations 2.3 and 2.4 it was determined, by Kolba and Parks, that the prime factor algorithm computed the DFT approximately 50 percent faster than Singletons mixed radix FFT (6). This indicated that the prime factor algorithm 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 prime factor algorithm 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

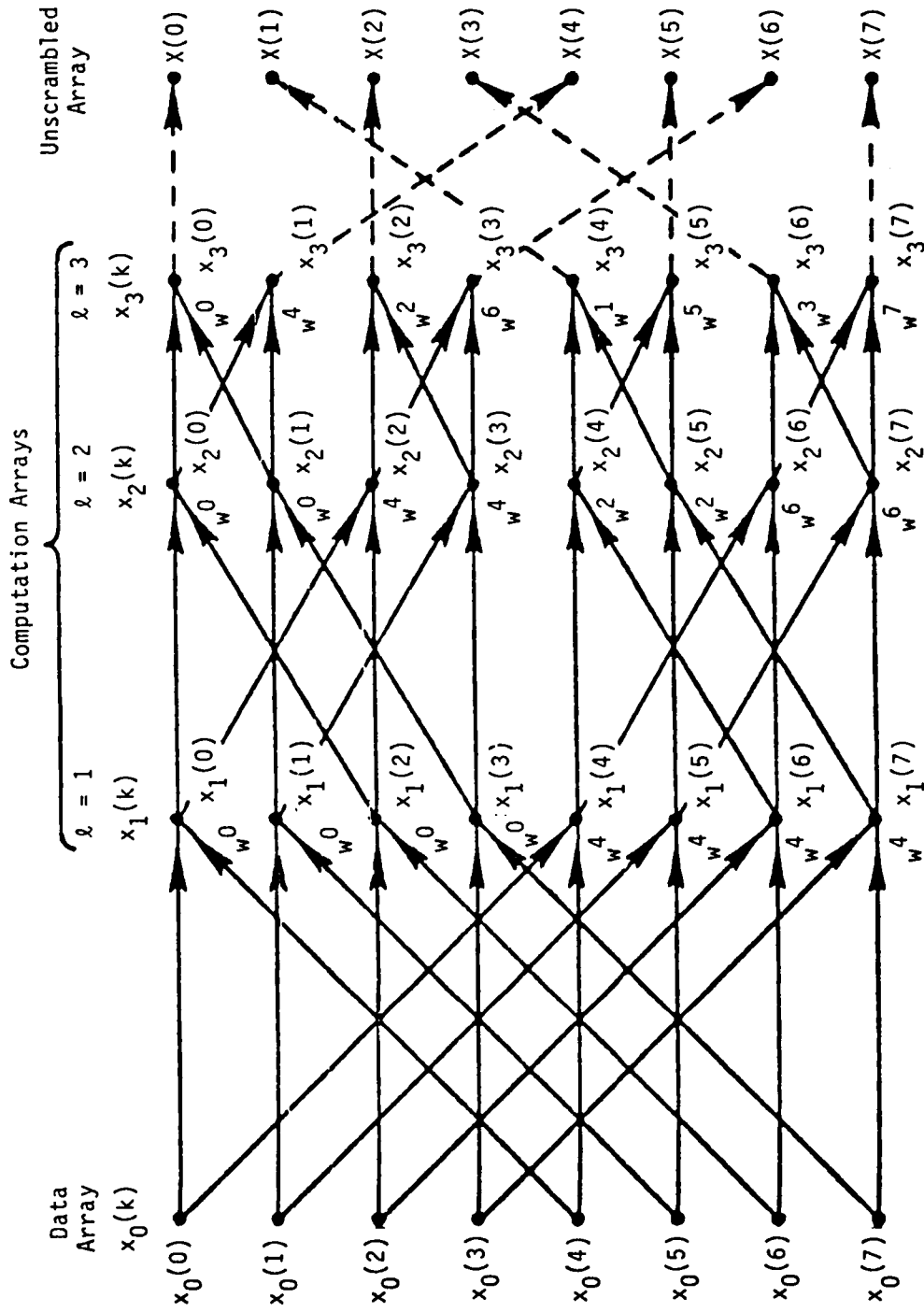
The mixed radix FFT is called a decimation in frequency algorithm since it decomposes the frequency parameter instead of the time or spatial parameter. The algorithm decomposes the data set into subsets of odd prime integers and even prime integers. Then, appropriate short transforms are performed. The Singleton (9) mixed radix is based on the method proposed by Cooley and Tukey (10). The sequence length N is factored into m different factors. The transform is then decomposed into m steps with N/n_i transformations of size n_i . The transform is computed such that the output array is in a scrambled format, and a procedure is performed to rearrange the results in order. In (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 (11). It has a total operations count that is the lowest known (4). A complex four butterfly split-radix FFT requires $N(M-1) + 4$ real multiplications and $3N(M-1) + 4$ real additions (12) 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 (5). However the main disadvantage in the implementation of the mixed radix FFT is the availability of enough memory to compute the small multi-variate DFT's in addition to the fact that it is not as fast as the radix-2 on systems with floating point software (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 transform, uses a divide and conquer scheme to compute the Discrete Fourier Transform (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 (4),(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 $n \log n$ (13). Because of its structure the radix-2 is a relatively efficient algorithm for computer architectures which use floating point software to perform mathematical computations (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 1D transform technique to be used on the 2D data processed in this paper.

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

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 (2)-(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 processing data.

2.2 2D FFT's

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

2.2.1 Row-Column 2D FFT

The row-column approach to performing the FFT is the most common in practice today for processing two dimensional signals. This is the approach described in Gonzales (14), and was used to generate frequency domain information for the images in this thesis.

The discrete Fourier transform for a square $N \times N$ 2D signal can be expressed as

$$X(u,v) = \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} x(n,m) \exp\left(-j2\pi \frac{un+vm}{N}\right) \quad (2.5)$$

for $u,v = 0,1,\dots,N-1$. Equation 2-5 can be expressed in separable form as

$$X(u,v) = \frac{1}{N} \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi un}{N}\right) \sum_{m=0}^{N-1} x(n,m) \exp\left(\frac{-j2\pi vm}{N}\right) \quad (2.6)$$

for $u,v = 0,1,\dots,N-1$. The principle advantage in writing the DFT in separable form is that successive application of the one dimensional transform will give $X(u,v)$. This becomes evident if equation 2-6 is expressed as

$$X(u,v) = \frac{1}{N} \sum_{n=0}^{N-1} \hat{x}(n,v) \exp\left(\frac{-j2\pi un}{N}\right) \quad (2.7)$$

where

$$\hat{x}(n,v) = N \left[\frac{1}{N} \sum_{m=0}^{N-1} x(n,m) \exp\left(\frac{-j2\pi vm}{N}\right) \right] \quad (2.8)$$

For each value of n the expression in the brackets is a one dimensional transform with frequency values $v=0,1,\dots,N-1$. Therefore the two

dimensional 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 2.8. The procedure is summarized in Fig. 2. By utilizing this technique and the radix-2 one dimensional FFT a two dimensional transform of image data can be performed.

2.3 Spatial Domain Techniques

In this section the spatial domain techniques most common in the enhancement of images are discussed. The theoretical concepts behind each technique are presented and follow closely the description found in (14).

2.3.1 Frame Averaging

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

$$g(x,y) = f(x,y) + \eta(x,y) \quad (2.9)$$

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

$$\bar{g}(x,y) = \frac{1}{M} \sum_{i=1}^M g_i(x,y) \quad (2.10)$$

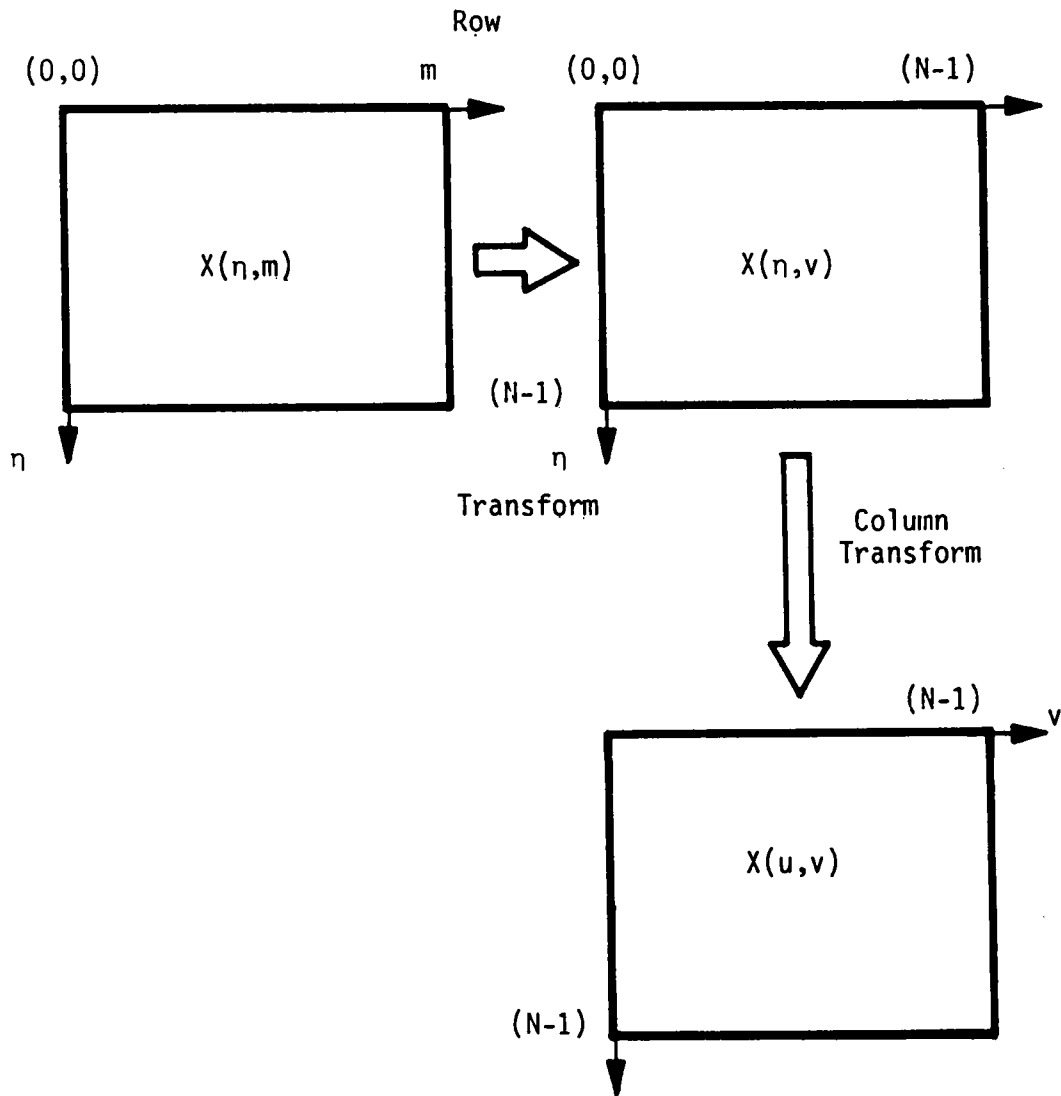


Figure 2. Diagram summarizing row-column method of performing 2D FFT

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

$$E[g(x,y)] = f(x,y) \quad (2.11)$$

and

$$\sigma_{\bar{g}}^2(x,y) = \frac{1}{M} \sigma_n^2(x,y) \quad (2.12)$$

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

$$\sigma_{\bar{g}}(x,y) = \frac{1}{\sqrt{M}} \sigma_n(x,y) \quad (2.13)$$

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

$$g(x,y) = \frac{1}{M} \sum_{(n,m) \in S} f(n,m) \quad (2.14)$$

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, which is based on taking the gradient of the image. Given some 3×3 region as in Fig. 3, the pixel at e with a new value defined as

$$e \leftarrow G = \sqrt{G_x^2 + G_y^2} \quad (2.15)$$

where G_y is the gradient along the y direction given by

$$G_y = (g + 2h + i) - (a + 2b + c) \quad (2.16)$$

and G_x is the gradient along x

$$G_x = (c + 2f + i) - (a + 2d + g) \quad (2.17)$$

It is evident in equation 2.16 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. Similarly, G_x is an estimate of the

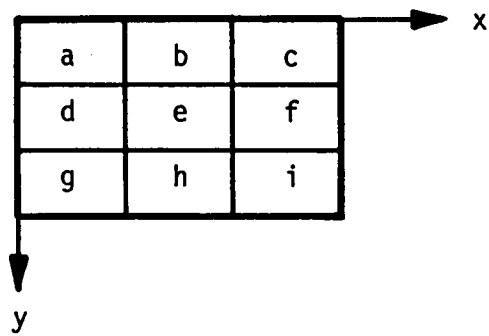


Figure 3. 3x3 Image region used for gradient operations

derivative in the x direction. Equations 2.15-2.17 can be implemented by using the two templates shown in Fig. 4, often referred to as Sobel operators. Implementation of equation 2.15 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 (14).

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

$$g(x,y) = f(x,y) * h(x,y) \quad (2.18)$$

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

$$G(u,v) = F(u,v)H(u,v) \quad (2.19)$$

This can be written succinctly as

$$f(x,y) * g(x,y) \leftrightarrow F(u,v)H(u,v) \quad (2.20)$$

which states that convolution in the spatial domain is equivalent to a multiplication in the frequency domain. The $H(u,v)$ term in equations 2.19 and 2.20 is typically referred to as the filter function in Fourier

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

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

Figure 4. Gradient (Sobel) templates

domain filtering applications. The $H(u,v)$ term in a Fourier domain filter is selected such 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 which 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 two dimensional ideal low pass filter has a transfer function which satisfies the following relationship.

$$H(u,v) = \begin{cases} 1, & \text{if } D(u,v) \leq D_0 \\ 0, & \text{if } D(u,v) > D_0 \end{cases} \quad (2.22)$$

where D_0 is a specified positive quantity, and $D(u,v)$ is the radial distance from the center of the frequency plane to the point (u,v) (14) and is given by equation 2.23.

$$D(u,v) = (u^2 + v^2)^{1/2} \quad (2.23)$$

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 due to the sharp cut-off of the ideal filter. This can be explained by referring to equation 2.19, which is the Fourier transform of equation 2.18. The spatial extent of $h(x,y)$ in equation 2.18 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 one dimensional sinc function is

$$\text{sinc}(u) = AX \left| \frac{\sin \pi u X}{\pi u X} \right| \quad (2.24)$$

By enlarging the bandwidth of the filter, the extent of the sinc function will become less. Therefore, the larger the "filter bandwidth" size, the less pronounced the ringing will be. The 2D circular ideal lowpass filter of 2.22 corresponds to a circularly symmetric Bessel function of the first kind of order 1 (15) 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 operation without the ringing is with an exponential low pass filter (ELPF). The ELPF with a cut-off frequency at a radial distance D_0 from the center of the frequency domain is given by equation 2.25, where n effects the rate of attenuation in the filter $H(u,v)$.

$$H(u,v) = \exp^{-[D(u,v)/D_0]^n} \quad (2.25)$$

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 lowpass 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 work. A more thorough discussion of the various types of low pass filters that are available can be found in (14) and (15).

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

$$H(u,v) = \begin{cases} 0, & \text{if } D(u,v) \leq D_0; \\ 1, & \text{if } D(u,v) > D_0 \end{cases} \quad (2.26)$$

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 reject filter is given by the relation

$$H(u,v) = \begin{cases} 0, & \text{if } D(u,v) \leq D_0; \\ 1, & \text{if } D(u,v) > D_0 \end{cases} \quad (2.27)$$

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

$$D(u,v) = [(u - u_0)^2 + (v - v_0)^2]^{1/2} \quad (2.28)$$

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 bandlimited 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 bandlimited noise is eliminated.

CHAPTER III

IMAGE PROCESSING HARDWARE AND SOFTWARE

In this chapter the image processing system used for implementing the FFT and the filtering routines is described. The manner in which image data are sampled and stored in system memory are discussed. The software that was used to integrate the FFT and filtering programs are also presented.

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 which is a high speed, flash, analog to digital converter and is typically interfaced to a host computer such as a DEC VAX or PC. A number of software support routines, such as 2D FFT's, histogram equalization, and NxN 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

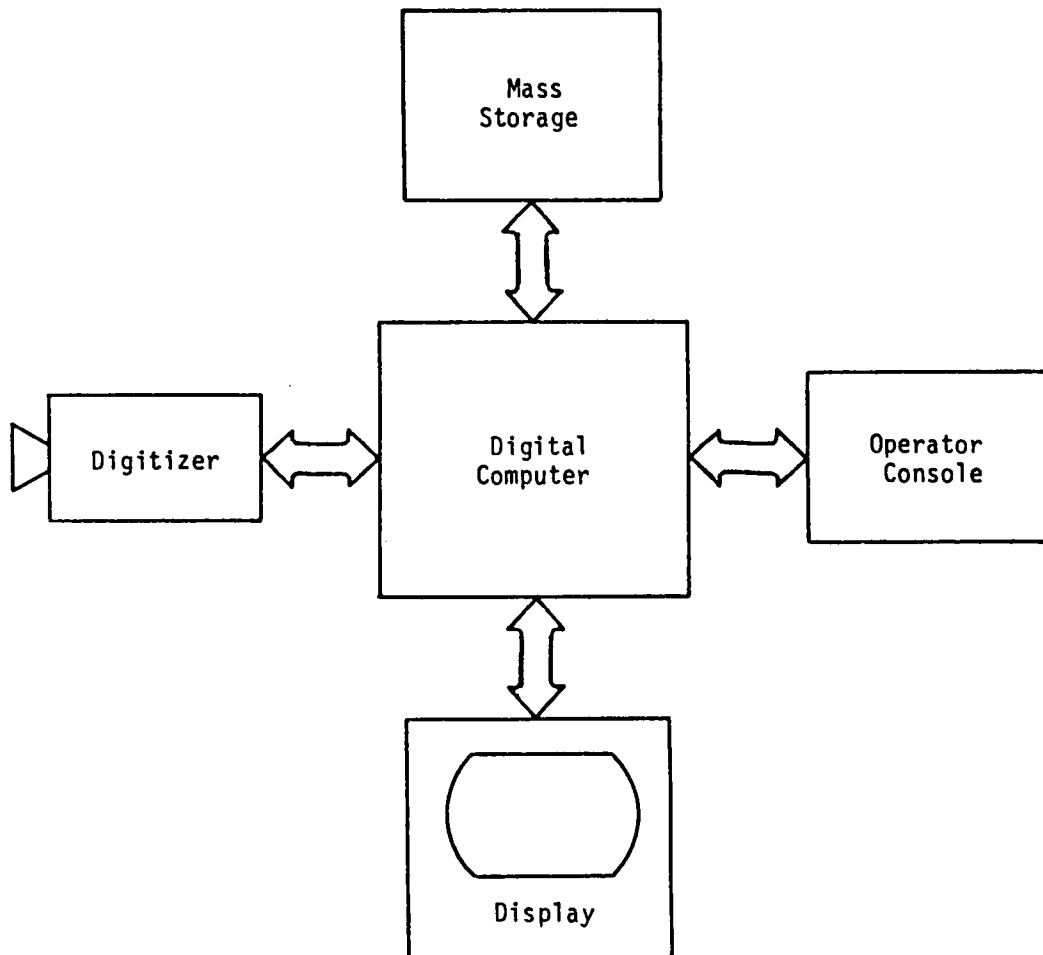


Figure 5. Layout of basic image processing system

different systems. The Quantex (see the appendix for detailed information), which is a real-time system with a pipeline processor, is able to perform real-time frame averaging, frame summing, high and low pass spatial filtering operations and pseudo coloring. 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 which 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 which integrates more enhancement techniques in a real-time environment using custom hardware. One dimensional Finite Impulse Response (FIR) filters, segmentation operations, (2D FFT's) and various other functions, 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

This section describes the programs that were written to perform the Fast Fourier Transform on image data and filtering operations on the transformed data. The program that performs the filtering operation utilizes 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 step in processing image data was to truncate the 640x480 image into a 480x480 image array, after which the data array was decimated to a 240x240 array and then zero padded to a 256x256 array which 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 on the computer disk. The decimation was performed by a program called "Half2" which read the image data by calling the Quantex subroutines (described in the appendix) DVPOP and DVPREA which opens the processor 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 DVPWR subroutine. The data array which results from the decimation process is shown in Fig. 6.

$X(1,479)$	$X(3,479)$	...	$X(255,479)$
$X(1,477)$	$X(3,477)$	...	$X(255,477)$
.	.		
.	.		
.	.	...	
$X(1,1)$	$X(3,1)$	...	$X(255,1)$

Figure 6. Data array resulting from the Decimation Procedure

The DVPWRI subroutine enables the program to place data anywhere in the memory/processor board. The data is 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 is 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 is 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 2D FFT is then implemented by first calling a radix-2 FFT subroutine which performs a 1D FFT on each of the rows of a 2D data array. The rows and columns are then transposed and the subroutine is called again to complete the 2D process.

The floating point data is 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

$$\Delta u = \frac{1}{N\Delta x} \quad (3.1)$$

and the equation

$$f_N = \frac{N_P}{N\Delta x} \quad (3.2)$$

where Δu is the sample interval in the Fourier domain in units of inverse length and 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 equations 3.1 and 3.2 yields

$$n_p = \frac{f_N}{\Delta u} = N_P \quad (3.3)$$

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 for 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 location which 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 iteratively. 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 (16) using the CSOS environment (17). By using the LNK command on the image processor the users 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.

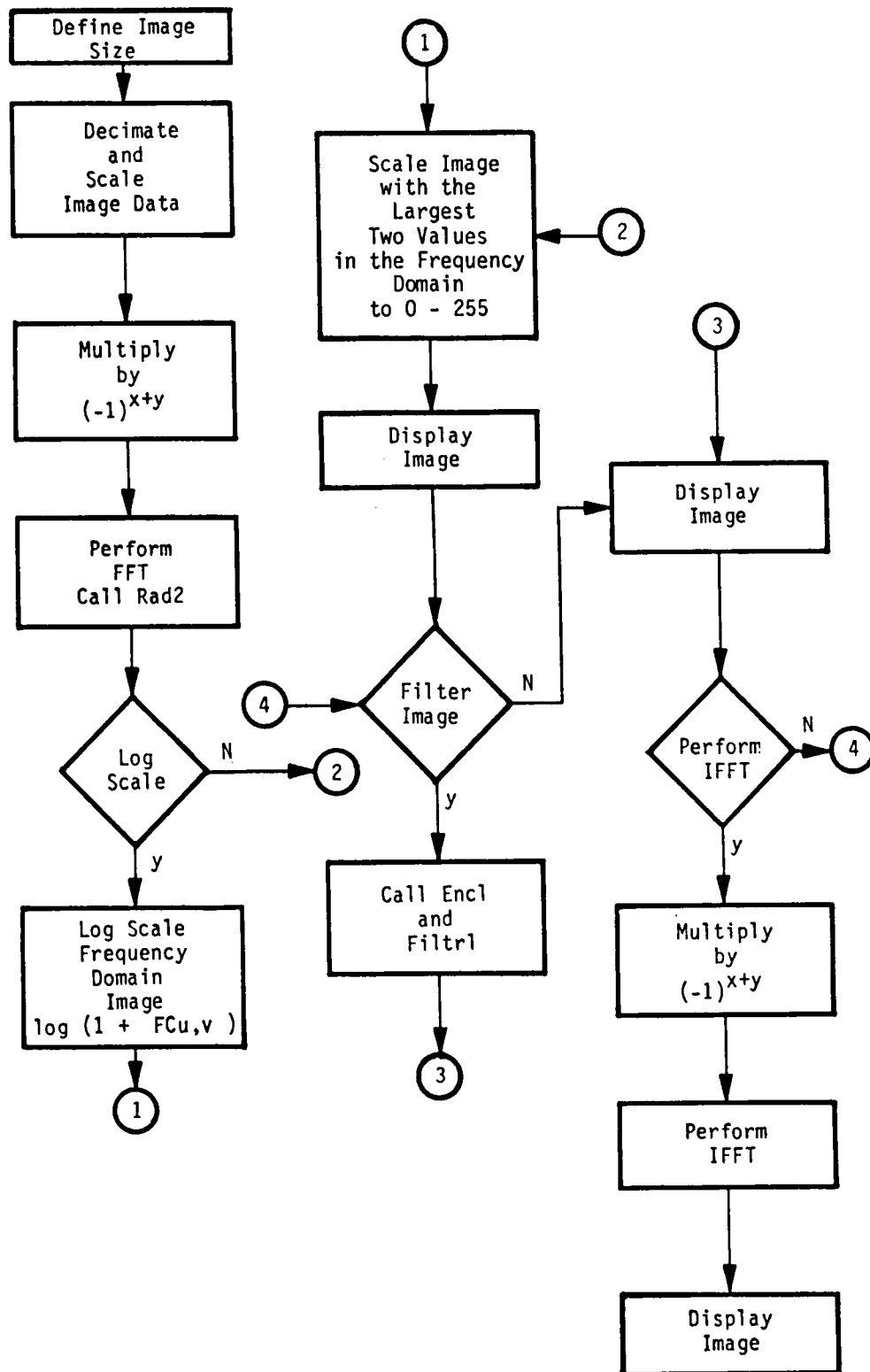


Figure 7. Flowchart of image enhancement procedure

CHAPTER IV

RESULTS

In this chapter a summary of results obtained using the FFT and filtering programs is given. The computation time and system memory utilization by the program are discussed. The results of frequency domain processed images are compared with those of spatial domain processed images in order to illustrate the advantage of one technique over the other.

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 which could be either low pass filtered, high pass filtered, or band-reject filtered as discussed in the last chapter. 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 640x480 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 256x256 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 256x256 array was processed in

approximately 15 minutes. 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 minute processing time of the 256x256 array is due to 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 2D Fourier transform took approximately 60 minutes. The 60 minute length for the procedure was due primarily to 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 rocket motor (SRM) which 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 due to 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 1x3 kernel and the average filter in Fig. 11 was performed with a 3x3 kernel. By using a 1x3 kernel size in the median filter process the edge detail of the image and the noise in Fig. 10 are enhanced. The 3x3 kernel in the average filter, however, created a blurring effect on the image as illustrated in Fig. 11, and in neither

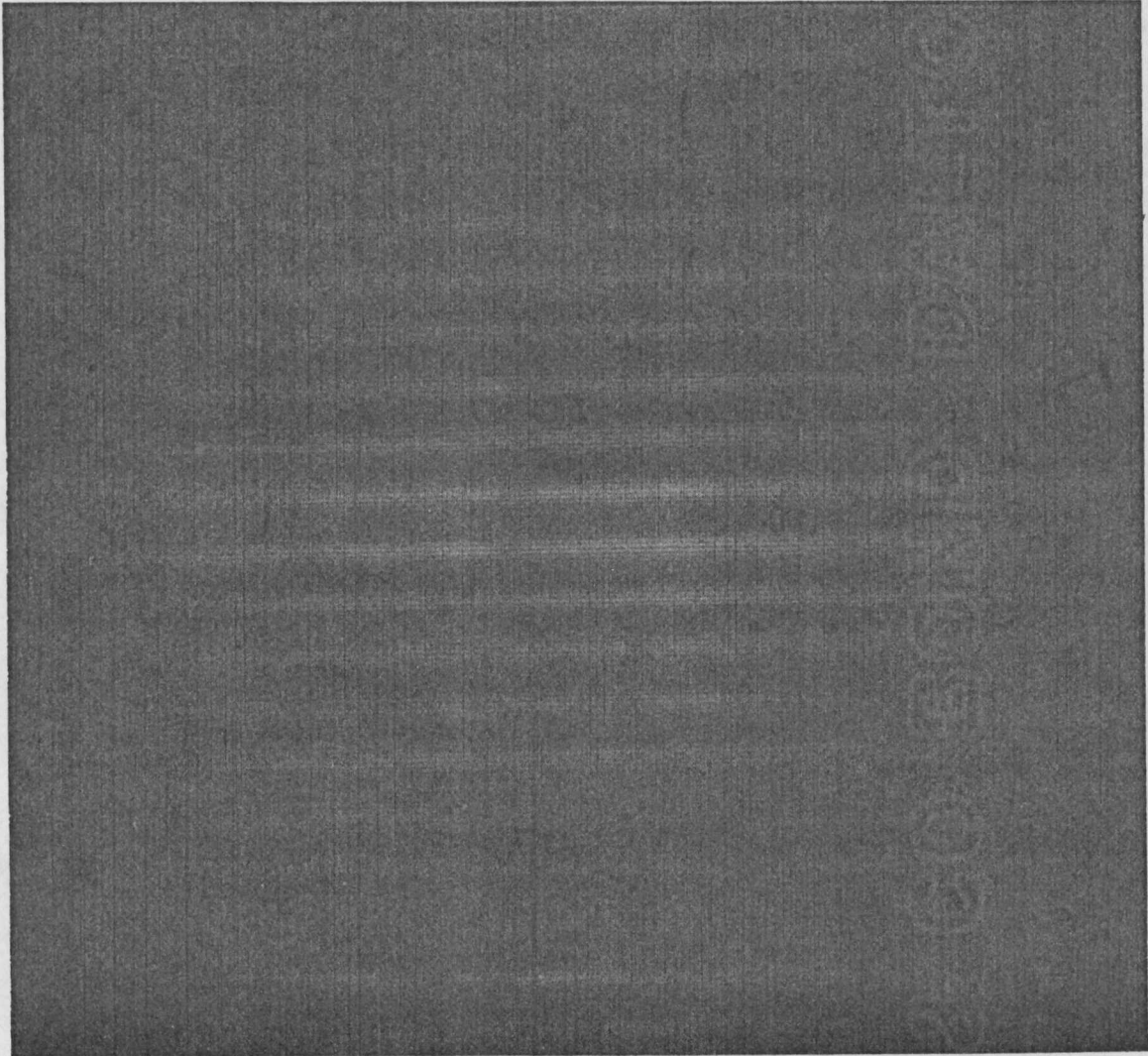


Figure 8. Noise corrupted image prior to frame averaging

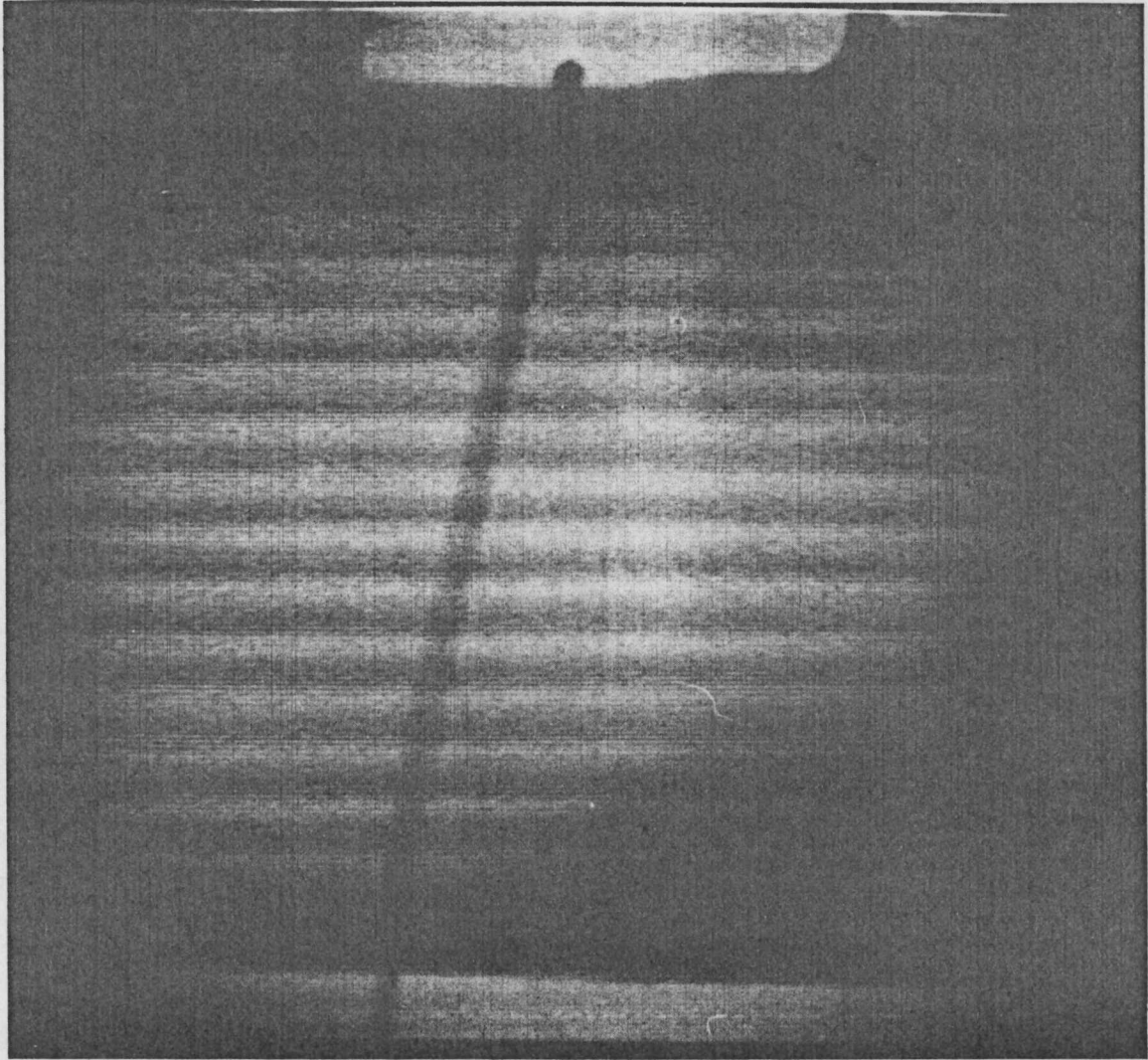


Figure 9. Image after frame averaging

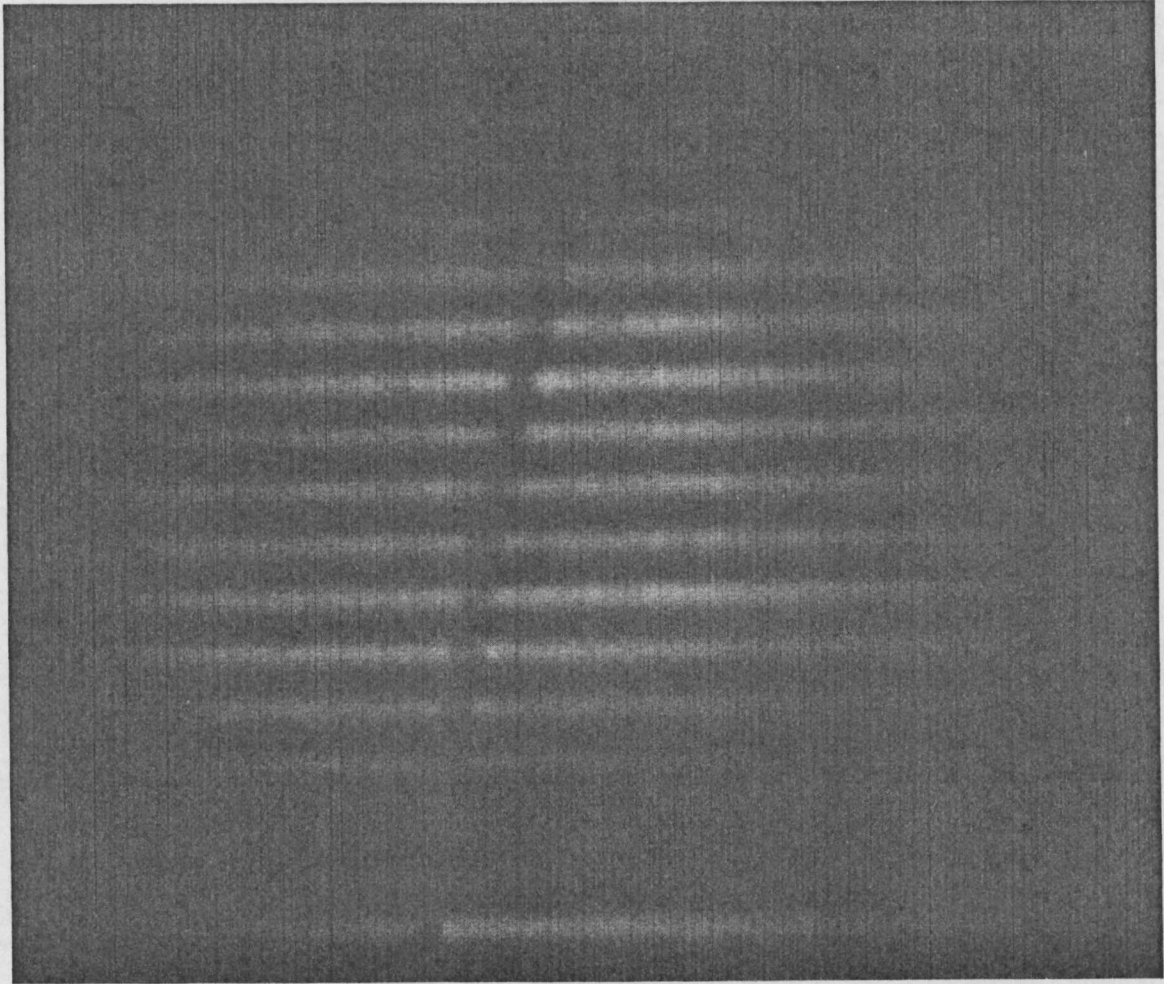


Figure 10. Image after average filtering

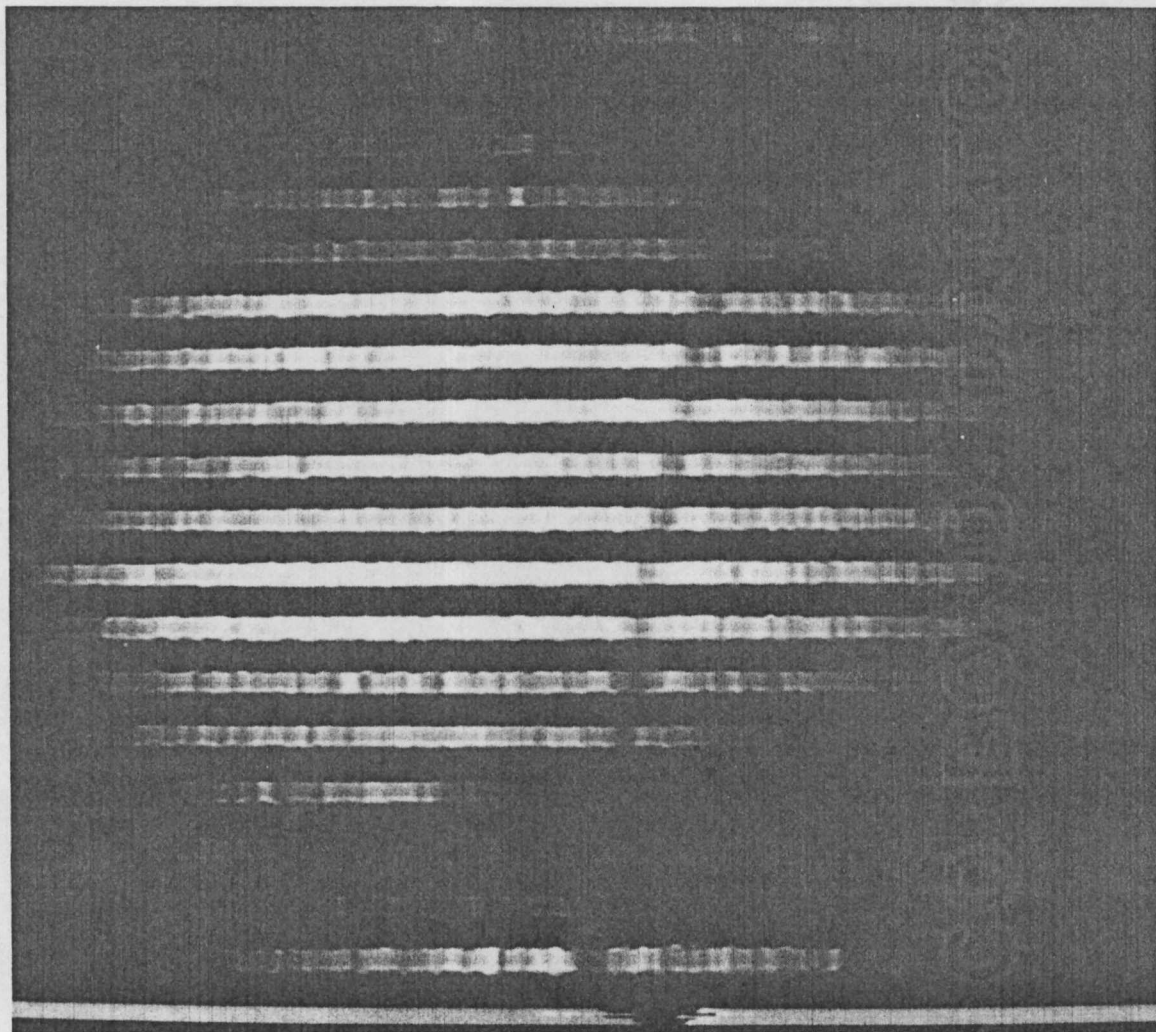


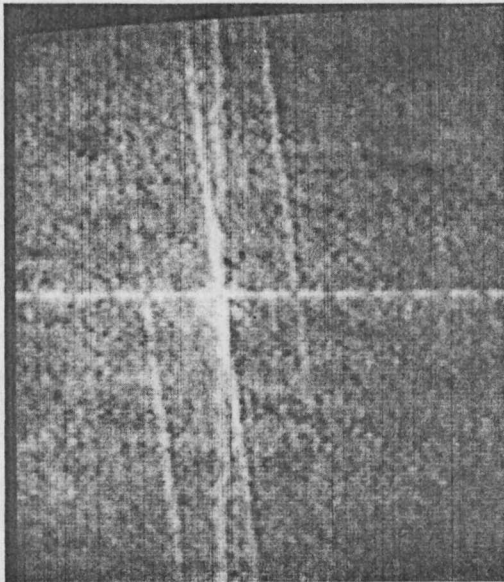
Figure 11. Image after median filtering

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 rocket motor. The low pass filtered image shown in Fig. 12(d) was created using a rectangular window function as the filter function as shown in Fig. 12(c). The image illustrates the smoothing effect that can be obtained with a frequency domain low pass filter. In Fig. 12(c), 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.

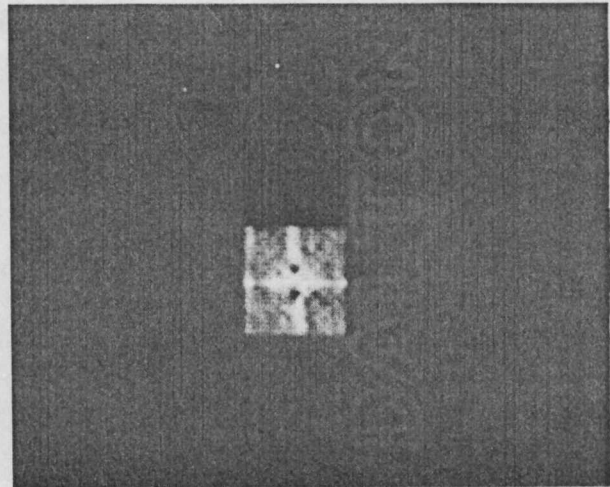
The edge region in each image was zero padded in the spatial domain after decimating the original image to 240×240 . The zero padding in the image extended the image length in the x and y directions to 256×256 which is a standard length for a radix-2 FFT. The zero padding eliminates wrap around error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

The results of applying an ideal high pass filter with a circular filter function, is illustrated in Fig. 13(c) 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.

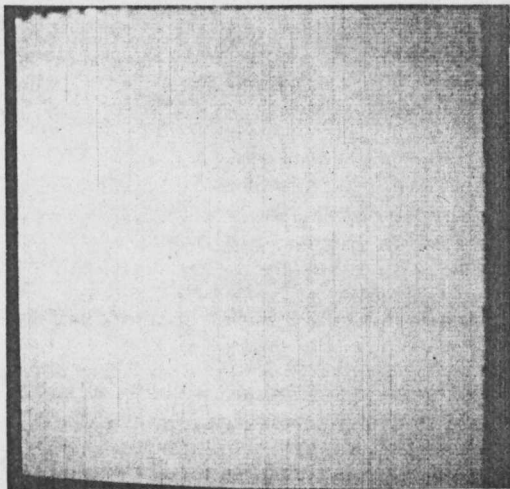
The ideal band-reject filter provides a means for removing band-limited periodic noise in the frequency domain. This is illustrated in Fig. 14. The noise identified in the frequency domain was eliminated by a circular filter with a rejection band with a radii



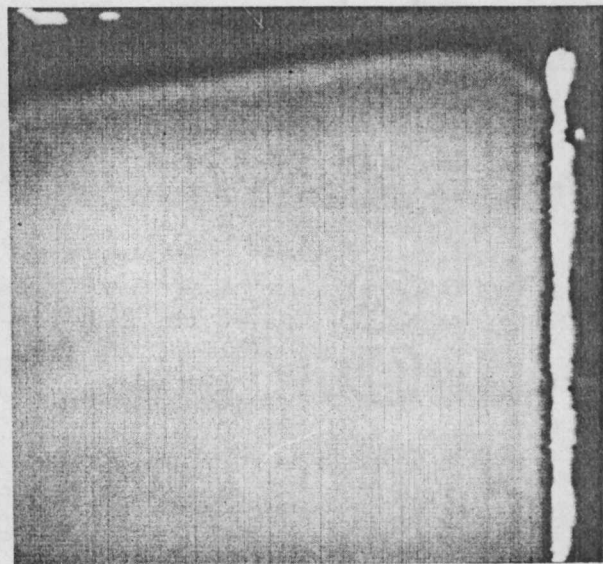
(b) Frequency Domain



(c) Filtered Frequency Domain

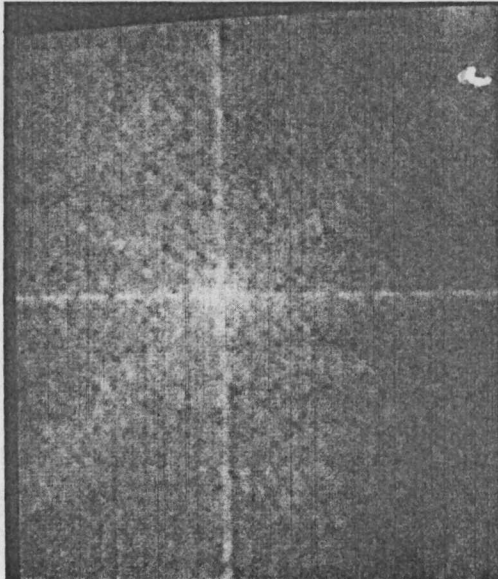


(a) Original Image

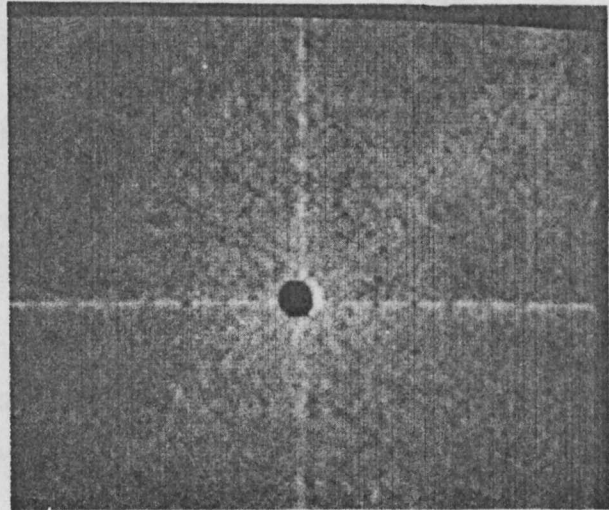


(d) Processed Image

Figure 12. Image illustrating ideal low-pass filtering technique



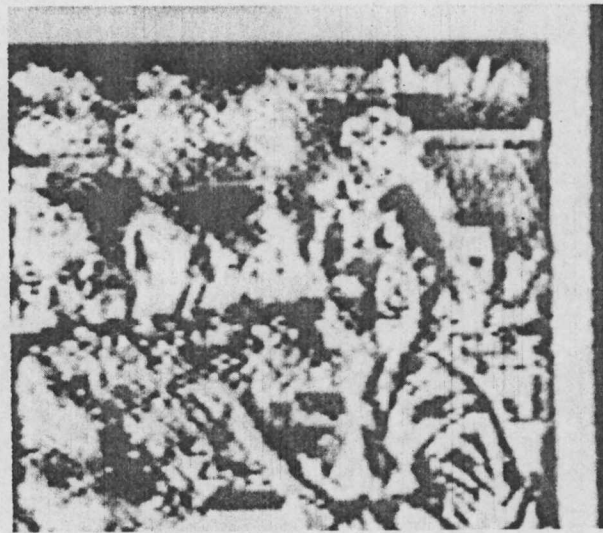
(b) Frequency Domain



(c) Filtered Frequency Domain

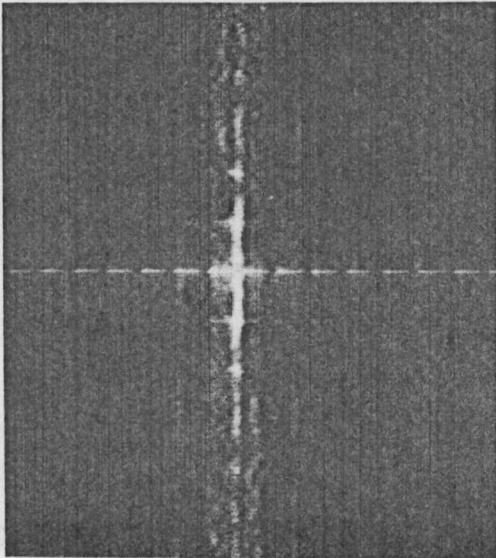


(a) Original Image

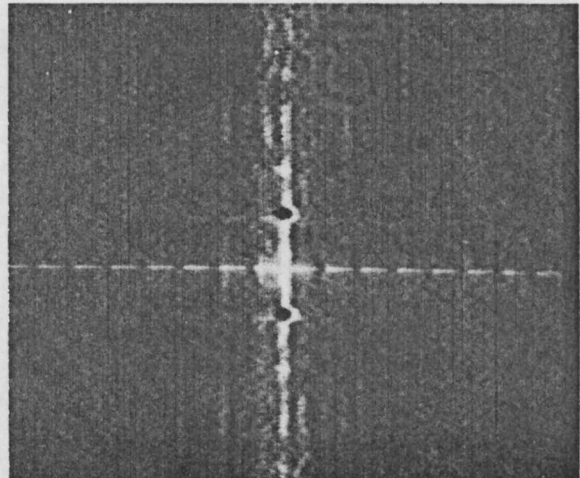


(d) Processed Image

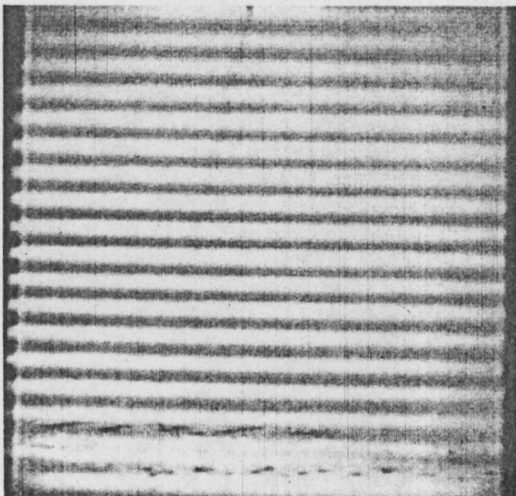
Figure 13. Image illustrating ideal high-pass filtering technique



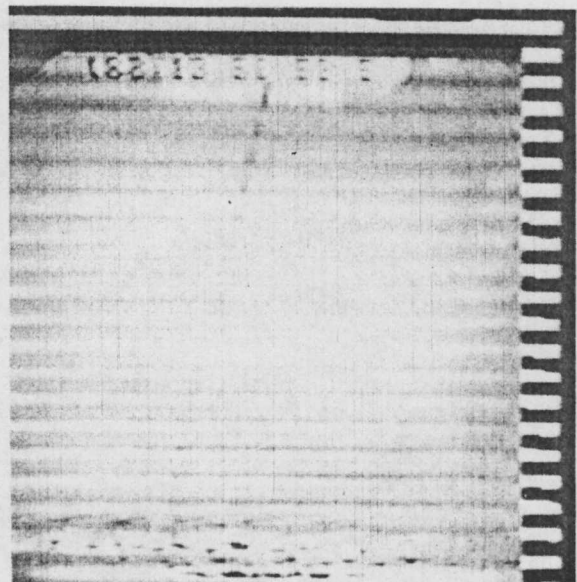
(b) Frequency Domain



(c) Filtered Frequency Domain



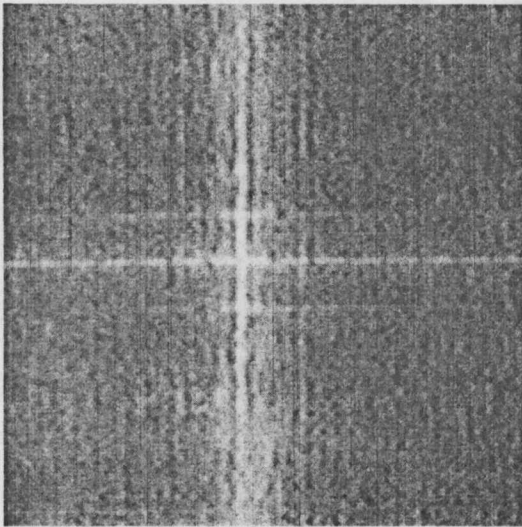
(a) Original Image



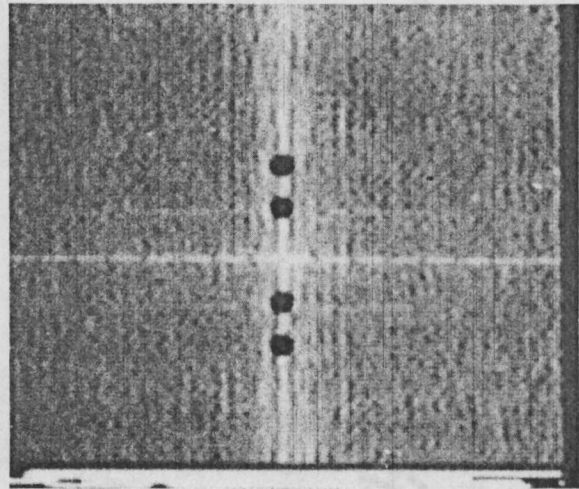
(d) Processed Image

Figure 14. Images illustrating the band-reject filtering technique and residual noise in resulting image due to noise harmonics

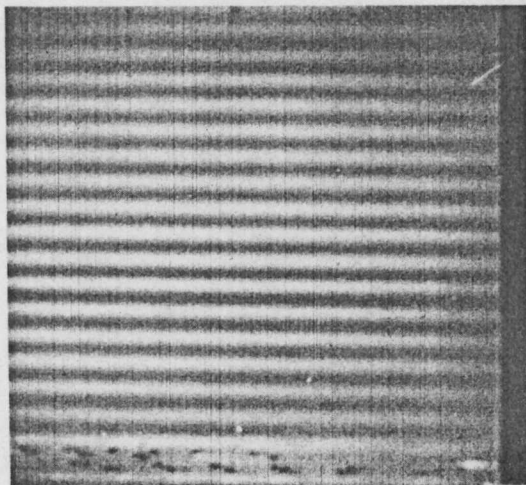
of 10 pixels. The resultant spatial domain image, however, still had residual noise present. This noise was due to second and third harmonics in the frequency domain image. In the images of the SRM shown in Figs. 15 and 16 this noise was removed by filtering out the second and third harmonics. In Fig. 15(c) and (d) it is evident that the fundamental term 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.



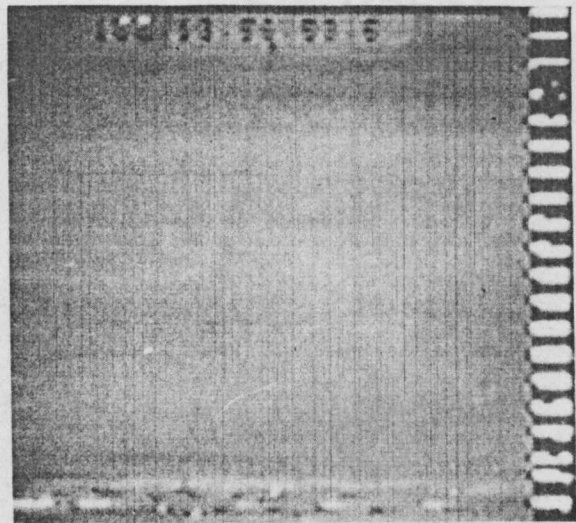
(b) Frequency Domain



(c) Filtered Frequency Domain

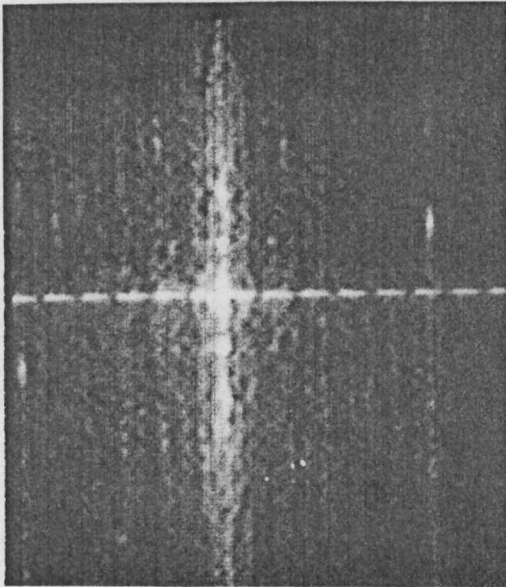


(a) Original Image

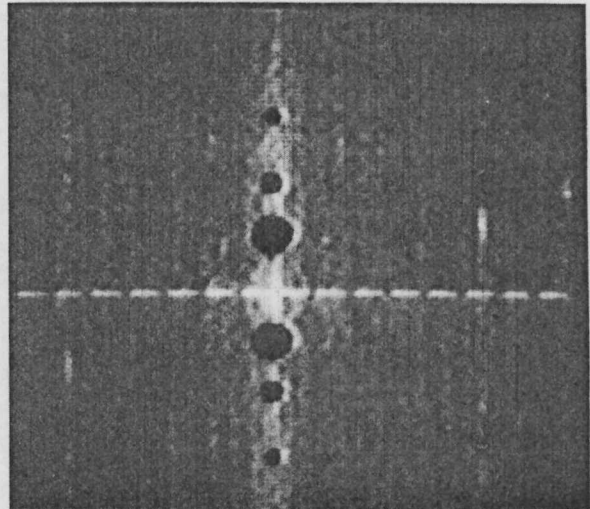


(d) Processed Image

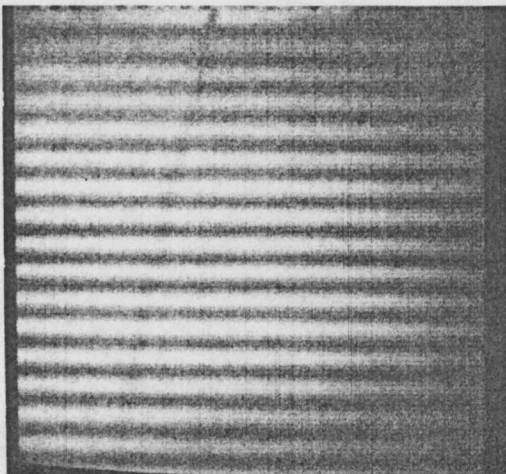
Figure 15. Images illustrating the band-reject filtering technique with second noise harmonics removed from frequency domain



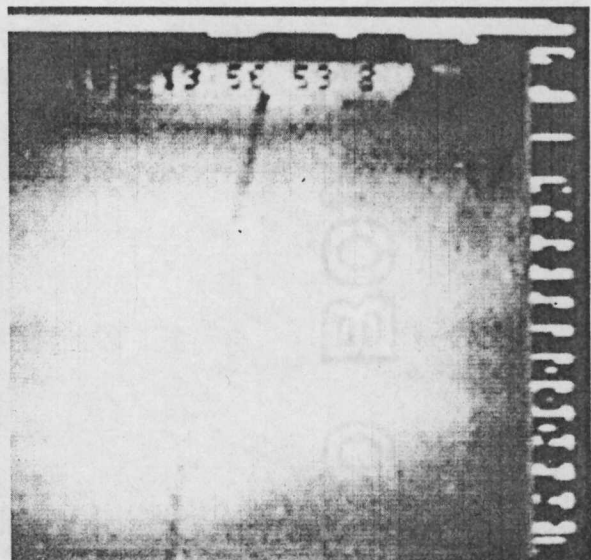
(b) Frequency Domain



(c) Filtered Frequency Domain



(a) Original Image



(d) Processed Image

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

CHAPTER V

SUMMARY AND CONCLUSIONS

In this thesis 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 2D row-column FFT and ideal filters was 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 3 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 co-processors with at least 4 megabytes of memory for 512x512 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 co-processor 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 256x256 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 frequencies 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, it was 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

REFERENCES

1. Belz, R.A., Girata, P.T., Nichols, J.A., Walker, R.Y., and Bapty, T. "Real Time Radiographic Analysis of Burning Solid Rocket Motors" 23rd Joint Propulsion Conf., San Diego, CA June 29-July 2 1987.
2. Mehalic, M.A. Effects of Computer Architecture on Fast Fourier Transforms. M.S. Thesis. Air Force Institute of Technology, Dayton OH: Dec. 1983.
3. Blaken, J.D. Efficient Computer Implementation of Fast Fourier Transform. M.S. Thesis. Air Force Institute of Technology, Dayton OH: Dec. 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, NJ: Prentice-Hall, Inc., 1979.
6. Kolba, Dean P., and Parks, Thomas W., "A Prime Factor FFT Algorithm Using High-Speed Convolution," IEEE Trans. ASSP, Aug. 1977.
7. Burrus, Sidney and Eschenbacher, Peter W. "An In-Place In-Order Prime Factor Algorithm," IEEE Trans. ASSP, Aug. 1981.
8. Burrus, Sidney and Johnson, Howard W. "On the Structure of Efficient DFT Algorithms," IEEE Trans. ASSP, Feb. 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. Sorenson, H.V., Heidelman, M.T., and Burrus, C.S. "On Computing the Split-Radix FFT," IEEE Trans. ASSP, vol. ASSP-34, pp. 152-156, Feb. 1986.
12. Duhamel P. and Hollman H. "Split Radix FFT Algorithm," Electronic Letters, Vol. 20, No. 1, January 1984.
13. Brigham, E.Oran. The Fast Fourier Transform. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1974.
14. Gonzales, Rafael C. and Wintz, Paul. Digital Image Processing. Sixth Edition. Reading, Massachusetts: Addison-Wesley Publishing Co., 1983.

15. Mersereau, Russel M., and Dudgeon, Dan E. Multidimensional Digital Signal Processing. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1984
16. Quantex Corp. Quantex QX-9000 Users Guide, October 1984.
17. IBM Corp. CSOS Users Guide, IBM Instruments 1983.

APPENDIX

APPENDIX

The Quantex 9210 Image Processor at 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/pipelined point processor board. The analog board consists of a flash A/D converter which 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 640X480 image.

The digitized A/D output is routed to the memory/processor board which is a three function device. It consists of a pipelined processor which 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

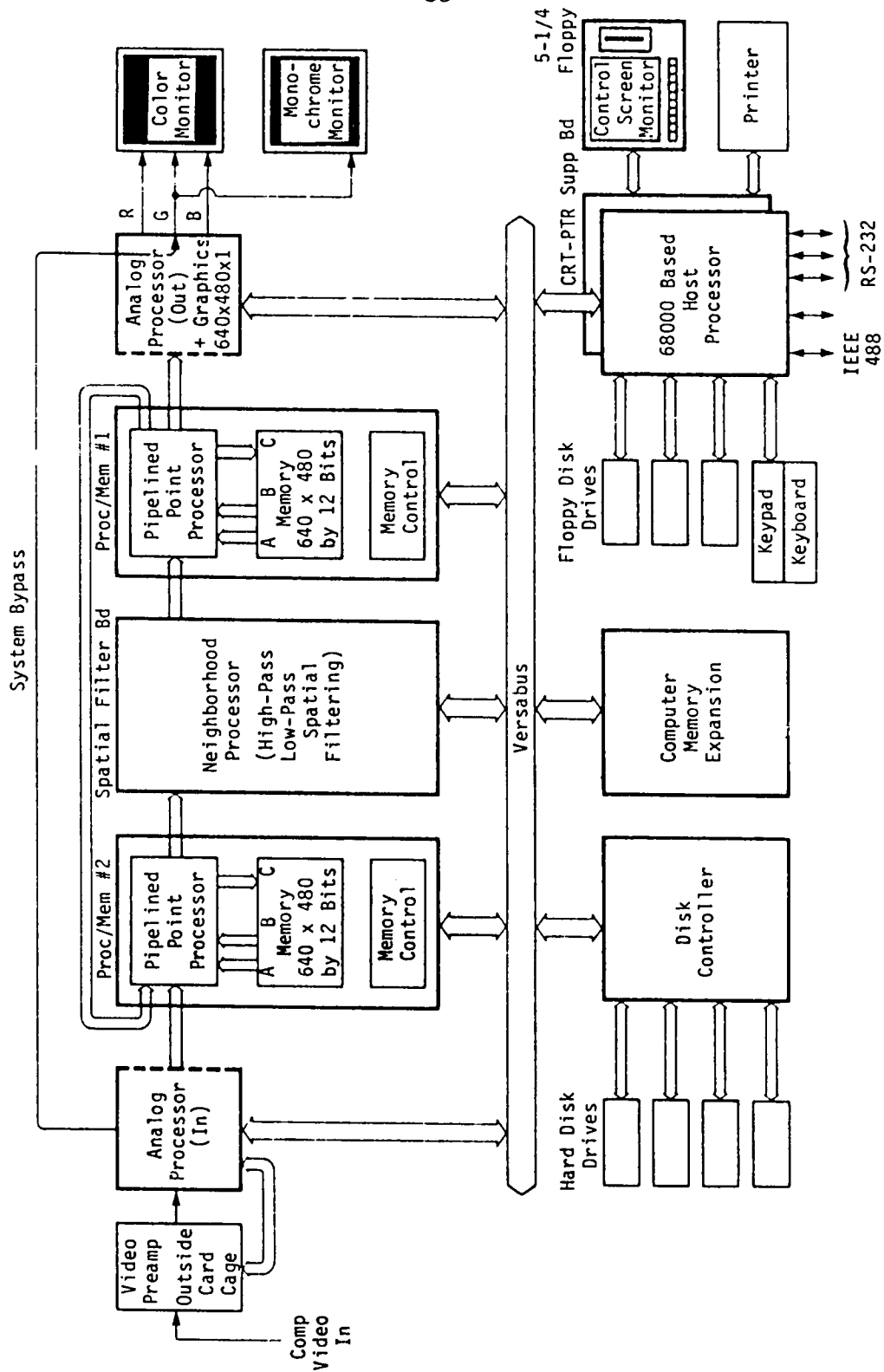


Figure A.1. QX-9210 system block diagram

is organized in a 640X480 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 which 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 which 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 Operations

The Quantex 9210 is also equipped with a second memory/processor board which allows the user to perform subtraction or differencing operations on a image in the video pipeline and a spatial board which 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 Co-Processor (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 utilizes a Motorola 68020, a true 32-bit 16 Mhz CPU, for intensive computational analysis while the IBM PC/AT utilizes 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 co-processor 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, multi-tasking facility. It can support the following languages: Basic, Fortran, Pascal, and MC68000 macro-assembler. 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 be 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 which 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/processors 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 which is defined in the users program for passing error codes. The other items which are unique to certain subroutines will 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 utilized 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 set-up as 4096 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 to. 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 similiar to the DVPLUT except that DVPLUW enables the user to write to all 4096 locations of the memory/processor board look-up table.

VITA

David Lee Carson was born August 19, 1958, in Akron, Ohio. Mr. Carson obtained the Bachelor of Science degree in Electrical Engineering with distinction from Tennessee State University in May 1981. After completing his undergraduate studies Mr. Carson was employed by the Tennessee Valley Authority and the Arnold Engineering Development Center for a period of three years. Mr. Carson returned to full-time graduate study at the University of Tennessee Space Institute in September of 1984. The Master of Science degree was conferred upon Mr. Carson in December of 1987.