RECURSIVE DIGITAL FILTERS: DESIGN AND APPLICATIONS TO IMAGE PROCESSING.

APPANNA T. CHOTTERA

University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
CHOTTERA, APPANNA T., "RECURSIVE DIGITAL FILTERS: DESIGN AND APPLICATIONS TO IMAGE PROCESSING." (1979). Electronic Theses and Dissertations. 3287.
https://scholar.uwindsor.ca/etd/3287

This online database contains the full-text of PhD dissertations and Masters’ theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000 ext. 3208.
The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us a poor photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

THIS DISSERTATION HAS BEEN MICROFILMED EXACTLY AS RECEIVED

Ottawa, Canada
K1A 0N4
RECURSIVE DIGITAL FILTERS: DESIGN AND APPLICATIONS TO IMAGE PROCESSING

by

Appanna T. Chottera

A Dissertation submitted to the Faculty of Graduate Studies through the Department of Electrical Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the University of Windsor

Windsor, Ontario, Canada
1979
© Appanna T. Chottera
ABSTRACT

The phase characteristics of a digital filter is of importance in many applications; specifically, in image processing problems. Part of the work reported in this thesis is devoted to the design of recursive digital filters in which a simultaneous approximation of the desired magnitude and linear phase is performed, using linear programming. The use of linear programming is facilitated via the linearization of the inherently non-linear approximation problem. Using this approach, both one and two dimensional quarter plane recursive digital filters have been designed, the examples which are provided with the design algorithm. Linear stability constraints are proposed, which can easily be incorporated into the linear programming design procedure, to enable the design of stable filters. These stability constraints are sufficient conditions for stability, and therefore allow the design of a subclass of one and two dimensional quarter plane recursive digital filters.

Considering the computational advantage of recursive digital filters, compared to most convolutional methods of filtering (specifically, convolution filtering via the Fast Fourier Transform), the second part of this thesis examines applications of quarter plane two dimensional recursive digital filters in image processing. The thesis considers applications in the areas of image enhancement and restoration problems. The problems considered in image enhancement are: high frequency emphasis and edge enhancement. The
problems of image restoration are considered for the cases of motion, focus and atmospheric turbulence blurs.
ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my supervisor, Dr. G.A. Jullien for his invaluable advice, help and constructive criticisms during the course of this research. The valuable advice and comments of Dr. W.C. Miller and Dr. J.J. Soltis, are gratefully acknowledged. In addition, the help of many of the graduate students and the technicians, Mr. Novasad and Mr. Reiter, of the Electrical Engineering Department is sincerely appreciated.

To my parents, I extend my sincerest thanks. Without their help and love, though far away, this work would not have started.

Thanks are also due to Mrs. Barbara L. Denomey for her excellent typing of this thesis.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>i</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF ILLUSTRATIONS</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF APPENDICES</td>
<td>xi</td>
</tr>
<tr>
<td>CHAPTER I INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 ONE AND TWO DIMENSIONAL SIGNAL PROCESSING</td>
<td>1</td>
</tr>
<tr>
<td>1.2 RECURSIVE DIGITAL FILTER DESIGN AND APPLICATIONS</td>
<td>3</td>
</tr>
<tr>
<td>1.3 PROBLEM STATEMENT</td>
<td>7</td>
</tr>
<tr>
<td>1.4 THESIS ORGANIZATION</td>
<td>8</td>
</tr>
<tr>
<td>CHAPTER II RECURSIVE DIGITAL FILTER DESIGN USING LINEAR PROGRAMMING</td>
<td>9</td>
</tr>
<tr>
<td>2.1 INTRODUCTION</td>
<td>9</td>
</tr>
<tr>
<td>2.2 LINEAR PROGRAMMING</td>
<td>9</td>
</tr>
<tr>
<td>2.2.a Linear Programming Theory</td>
<td>10</td>
</tr>
<tr>
<td>2.2.b Dual Linear Programming</td>
<td>12</td>
</tr>
<tr>
<td>2.3 ONE DIMENSIONAL RECURSIVE FILTER DESIGN</td>
<td>14</td>
</tr>
<tr>
<td>2.3.a Theory of Approximation</td>
<td>14</td>
</tr>
<tr>
<td>2.3.b Stability Constraints in One Dimension</td>
<td>21</td>
</tr>
<tr>
<td>2.3.c Design Procedure</td>
<td>25</td>
</tr>
<tr>
<td>2.3.d Computational Considerations</td>
<td>30</td>
</tr>
<tr>
<td>2.3.e Examples of Design</td>
<td>34</td>
</tr>
<tr>
<td>2.4 TWO DIMENSIONAL RECURSIVE FILTER DESIGN</td>
<td>48</td>
</tr>
<tr>
<td>2.4.a Theory of Approximation</td>
<td>49</td>
</tr>
<tr>
<td>2.4.b Stability Constraints in Two Dimensions</td>
<td>51</td>
</tr>
<tr>
<td>2.4.c Design Procedure</td>
<td>52</td>
</tr>
<tr>
<td>2.4.d Computational Considerations</td>
<td>56</td>
</tr>
<tr>
<td>2.4.e Design Examples</td>
<td>61</td>
</tr>
<tr>
<td>2.5 SUMMARY</td>
<td>70</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>2.1</td>
<td>Graphical Explanation of the Linear Program</td>
</tr>
<tr>
<td>2.2</td>
<td>Constraint Regions for Poles of First Order Transfer Function</td>
</tr>
<tr>
<td>2.3</td>
<td>Range of $\tau_s$ in the Impulse Response</td>
</tr>
<tr>
<td>2.4</td>
<td>Desired Magnitude Characteristics</td>
</tr>
<tr>
<td>2.5</td>
<td>Designed Magnitude Characteristics</td>
</tr>
<tr>
<td>2.6</td>
<td>Designed Group Delay Characteristics</td>
</tr>
<tr>
<td>2.7</td>
<td>Designed Impulse Response</td>
</tr>
<tr>
<td>2.8</td>
<td>Pole-Zero Positions of the Designed Filter</td>
</tr>
<tr>
<td>2.9</td>
<td>Realization of Differentiator with Half Sample Delay</td>
</tr>
<tr>
<td>2.10</td>
<td>Desired Magnitude Characteristics</td>
</tr>
<tr>
<td>2.11</td>
<td>Designed Filter Magnitude Characteristics</td>
</tr>
<tr>
<td>2.12</td>
<td>Designed Filter Group Delay Characteristics</td>
</tr>
<tr>
<td>2.13</td>
<td>Error in the Magnitude</td>
</tr>
<tr>
<td>2.14</td>
<td>Impulse Response of the Designed Filter</td>
</tr>
<tr>
<td>2.15</td>
<td>Pole-Zero Positions of the Designed Filter</td>
</tr>
<tr>
<td>2.16(a)</td>
<td>Desired Magnitude Specifications</td>
</tr>
<tr>
<td>(b)</td>
<td>Designed Magnitude Response</td>
</tr>
<tr>
<td>(c) &amp; (d)</td>
<td>Designed Group Delays $\tau_1$ and $\tau_2$</td>
</tr>
<tr>
<td>2.17(a)</td>
<td>First 16x16 Pixels of the Impulse Response</td>
</tr>
<tr>
<td>(b)</td>
<td>Designed Magnitude Response</td>
</tr>
<tr>
<td>2.18(a)</td>
<td>Desired Magnitude Specifications</td>
</tr>
<tr>
<td>(b)</td>
<td>Designed Magnitude Response</td>
</tr>
<tr>
<td>(c) &amp; (d)</td>
<td>Designed Group Delays $\tau_1$ and $\tau_2$</td>
</tr>
<tr>
<td>2.19(a)</td>
<td>First 16x16 Pixels of the Impulse Response</td>
</tr>
<tr>
<td>(b)</td>
<td>Coefficients of the Designed Filter</td>
</tr>
<tr>
<td>2.20(a)</td>
<td>Desired Magnitude Specification</td>
</tr>
<tr>
<td>(b)</td>
<td>Designed Magnitude Response</td>
</tr>
<tr>
<td>(c) &amp; (d)</td>
<td>Designed Group Delays $\tau_1$ and $\tau_2$</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2.21(a)</td>
<td>First 16x16 Pixels of Impulse Response</td>
</tr>
<tr>
<td></td>
<td>Coefficients of the Designed Filter</td>
</tr>
<tr>
<td>3.1(a)</td>
<td>Original Image</td>
</tr>
<tr>
<td></td>
<td>Enhanced Image</td>
</tr>
<tr>
<td>3.2(a)</td>
<td>Designed High Pass Filter Magnitude Response</td>
</tr>
<tr>
<td></td>
<td>Designed Group Delay Responses $\tau_1$ and $\tau_2$</td>
</tr>
<tr>
<td></td>
<td>First 8x8 Pixels of the Impulse Response</td>
</tr>
<tr>
<td></td>
<td>Coefficients of the Designed Filter</td>
</tr>
<tr>
<td>3.3(a)</td>
<td>Desired Magnitude Specifications</td>
</tr>
<tr>
<td></td>
<td>Designed Magnitude Response</td>
</tr>
<tr>
<td></td>
<td>Designed Group Delays $\tau_1$ and $\tau_2$</td>
</tr>
<tr>
<td></td>
<td>First 8x8 Pixels of the Designed Filter Impulse Response</td>
</tr>
<tr>
<td></td>
<td>Coefficients of the Designed Filter</td>
</tr>
<tr>
<td>3.4(a)</td>
<td>Original Image</td>
</tr>
<tr>
<td></td>
<td>Edge Enhanced Image with Linear Phase Filter</td>
</tr>
<tr>
<td></td>
<td>Edge Enhanced Image with Non-Linear Phase Filter</td>
</tr>
<tr>
<td>3.5(a)</td>
<td>Magnitude Response</td>
</tr>
<tr>
<td></td>
<td>Group Delays $\tau_1$ and $\tau_2$</td>
</tr>
<tr>
<td></td>
<td>Impulse Response</td>
</tr>
<tr>
<td></td>
<td>Coefficients of the Filter</td>
</tr>
<tr>
<td>3.6(a)</td>
<td>Motion Blur Spatial Response</td>
</tr>
<tr>
<td></td>
<td>Motion Blur Transfer Function</td>
</tr>
<tr>
<td></td>
<td>Amount of Phase Shift Introduced by (b)</td>
</tr>
<tr>
<td>3.7(a)</td>
<td>Blurring Transfer Function. Blur Length Equals 4 Pixels; Angle=45° w.r.t. Horizontal</td>
</tr>
<tr>
<td></td>
<td>Original Image</td>
</tr>
<tr>
<td></td>
<td>Image Blurred by (a), With Noise, SNR=30 db</td>
</tr>
<tr>
<td>3.8(a)</td>
<td>Restoration Filter Response</td>
</tr>
<tr>
<td></td>
<td>Restoration Filter Response Along $\theta=45^\circ$</td>
</tr>
<tr>
<td></td>
<td>Phase Shift Introduced by (b)</td>
</tr>
<tr>
<td>3.9(a)</td>
<td>Recursive Filter Implementation for Motion Deblur</td>
</tr>
<tr>
<td></td>
<td>Frequency Response of (a)</td>
</tr>
<tr>
<td>3.10(a)</td>
<td>Impulse Response of Desired Restoration Filter</td>
</tr>
<tr>
<td></td>
<td>Impulse Response of Recursive Filter Implementation</td>
</tr>
<tr>
<td></td>
<td>Coefficients of the Recursive Filter $H_1(Z_1, Z_2)$</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3.11(a)</td>
<td>Original Image</td>
</tr>
<tr>
<td></td>
<td>(b) Blurred Image with Noise, Blur Length=4, Angle=45°, SNR=30 db</td>
</tr>
<tr>
<td></td>
<td>(c) Image Restored Using FFT</td>
</tr>
<tr>
<td></td>
<td>(d) Image Restored Using Recursive Implementation of Figure 3.9(a)</td>
</tr>
<tr>
<td>3.12(a)</td>
<td>Motion Blurring Transfer Function; Blur Length=8 Pixels, Angle=22.5° w.r.t.</td>
</tr>
<tr>
<td></td>
<td>Horizontal</td>
</tr>
<tr>
<td></td>
<td>(b) PSEC Restoration Filter Corresponding to (a) and SNR=30 db</td>
</tr>
<tr>
<td></td>
<td>(c) Recursive Filter Approximation to (b)</td>
</tr>
<tr>
<td>3.13(a)</td>
<td>Impulse Response of the Desired Restoration Filter</td>
</tr>
<tr>
<td></td>
<td>(b) Impulse Response of Recursive Filter Implementation</td>
</tr>
<tr>
<td>3.14</td>
<td>Coefficients of the Recursive Filter</td>
</tr>
<tr>
<td></td>
<td>$H_1(z_1, z_2)$</td>
</tr>
<tr>
<td>3.15(a)</td>
<td>Original Image</td>
</tr>
<tr>
<td></td>
<td>(b) Motion Blurred Image With Noise</td>
</tr>
<tr>
<td></td>
<td>(c) Restoration Using FFT</td>
</tr>
<tr>
<td></td>
<td>(d) Restoration Using Recursive Filter</td>
</tr>
<tr>
<td></td>
<td>Implementation of Figure 3.9(a)</td>
</tr>
<tr>
<td>3.16(a)</td>
<td>Focus Blur Point Spread Function</td>
</tr>
<tr>
<td></td>
<td>(b) Transfer Function of Focus Blur</td>
</tr>
<tr>
<td></td>
<td>(c) Phase Shift Introduced by (b)</td>
</tr>
<tr>
<td>3.17(a)</td>
<td>Focus Blurring Transfer Function Blur Radius=4 Pixels</td>
</tr>
<tr>
<td></td>
<td>(b) Original Image</td>
</tr>
<tr>
<td></td>
<td>(c) Focus Blurred Noisy Image; SNR=30 db</td>
</tr>
<tr>
<td>3.18(a)</td>
<td>Restoration Filter Response</td>
</tr>
<tr>
<td></td>
<td>(b) Restoration Filter Response Along 45° Radial Direction</td>
</tr>
<tr>
<td></td>
<td>(c) Phase Shift Introduced by (b)</td>
</tr>
<tr>
<td>3.19(a)</td>
<td>Recursive Filter Implementation for Focus Deblur</td>
</tr>
<tr>
<td></td>
<td>(b) Frequency Response of (a)</td>
</tr>
<tr>
<td>3.20</td>
<td>Coefficients of Two Dimension Filter $H_1(z_1, z_2)$</td>
</tr>
<tr>
<td>3.21</td>
<td>Coefficients of Two Dimension Filter $H_2(z_1, z_2)$</td>
</tr>
<tr>
<td>3.22(a)</td>
<td>Restoration by Convolution Via FFT</td>
</tr>
<tr>
<td></td>
<td>(b) Restoration Using Recursive Filter</td>
</tr>
<tr>
<td></td>
<td>Implementation of Figure 3.19(a)</td>
</tr>
<tr>
<td>3.23(a)</td>
<td>Atmospheric Turbulance Blur Transfer Function</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3.23(b)</td>
<td>One Dimensional Plot of the Transfer Function Along any Radial Direction</td>
</tr>
<tr>
<td>3.24(a)</td>
<td>Restoration Filter Response for Atmospheric Blur</td>
</tr>
<tr>
<td>(b)</td>
<td>Plot of Magnitude Blur of (a) Along 45° Radial Direction</td>
</tr>
<tr>
<td>3.25(a)</td>
<td>Magnitude Response of the Designed Filter</td>
</tr>
<tr>
<td>(b)</td>
<td>Coefficients of the Designed Filter</td>
</tr>
<tr>
<td>3.26(a)</td>
<td>Impulse Response Corresponding to Desired Magnitude Specification</td>
</tr>
<tr>
<td>(b)</td>
<td>Impulse Response of the Designed Filter</td>
</tr>
<tr>
<td>3.27(a)</td>
<td>Original Image</td>
</tr>
<tr>
<td>(b)</td>
<td>Blurred Noisy Image</td>
</tr>
<tr>
<td>(c)</td>
<td>Restoration Using FFT</td>
</tr>
<tr>
<td>(d)</td>
<td>Restoration Using Recursive Filter</td>
</tr>
<tr>
<td>3.28(a)</td>
<td>Recursive Filter Implementation for Atmospheric Turbulence Deblur</td>
</tr>
<tr>
<td>(b)</td>
<td>Desired Restoration Filter Frequency Response</td>
</tr>
<tr>
<td>(c)</td>
<td>Frequency Response Realized by (a)</td>
</tr>
<tr>
<td>3.29(a)</td>
<td>Impulse Response Corresponding to Figure 3.28(b)</td>
</tr>
<tr>
<td>(b)</td>
<td>Impulse Response Corresponding to Figure 3.28(c)</td>
</tr>
<tr>
<td>(c)</td>
<td>Coefficients of $H(z_1, z_2)$</td>
</tr>
<tr>
<td>3.30(a)</td>
<td>Original Image</td>
</tr>
<tr>
<td>(b)</td>
<td>Atmospheric Turbulence Blurred Image With Noise</td>
</tr>
<tr>
<td>(c)</td>
<td>Restoration Using FFT</td>
</tr>
<tr>
<td>(d)</td>
<td>Implementation of Figure 3.28(a)</td>
</tr>
<tr>
<td>A1</td>
<td>An Example of Input and Output Masks to Calculate $y(3,3)$ for Equation (A.1)</td>
</tr>
<tr>
<td>A2</td>
<td>An Example of Input and Output Masks to Calculate $y(2,2)$ for Equation (A.8)</td>
</tr>
<tr>
<td>A3</td>
<td>Direction of Delays in Spatial Domain</td>
</tr>
<tr>
<td>B1</td>
<td>Schematic of Image Formation System</td>
</tr>
<tr>
<td>B2</td>
<td>Block Diagram Representation of Image Formation and Recording</td>
</tr>
<tr>
<td>C1</td>
<td>Figure C1</td>
</tr>
<tr>
<td>D1</td>
<td>Solution Region Corresponding to Various Types of Constraints</td>
</tr>
<tr>
<td>E1</td>
<td>Figure E1</td>
</tr>
<tr>
<td>G1</td>
<td>Block Diagram of Image Processing Facility</td>
</tr>
</tbody>
</table>
Figure

G2  Flying Spot Scanner System.................. 192
G3  Timing and Control Circuit and
    Integrator Set-Up.......................... 195
G4  Timing and Control Circuit.................. 197
G5  Timing Diagram................................ 198
G6  Photomultiplier Tube Output Amplifier..... 200
G7  Image Display and Reproduction
    Hardware.................................... 202
H1  Sampling of Image in a Rectangular Array..... 238

Table

4.1  Computations for Recursive Filtering...... 136
**LIST OF APPENDICES**

<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPENDIX A</td>
<td>Digital Filtering Fundamentals</td>
<td>145</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>Digital Image Processing Fundamentals</td>
<td>163</td>
</tr>
<tr>
<td>APPENDIX C</td>
<td>Proof of Stability Criterion for One Dimensional Filter Design</td>
<td>169</td>
</tr>
<tr>
<td>APPENDIX D</td>
<td>On The Stability Criterion for Second Order One Dimensional Filter</td>
<td>173</td>
</tr>
<tr>
<td>APPENDIX E</td>
<td>Proof of the Sufficiency of the Two Dimensional Stability Constraint</td>
<td>178</td>
</tr>
<tr>
<td>APPENDIX F</td>
<td>Two Dimensional Recursive Digital Quarter Plane Filter Design in Cascade Form</td>
<td>181</td>
</tr>
<tr>
<td>APPENDIX G</td>
<td>Image Processing Hardware</td>
<td>189</td>
</tr>
<tr>
<td>APPENDIX H</td>
<td>Filter Design, Image Filtering and Other Additional Image Processing Computer Programs</td>
<td>204</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

1.1 One and Two Dimensional Signal Processing

In recent years there has been a rapid growth of interest in computer processing of both one and two dimensional digital signals. Examples of one dimensional signals of interest are speech, ECG (Electro Cardiogram) and EEG (Electro Encephalogram). Two dimensional signals of interest include photographic data, such as medical x-rays, aerial photographic data used for geographic purposes and non-pictorial data such as the data corresponding to seismic signals obtained in the exploration for oil and gas.

The purpose of processing such signals is manifold. In the case of speech signals, for example, it may be required to extract information corresponding to the identity of a speaker, or in the case of ECG it may be data compression, where the purpose is to reduce storage requirement to a minimum. In the two dimensional case examples are the enhancement of satellite pictures for improving image quality, or, in the case of ERTS (Earth Resource Technology Satellite) pictures, the purpose may be to obtain a classification of earth's resources.

In many cases the processing of such signals by analog techniques is unattractive. In the one dimensional signal processing, for example, consider an analog filter which requires twenty poles to accomplish a design objective. In this case even if the filter is realized actively with
isolation between stages, it probably will be almost impos-
sible to tune. In the two dimensional case an example of
analog processing is the processing of photographic data by
optical techniques. Although optical processing is a com-
pletely parallel process (and hence fast), digital processing
is much more flexible and does not require the set up proce-
dures normally associated with, for example, optical filtering.
Optical filtering is normally restricted to handling only
linear filtering problems. In an all digital environment, it
is possible to carry out iterative processes and processes
requiring tests and decisions as well as normal linear, and
non-linear filtering algorithms. To carry out these processes
one may employ a general purpose digital computer, along with
some dedicated hardware to perform specific tasks.

One of the most common signal processing operations
which can be performed digitally is linear filtering. As an
example, the edge enhancement of a picture can be carried out
by two dimensional high pass filtering. The need for such
types of processing and the decreasing cost of performing
these processes digitally has given rise to considerable
research interest in the area of digital filter design and
their applications. Digital filters are of two types, namely;
a) Finite impulse response (FIR) and b) Infinite impulse re-
ponse or Recursive. As a part of this research interest,
this dissertation presents a method for the design of, a class
of digital filters called the recursive (infinite impulse
response) digital filters in one and two dimensions and
examines their applications in certain image processing pro-
blems. A complete description of the types of digital filters and various other terminologies used in digital filtering and digital image processing is given in Appendices A and B.

1.2 Recursive Digital Filter Design and Applications

In recent years considerable work in the area of recursive digital filter design has been and is still being carried out, primarily because recursive digital filters offer greater speed of filtering, smaller memory requirements and easier implementations compared to FIR filters. Many of the one dimensional recursive digital filter design techniques can be found in [1] and many of the recent two dimensional design techniques are presented in [2,3,4,5,6, 7,8,9,10,11,12]. These techniques consider only the frequency domain approximation of given arbitrary specifications. The techniques of spatial domain approximations are not considered in this thesis because these design procedures do not incorporate constraints on filter coefficients, which are required for a stable design [13]. Due to the huge amount of literature in this area, a complete survey of frequency domain design techniques is not attempted here. However, a brief discussion of the existing design techniques with some comparisons are given in the following paragraphs.

Some of the one dimensional techniques given in [1] and the two dimensional design techniques of [10,11,12] are analytical in nature, i.e., the desired type of filters are
obtained by using certain types of transformations (e.g., impulse invariance, bilinear, etc.) on a prototype filter. This, however, provides very little control over the response of the designed filter. Also, if the desired specifications are arbitrary, which is true in many applications such as speech and image processing, then in these situations these methods of design are of very little use. The remaining design techniques reported in [1,2,3,4,5,6,7,8,9] employ either linear or non-linear optimization procedures in designing stable recursive digital filters in which approximations can be carried out to arbitrary specifications. Many of these design techniques design filters to approximate only arbitrary magnitude characteristics; however, this is an incomplete specification of the filter since it is indicated in [1] and also shown by Huang [14], that phase characteristics should be given as much importance as magnitude characteristics. Also in many applications linear phase is important, where dispersion due to non-linear phase characteristics is harmful, specifically in image processing problems. In the case of FIR filters, linear phase filters are easily designed using the design procedures of [14,15], which use linear programming [16]. In the case of existing recursive digital filter design techniques, the linear phase is realized via group delay equalization [1,3], where a non-linear optimization procedure is employed for the approximation. The overall design procedure of [1,3] involves two steps: a) the approximation of magnitude followed by b) group delay equalization which compensates for the non-linearities in the
group delay characteristics of the magnitude only filter. However, there are two drawbacks to using the non-linear optimization approach: a) it requires the specification of initial values for parameters of optimization, b) the optimum obtained in the approximation procedure is a local one rather than the global optimum. In addition to this, as indicated in [1, pp. 288-291], in situations where the group delay characteristics of a magnitude only design are highly non-linear, the equalization is impractical. Furthermore, in the two dimensional case, finding a good set of initial values for the parameters of optimization is much more difficult than in the one dimensional case. However, the non-linear optimization techniques of design given in [1,2,3,4,5,6,7,9] are still useful in situations where other types of approximations (e.g., approximating to the real part of the transfer function or a magnitude squared transfer function approximation) are involved, specifically in the two dimensional case.

In comparison, the use of a linear optimization approach, such as linear programming, not only overcomes the drawbacks of the non-linear optimization procedures, but also it is possible to approximate simultaneously linear phase and arbitrary magnitude characteristics. A characteristic of linear programming that is worth noting is that, if a solution to the problem exists, then it is unique and the optimum obtained is the absolute optimum consistent with the constraints of the problem. As indicated earlier, linear programming has been extensively
used in designing FIR filters where both linear phase and magnitude approximations are carried out simultaneously. Due to its success in the case of FIR filters, linear programming has also been suggested [23] for recursive filter design where simultaneous approximations of linear phase and magnitude may be carried. However, the linear programming methods of designs presented so far [8,17,18,19,20,21] approximate either magnitude, magnitude squared or phase characteristics only. In this dissertation, a method is presented to design both one and two dimensional (quarter plane) recursive digital filters to simultaneously approximate both magnitude and linear phase characteristics, using linear programming. A preliminary investigation of this method by the author is reported in [22].

Another aspect of the research reported in this thesis deals with the applications in image processing area. In recent years, the use of FIR filters has become very common [23,24,25,26,27], specifically with the advent of the fast fourier transform (FFT) [28]. This has not been the case, however, for recursive filters; their uses have been few [29,30], because of difficulty encountered at the early attempts at designing stable filters. As indicated earlier, recent design techniques, including the method to be presented in this thesis, are able to design stable filters and at the same time able to approximate desired frequency domain characteristics. Considering the above, coupled with the computational advantage of the recursive filters compared to FIR filters [31], this dissertation sets out to investigate the use of quarter plane recursive digital filters in
image processing. Some preliminary work carried out by the author, in the applications of recursive digital filters to image processing can be found in [32].

It should be noted here, that at the time this research was carried out, the theory and design of half plane filters were just being proposed and therefore neither the design nor the applications of half plane filters are considered in this thesis.

1.3 Problem Statement

Given an arbitrary magnitude characteristic, the problem of finding coefficients of the recursive digital transfer function using the linear programming approach, where the filter transfer function simultaneously approximates magnitude and linear phase characteristics, is considered in this thesis. The linear programming approach is first applied to the design of one dimensional filters, and then extended to the design of two dimensional quarter plane filters. Since the approximation procedure is based on linear programming, the constraints that are to be used for stable filter design are required to be linear. Therefore the design approach presented also considers various linear stability constraints that can be incorporated in the design technique for stable filters.

Given a blurred image, the thesis considers the inverse filtering problem using recursive filter implementation for restoration from motion, focus and atmospheric turbulence blurs. The thesis also considers the application
of recursive digital filters to image enhancement applications where the problem involves the high frequency emphasis and enhancement of the edges of a given image.

1.4 Thesis Organization

In Chapter II, techniques are presented for the design of both one and two dimensional recursive digital filters. The techniques make use of linear programming for the simultaneous approximation of both magnitude and linear phase characteristics.

Chapter III presents the applications of two dimensional recursive digital filters to image processing problems. The problems considered in the applications area are restricted to image enhancement and restoration.

In Chapter IV, the discussions of the design method and results of applications of recursive digital filters to image processing are presented. In addition, some extensions to the research work presented in this thesis are also discussed.

Finally, Chapter V presents the conclusions that can be obtained from the research work presented in this thesis.
CHAPTER II
RECURSIVE DIGITAL FILTER DESIGN USING LINEAR PROGRAMMING

2.1 Introduction

Linear programming has been widely used in the design of both analog and digital filters. The design of analog filters, using linear programming, was considered by Mathew's et al [33]. Later on, the linear programming approach was used by Rabiner and Hu [25,26] in the design of FIR digital filters. Their design approach included the approximation of both magnitude and linear phase specifications and the designs were carried out in both one and two dimensions. Following this, the linear programming approach was used in the design of one dimensional [17,18,19,20,21] and two dimensional [8] recursive digital filters. However, these techniques designed filters that approximated magnitude or magnitude squared specifications [17,18,19,20] or phase only specification [21] using all pass filters and therefore these methods are of very little use in situations where both magnitude linear phase approximations are required.

In this chapter, a linear programming method for designing recursive digital filters is presented, where a simultaneous approximation of linear phase and arbitrary magnitude response is performed. For completeness, a discussion on the general linear programming technique is presented below.

2.2 Linear Programming [16]

This section briefly introduces the theory of linear programming so as to facilitate the understanding of the linear programming design technique. This includes the con-
cept of dual linear programming, which in many cases becomes useful in reducing the number of variables in the linear program-
ning problem.

2.2.a **Linear Programming Theory**

A linear programming problem can be mathematically stated in the following form - find a vector \( (w_1, w_2, w_3, \ldots w_M) \), subject to the constraints:

\[
\begin{bmatrix}
w_1 \\
w_2 \\
\vdots \\
w_M \\
\end{bmatrix}
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & \cdots & a_{1N} \\
a_{21} & a_{22} & a_{23} & \cdots & a_{2N} \\
\vdots \\
a_{M1} & \cdots & \cdots & \cdots & a_{MN} \\
\end{bmatrix}
\begin{bmatrix}
b_1 \\
b_2 \\
\vdots \\
b_M \\
\end{bmatrix} \leq \begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_N \\
\end{bmatrix};
\]

(\( M < N \))

(2.2.1)

such that,

\[
g = \begin{bmatrix}
w_1 \\
w_2 \\
\vdots \\
w_M \\
\end{bmatrix}
\begin{bmatrix}
b_1 \\
b_2 \\
\vdots \\
b_M \\
\end{bmatrix}
\]

(2.2.2)

is maximized.

Here, the variables \( (w_1, w_2, \ldots w_M) \) may be constrained; for example \( w_i \geq 0, \) \( i = 1, 2, \ldots M. \)

A characteristic of the linear program is that given there is a solution, it is guaranteed to be a unique solution and there are well defined procedures for arriving at this solution within \( (M+N) \) iterations. The procedures also determine if the solution is constrained or unconstrained.
Figure 2.1 Graphical Explanation of the Linear Program.
Figure 2.1 shows the graphical interpretation of the linear program with two variables. Each constraint line \((c_1 - c_5)\) is a linear inequality in two variables, \(w_1\) and \(w_2\). Therefore a straight line can be drawn representing the linear equality, and part of the solution space (shown by shaded lines) is eliminated as a region that does not belong to the possible solution. When all the constraint lines have been drawn, only a small region as shown in Figure 2.1, is admissible as the solution region in which to find the maximum. It should be noted that an important property of the linear programming problem is that if there is to be a solution, the constraint equations should form a polyhedron, and that the maximum or the minimum value of the desired linear function occurs at an extreme point of the polyhedron. Thus the procedure is to compute the value of the objective function at each of the extreme points and choose an extreme point as the solution for which the objective function value is a maximum. The maximum, thus obtained, is the absolute maximum consistent with the constraints in the linear programming problem.

2.2.b **Dual Linear Programming**

The linear programming problem described above can be considered as the 'Primal Problem' of linear programming. Rewriting (2.2.2) and (2.2.1) in the matrix notation, we have:
Maximize

$$\tilde{g} = w^T b$$  \hspace{2cm} (2.2.3)

Subject to constraints

$$w^T A \preceq c^T$$  \hspace{2cm} (2.2.4)\footnote{$c^T$ refers to transpose of matrix vector $c$}

where,

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix} \quad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix}$$

Depending on whether $w$ is constrained or unconstrained, the primal problem given by (2.2.3) and (2.2.4) can assume either of the two types of dual problems\footnote{$c^T$ refers to transpose of matrix vector $c$}, indicated below:

If $w$ is constrained, i.e.,

$$w \succeq 0$$  \hspace{2cm} (2.2.5)

then the dual problem is a 'symmetrical dual linear programming problem' and is stated as:

Minimize

$$f = c^T x$$  \hspace{2cm} (2.2.6)

Subject to

$$Ax \preceq b$$  \hspace{2cm} (2.2.7)

and

$$x \succeq 0$$  \hspace{2cm} (2.2.8)

However, if $w$ is unconstrained (i.e., it can assume both positive and negative values), then the dual problem is an 'unsymmetrical dual linear programming' problem, and is stated as:
Minimize
\[ f = c^T x \]  \hspace{1cm} (2.2.9)

Subject to
\[ A x = b \]  \hspace{1cm} (2.2.10)

and
\[ x > 0 \]  \hspace{1cm} (2.2.11)

It should be noted that the solution of the primal and dual problem can be obtained in the solution of either of the problems because maximum \( g = \) minimum \( f \).

The problem of interest here is the unsymmetrical primal-dual problem, since the design of digital filters can be formulated as a linear programming program given by (2.2.3) and (2.2.4), having no constraints on the variable vector \( w \).

2.3 One Dimensional Recursive Filter Design

In this section, a frequency domain approximation procedure for one dimensional recursive digital filters, using linear programming, is presented. Suitable stability constraints on filter coefficients are indicated for stable designs and an algorithm for designing linear phase filters is described. Details of computations and design examples are also included.

2.3.a Theory of Approximation

Let \( H(Z) \) be the transfer function of a recursive digital filter. Assume \( H(Z) \) has the form,

\[ H(Z) = \frac{P(Z)}{Q(Z)} = \frac{a_0 + a_1 z + a_2 z^2 + \ldots + a_N z^N}{b_0 + b_1 z + b_2 z^2 + \ldots + b_M z^M} \]  \hspace{1cm} (2.3.1)
where \( z = e^{-j\Omega} \) and \( \Omega \) is the normalized frequency variable. The degree of numerator is less than or equal to that of the denominator, i.e., \( N \leq M \). The term \( b_0 \) can be set equal to 1.0 without any loss of generality.

Now, given arbitrary magnitude and phase specifications, it is desired to formulate the digital filter transfer function approximation problem into a linear programming problem such that the constraints of the linear program are in terms of the filter coefficients. This can be carried out as follows.

Let \( R(\Omega_i) \) and \( \Phi(\Omega_i) \) be the given magnitude and phase specifications, respectively, specified at a discrete set of frequency points \( \Omega_i \), \( i = 1, 2, \ldots, L \). The real component \( Y(\Omega_i) \) and the imaginary component \( Y'(\Omega_i) \) of the frequency domain specifications can then be written as:

\[
Y(\Omega_i) = R(\Omega_i) \cdot \left[ \cos \Phi(\Omega_i) \right] \quad (2.3.2)
\]

\[
Y'(\Omega_i) = R(\Omega_i) \cdot \left[ \sin \Phi(\Omega_i) \right] \quad (2.3.3)
\]

Now, the problem of approximating the characteristics of a recursive digital filter to (2.3.2) and (2.3.3) can be formulated into a linear programming problem by following the procedure of Matthews et al [16].

Define \( r(\Omega_i) \) as,

\[
r(\Omega_i) = Y(\Omega_i) + jY'(\Omega_i) - \frac{p(e^{-j\Omega_i})}{Q(e^{-j\Omega_i})};
\]

for \( i = 1, 2, \ldots, L \) \quad (2.3.4)
which is a complex error between the desired characteristics and the approximating filter. Multiplying (2.3.4) on both sides by \( Q(e^{-j\Omega_i}) \), a weighted error function (which is linear in terms of filter coefficients) is obtained, and is given by:

\[
r(\Omega_i)Q(e^{-j\Omega_i}) = Y(\Omega_i)Q(e^{-j\Omega_i}) + jY'(\Omega_i)Q(e^{-j\Omega_i})
\]

\[
- P(e^{-j\Omega_i}) \quad \text{for } i=1,2,\ldots,L
\]  

(2.3.5)

The quantities in the expression (2.3.5) can be separated into their real and imaginary components as:

\[
r(\Omega_i)Q(e^{-j\Omega_i}) = e(\Omega_i) + j e'(\Omega_i)
\]  

(2.3.6)

\[
P(e^{-j\Omega_i}) = P_R(\Omega_i) - j P_I(\Omega_i)
\]  

(2.3.7)

\[
Q(e^{-j\Omega_i}) = Q_R(\Omega_i) - j Q_I(\Omega_i)
\]  

(2.3.8)

Substitution of (2.3.6), (2.3.7) and (2.3.8) into (2.3.5) results in the following:

\[
e(\Omega_i) + j e'(\Omega_i) = Y(\Omega_i)[Q_R(\Omega_i) - j Q_I(\Omega_i)] + j Y'(\Omega_i)[Q_R(\Omega_i)
\]

\[
- j Q_I(\Omega_i)] - P_R(\Omega_i) + j P_I(\Omega_i)
\]  

(2.3.9)

Equating the real and imaginary parts in expression of (2.3.9) results in:

\[
e(\Omega_i) = Y(\Omega_i)Q_R(\Omega_i) + Y'(\Omega_i)Q_I(\Omega_i) - P_R(\Omega_i)
\]  

(2.3.10)

\[
e'(\Omega_i) = Y'(\Omega_i)Q_R(\Omega_i) - Y(\Omega_i)Q_I(\Omega_i) + P_I(\Omega_i)
\]  

(2.3.11)
where \( e(\Omega_i) \) and \( e'(\Omega_i) \) are linear in terms of filter coefficients.

It is now possible to put Equations (2.3.10) and (2.3.11) into a linear programming problem such that the weighted error in (2.3.6) is minimized.

Choose a quantity \( \varepsilon \) such that,

\[
|e(\Omega_i)| \leq \varepsilon \tag{2.3.12}
\]

and

\[
|e'(\Omega_i)| \leq \varepsilon, \text{ for } i=1, 2, \ldots L \tag{2.3.13}
\]

Expressions (2.3.12) and (2.3.13) can be re-written as,

\[
-\varepsilon \leq e(\Omega_i) \leq \varepsilon \tag{2.3.14}
\]

and,

\[
-\varepsilon \leq e'(\Omega_i) \leq \varepsilon \tag{2.3.15}
\]

From the above, one can write four inequalities as follows:

\[
e(\Omega_i) - \varepsilon \leq 0
\]

\[
e'(\Omega_i) - \varepsilon \leq 0
\]

\[
-e(\Omega_i) - \varepsilon \leq 0
\]

\[
-e'(\Omega_i) - \varepsilon \leq 0
\]

Let \( \xi = -\varepsilon \), in which case the inequalities are as follows:

\[
e(\Omega_i) + \xi \leq 0
\]

\[
e'(\Omega_i) + \xi \leq 0
\]

\[
-e(\Omega_i) + \xi \leq 0
\]

\[
-e'(\Omega_i) + \xi \leq 0
\]
After substitution of \(e(\Omega_i)\) and \(e'(\Omega_i)\) from (2.3.10) and (2.3.11) into (2.3.17), the linear programming problem for filter design can be stated in a form similar to (2.2.1) and (2.2.2), as,

Maximize \[ g = \xi \]

Subject to,

\[
\begin{align*}
Y(\Omega_i)Q_R(\Omega_i) + Y'(\Omega_i)Q_I(\Omega_i) - P_R(\Omega_i) + \xi & \leq 0 \quad (2.3.18) \\
Y'(\Omega_i)Q_R(\Omega_i) - Y(\Omega_i)Q_I(\Omega_i) + P_I(\Omega_i) + \xi & \leq 0 \quad (2.3.19) \\
-Y(\Omega_i)Q_R(\Omega_i) - Y'(\Omega_i)Q_I(\Omega_i) + P_R(\Omega_i) + \xi & \leq 0 \quad (2.3.20) \\
-Y'(\Omega_i)Q_R(\Omega_i) + Y(\Omega_i)Q_I(\Omega_i) - P_I(\Omega_i) + \xi & \leq 0 \quad (2.3.21)
\end{align*}
\]

Maximizing \(\xi\), minimizes the weighted errors in (2.3.10) and (2.3.11).

It can be clearly seen from (2.3.17) that in the ideal situation, the maximum value of \(\xi\) can be equal to zero. Also, \(\xi\) cannot be positive, and if it does become positive, the solution is meaningless. Thus the upperbound on \(\xi\) is zero. A careful examination of the approximation procedure reveals that the minimization of the weighted errors in the error set,

\[
\left[ e(\Omega_i), e'(\Omega_i) \right] ; i=1,2,...,L
\]

is performed in the mini-max sense. Therefore the absolute value of \(\xi\) is equal to the magnitude of the largest deviation from zero of the real component \(e(\Omega_i)\) or the imaginary component \(e'(\Omega_i)\) of the weighted error.
The constraints, (2.3.18) through (2.3.21), are sufficient to carry out an approximation to the desired specifications. If the designed filter is desired to be stable, however, additional constraints will be needed. Since the approximation procedure involves linear programming, these additional constraints will have to be linear in form. The general form of these constraints could be as follows:

\[ \sum_{m=0}^{M} f_m(b_m, \Omega_i) \leq 0; \quad i=1,2,\ldots,L \quad (2.3.22) \]

The inequalities in (2.3.18) through (2.3.22) can be simplified as follows:

Let

\[ Q_R(\Omega_i) = 1 + \sum_{m=1}^{M} b_m \cos(m\Omega_i) \quad (2.3.23) \]

\[ Q_I(\Omega_i) = \sum_{m=1}^{M} b_m \sin(m\Omega_i) \quad (2.3.24) \]

\[ P_R(\Omega_i) = 1 + \sum_{n=0}^{N} a_n \cos(n\Omega_i) \quad (2.3.25) \]

\[ P_I(\Omega_i) = \sum_{n=0}^{N} a_n \sin(n\Omega_i) \quad (2.3.26) \]

\[ \sum_{m=0}^{M} f_m(b_m, \Omega_i) \leq 0 \quad (2.3.27) \]

Substituting inequalities (2.3.23) through (2.3.26) into inequalities (2.3.18) through (2.3.21) and simplifying, one can rewrite the linear programming problem as:
Maximize

\[ g = \xi \]  \hspace{1cm} (2.3.28)

Subject to

\[ \sum_{m=1}^{M} b_m \left( Y(\Omega_i) \cos(m\Omega_i) + Y'(\Omega_i) \sin(m\Omega_i) \right) - \sum_{n=0}^{N} a_n \cos(n\Omega_i) + \xi \leq -Y(\Omega_i) \] \hspace{1cm} (2.3.29)

\[ \sum_{m=1}^{M} b_m \left( Y'(\Omega_i) \cos(m\Omega_i) - Y(\Omega_i) \sin(m\Omega_i) \right) + \sum_{n=0}^{N} a_n \sin(n\Omega_i) + \xi \leq -Y'(\Omega_i) \] \hspace{1cm} (2.3.30)

\[ -\sum_{m=1}^{M} b_m \left( Y(\Omega_i) \cos(m\Omega_i) + Y'(\Omega_i) \sin(m\Omega_i) \right) + \sum_{n=0}^{N} a_n \cos(n\Omega_i) + \xi \leq Y(\Omega_i) \] \hspace{1cm} (2.3.31)

\[ -\sum_{m=1}^{M} b_m \left( Y'(\Omega_i) \cos(m\Omega_i) - Y(\Omega_i) \sin(m\Omega_i) \right) - \sum_{n=0}^{N} a_n \sin(n\Omega_i) + \xi \leq Y'(\Omega_i) \] \hspace{1cm} (2.3.32)

and

\[ \sum_{m=1}^{M} b_m f_m(\Omega_i) < 0 \]

which completely defines the linear programming design problem for a one dimensional recursive digital filter.
2.3.b Stability Constraints in One Dimension

As indicated earlier, because of the use of linear programming in the approximation procedure, the constraints that can be used to design stable filters have to be linear in form. There are two types of constraints that readily satisfy the above requirement. These are as follows:

(a) Monotonicity of denominator filter coefficients \[34\],

\[ \text{i.e., } b_0 > b_1 > b_2 > \ldots > b_{n-1} > b_n > 0 \quad (2.3.33) \]

where \( b_0 \) can be equal to 1. As shown in \( [34] \), the above constraint ensures that all the roots of \( Q(z) \) lie outside the unit circle \( |z| = 1 \) and hence ensures the stability of the transfer function \( H(z) \).

(b) Real part of the denominator polynomial greater than zero on the unit circle \( [35] \),

\[ \text{i.e., } \Re\{Q(z)\} > 0 \text{ for } |z| = 1 \quad (2.3.34) \]

This also ensures stability of \( H(z) \), if (2.3.33) is specified over the unit circle in the \( z \) plane.

The constraint of (b) has been successfully used in the design of all pass digital filters \( [21] \). It should, however, be noted that these constraints are just sufficient conditions for stability (the proof of the sufficiency of (b) is given in Appendix C). Also, since these constraints are just sufficient conditions for stability, they generate a subset of possible stable filter realizations.

From a closer examination of the two possible types
of constraints that can be used in the design, it appears that the latter of the two constraints yields a larger subclass of stable filters compared to the other. This can at least be shown to be true in the case of filters whose order is less than or equal to 2.

Proof for the First Order Filter

Let

$$Q(Z) = 1 + b_1 Z$$  \hspace{1cm} (2.3.35)

where $Z = e^{-j\Omega}$ and $\Omega$ is the normalized frequency variable.

Use of constraint (a) results in the following:

$$1 > b_1 > 0$$  \hspace{1cm} (2.3.36)

Use of type (b) constraint results in:

$$1 + b_1 \cos\Omega > 0, \ 0 < \Omega < \pi$$ \hspace{1cm} (2.3.37)

From (2.3.37) it is clear that $b_1$ is constrained to be as follows:

$$-1 < b_1 < 1$$

Consider now a first order transfer function $H(Z)$, such that,

$$H(Z) = \frac{P(Z)}{Q(Z)}$$

where $Q(Z)$ is given by (2.3.35). Therefore the pole of $H(Z)$ is at $Z = -1/b$. From Figure 2.2 it can be observed that the application of the constraint (2.3.36) restricts the poles of the transfer function $H(Z)$ to the left half of the real axis in the $Z$ plane. On the other hand, if (2.3.37) is used
Figure 2.2 Constraint Regions for Poles of 1st Order Transfer Function.
then this results in $H(z)$ having poles on the entire real axis except for the portion which is inside and on the unit circle. Thus, constraint type (b) yields a larger class of stable filters compared to type (a) constraint.

In the second order example, (which is shown in Appendix D) although the proof is not so rigorous, it can still be seen that the type (b) constraint yields a larger subclass of stable filters compared to that of type (a). Therefore in all the design examples, the constraint type (b) has been used to design stable filters. The actual stability constraint that is used in the design is a slightly modified form of (2.3.34) and is given by,

$$\text{Re}\{Q(z)\} > \Delta C \text{ for } |z| = 1 \quad (2.3.38)$$

where $\Delta C$ is a small positive quantity.

Since $\text{Re}\{Q(z)\} = 1 + \sum_{m=1}^{M} b_m \cos(m\Omega_i)$, (3.3.38) can be rewritten as,

$$\sum_{m=1}^{M} b_m \cos(m\Omega_i) > \Delta C - 1; \ 0 < \Omega_i < \pi \quad (2.3.39)$$

or

$$\sum_{m=1}^{M} b_m \left[ -\cos(m\Omega_i) \right] < 1 - \Delta C; \ 0 < \Omega_i < \pi \quad (2.3.40)$$

Proper choice of $\Delta C$ and a proper number of points at which the constraint (2.3.40) is specified can ensure a stable filter design.
2.3.c Design Procedure

The linear programming approach of filter design outlined in the previous section is now used in the design of linear phase filters. The desired linear phase characteristics can be specified in terms of the spatial delay.

Let \( \phi(\Omega) \) be the phase characteristics in the frequency domain. As indicated in Appendix A, the delay characteristics \( \tau(\Omega) \) is given by:

\[
\tau(\Omega) = -\frac{d \phi(\Omega)}{d\Omega}
\]  \hspace{1cm} (2.3.41)

Therefore, if \( \phi(\Omega) \) is desired to be linear with respect to the frequency \( \Omega \), then \( \tau(\Omega) \) needs to be a constant. Letting \( \tau(\Omega) = \tau_s \), where \( \tau_s \) is a constant, the desired linear phase characteristics can be written as:

\[
\phi(\Omega) = -\tau_s \Omega
\]  \hspace{1cm} (2.3.42)

Using (2.3.42), the real and imaginary components of the desired specifications given in (2.3.2) and (2.3.3) can be written as:

\[
\begin{align*}
Y(\Omega_i) &= R(\Omega_i) \cos(-\tau_s \Omega_i) \\
Y'(\Omega_i) &= R(\Omega_i) \sin(-\tau_s \Omega_i); \ i = 1, 2, \ldots L
\end{align*}
\]  \hspace{1cm} (2.3.43)

The linear programming problem can therefore be rewritten as:

Maximize

\[
g = \zeta
\]

Subject to the following constraints:
Thus given an arbitrary magnitude characteristics, it is now possible to approximate these characteristics together with a linear phase specification (specified in terms of delay $\tau_s$) by a recursive filter of specified order. A good design is normally obtained for a particular value of $\tau_s$ that lies within a certain range of values. This range can be determined by examining the impulse response corresponding to the given arbitrary magnitude characteristics. Figure 3.3 shows the range of $\tau_s$ in the impulse response.
Figure 2.3  Range of \( \tau_s \) in the Impulse Response.
corresponding to low pass type magnitude characteristics. Searching for a proper $\tau_s$ in the entire range may result in excessive design time. However, given the order of the filter, one can specify appropriate upper and lower limits for $\tau_s$. Thus if $K$ is chosen as the order of filter, then an appropriate range for $\tau_s$ is $(K-5) \leq \tau_s \leq K$. An argument as to why the upper limit for $\tau_s$ is equal to $K$ can be understood from the following.

Consider an all pass transfer function of order $K$. A property of this function is that $[1]$,

$$\int_{0}^{\pi} \tau(\Omega) d\Omega = K\pi$$

From the above, it can be seen that to realize a constant delay exactly equal to $K$, up to the nyquist rate, the filter order should at least be equal to $K$. Therefore, in the case of a general recursive digital filter, the order of the filter should at least be greater than or equal to $K$, if it is to realize, approximately, a constant delay $K$ and also at the same time approximate given arbitrary magnitude specifications.

The lower limit for $\tau_s$ is chosen to be equal to $(K-5)$, although the ideal lower limit can be equal to 1. However, choice of the lower limit, indicated above has not only reduced the excessive design time but also has provided consistently good results, as shown in the examples later.

A suitable procedure for designing linear phase filters is as follows:

1) Specify the magnitude characteristics; i.e., $R(\Omega)$;
2) Choose the filter order; say \( K \) and range of \( \tau_s \) as indicated above;

3) Solve the linear programming problem for each \( \tau_s \) starting from upper limit of \( K \) (in descending order) until a maximum in \( \xi \) is obtained. Denote the corresponding \( \tau_s \) as \( \tau_{s_{\text{max}}} \);

4) Choose the coefficients of the filter for which maximum \( \xi \) was obtained and compute frequency response and errors.

The error measures in step 4) are a) the squared error sum in magnitude, \( E_1 \), which is given by:

\[
E_1 = \sum_{i=1}^{L} \left[ R(\Omega_i) - |H(\Omega_i)| \right]^2 \tag{2.3.49}
\]

b) the squared error sum in the delay characteristics in the pass band of the filter, \( E_2 \), given by:

\[
E_2 = \sum_{\text{(Pass Band)}} \left[ \tau_{s_{\text{max}}} - \tau(\Omega_i) \right]^2 \tag{2.3.50}
\]

\( R(\Omega_i) \) and \( \tau_{s_{\text{max}}} \) are the desired magnitude characteristics and the constant group delay and \( |H(\Omega_i)| \) and \( \tau(\Omega_i) \) are the magnitude and overall group delay characteristics of the designed filter at the specified discrete frequency points \( \Omega_i \).

As indicated above, the error measure incorporates the error in delay characteristics rather than the error in phase. A point to note is that the degree of linearity of the phase can be observed much more clearly using group delay characteristics since they are the differential of phase characteristics. Also, the error \( E_2 \) in (2.3.50) is computed only over the pass band of the filter instead of computing over all the frequency points of specification,
because, any non-linearity in phase or group delay of the
filter is of little consequence outside the pass band re-
gion, since in this region the magnitude of the filter
characteristics is not significant.

Using the procedure described above, a large number
of filters have been designed and the examples of these
designs are presented in Section 2.3.e.

2.3.d Computational Considerations

Reconsider the linear programming design problem of
(2.3.27) through (2.3.32). This problem can be written in
the matrix notation of section (2.2.a) as follows.

Find a vector \( (b_1, b_2, \ldots, b_M, a_0, a_1, \ldots, a_N, \xi) \),
such that

\[
\begin{bmatrix}
 b_1 & b_2 & \cdots & b_M & a_0 & a_1 & \cdots & a_N & \xi
\end{bmatrix}
\begin{bmatrix}
 D
\end{bmatrix}
\leq
\begin{bmatrix}
 c
\end{bmatrix}
\]

(2.3.51)

such that

\[
g = \begin{bmatrix}
 b_1 & b_2 & \cdots & b_M & a_0 & a_1 & \cdots & a_N & \xi
\end{bmatrix}
\begin{bmatrix}
 0 \\
 0 \\
 \vdots \\
 0 \\
 1
\end{bmatrix}
\]

(2.3.52)

is maximized. The matrix D is made up of several submatrices
as given below:
Matrices $D_1$ and $D_2$ are of size $M \times L$ and their elements are given by:

$$d_{1ij} = Y(\Omega_j) \cos [i\Omega_j] + Y(\Omega_j) \sin [i\Omega_j]$$  \hspace{1cm} (2.3.54)$$

$$d_{2ij} = Y'(\Omega_j) \cos [i\Omega_j] - Y(\Omega_j) \sin [i\Omega_j]$$ \hspace{1cm} (2.3.55)$$

$i=1,2,\ldots, M$; $j=1,2,\ldots, L$

Matrices $D_3$ and $D_4$ are of size $(N+1) \times L$ and their elements are given by:

$$d_{3ij} = -\cos [(i-1)\Omega_j]$$ \hspace{1cm} (2.3.56)$$

$$d_{4ij} = \sin [(i-1)\Omega_j]$$ \hspace{1cm} (2.3.57)$$

$i=1,2,\ldots, (N+1)$; $j=1,2,\ldots, L$

Matrix $D_9$ is the stability constraint matrix corresponding to the stability constraint given by (2.3.40). The size of the matrix is $M \times L_1$, where $L_1$ may be greater than or equal to $L$. The elements of this matrix are given by:

$$d_{9ij} = -\cos(i\Omega_j);$$ \hspace{1cm} (2.3.58)$$

$i=1,2,\ldots, M$; $j=1,2,\ldots, L_1$
D_{11}, D_{12} and D_{13} are row matrices. Matrix D_{11} is of size 1 x (2 \cdot M) and is given by:

\[ d_{11,i} = 1 ; i=1,2,\ldots,(2 \cdot M) \] (2.3.59)

The matrices D_{10} and D_{13} which are of size (N+1) x L_1 and 1 x L_1 respectively, are equal to zero. The rest of the submatrices of D are as follows:

\[ D_5 = -D_1 ; D_6 = -D_2 \] (2.3.60)

\[ D_7 = -D_3 ; D_8 = -D_4 \]

and

\[ D_{12} = D_{11} \] (2.3.61)

The row matrix C also contains submatrices as shown below:

\[ C = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \end{bmatrix} \] (2.3.62)

Matrices c_1 through c_4 are of size 1 x L and matrix c_5 is of size 1 x L_1. Their elements are as follows:

\[ c_{1,i} = -Y(\Omega_i); c_{2,i} = -Y'(\Omega_i); i=1,2,\ldots,L \] (2.3.63)

\[ c_{5,i} = 1-\Delta C \hspace{1cm}; i=1,2,\ldots,L_1 \] (2.3.64)

and

\[ c_3 = -c_1 \text{ and } c_4 = -c_2 \] (2.3.65)

where Y(\Omega) and Y'(\Omega) are real and imaginary components respectively of the desired specifications.
As can be seen from (2.3.60), (2.3.61) and (2.3.65), the coefficient matrix $D$, and matrix $C$ have submatrices whose elements are redundant. Also submatrices $D_{10}$ and $D_{11}$ are null matrices. This property can therefore be made use of in reducing the storage requirements.

The linear programming problem of (2.3.51) and (2.3.52) may be solved by a straightforward application of the simplex [16] (or the revised simplex [16]) algorithm. However, this involves a large number of computations because of the large number of variables involved. In the straightforward problem, there are $4L_1$ inequalities with $(M+N+2)$ unknowns where $(M+N+1)$ are denominator and numerator coefficients of the filter. These coefficient variables are unconstrained and so they have to be replaced by the difference of two positive variables, as the simplex algorithm requires that the variables be greater than or equal to zero. Also each inequality in the problem is replaced by an equality by adding a slack variable. Thus the resulting linear program will have $(4L_1 + 2M + 2N + 1)$ variables. For large order filters this would involve very large amounts of computation. To avoid this problem, one can turn to the dual linear program. In the dual system, which is similar to the system of (2.2.10) through (2.2.12), after the addition of artificial variables to each equality, the result is $(4L_1 + M + N + 1)$ variables. This is a smaller number of variables than the primal problem and can be more easily handled.

Finally, there does not exist any rule for choosing the
number of sample frequencies (i.e., the value for \( L_1 \)) at which to specify the desired specifications, although it must be large enough to ensure that the response is well represented. Brophy and Salazar [21] suggest that \( L_1 \) be greater than four times the order of the filter. Therefore, if \( K \) is the order of the filter, then \( L_1 > 4K \). The stability constraint is also specified over 4K (or greater) number of points, so as to reasonably ensure the stability of the filter.

3.3.e **Examples of Design**

In the examples presented here, the desired characteristics are specified over 81 equally spaced discrete frequency points. The stability constraint is also specified over the same frequency points. In plotting the frequency response of the designed filters, the phase response is eliminated in favour of the group delay response, since the non-linearities in phase can be observed more clearly in the group delay response.

**Example 1 - Band Pass Filter**

The desired specifications are as follows:

- The lower pass band frequency \( \Omega_{p_1} = 0.35\pi \)
- The upper pass band frequency \( \Omega_{p_2} = 0.55\pi \)

The complete magnitude specification \( R(\Omega_i) \) is given by:

\[
R(\Omega_i) = e^{-k(\Omega_i - \Omega_{p_1})^2} \quad \text{for} \quad 0 < \Omega_i < \Omega_{p_1}
\]
\[
R(\Omega_i) = 0 \quad \text{for} \quad \Omega_{p_1} < \Omega_i < \Omega_{p_2}
\]
\[
R(\Omega_i) = e^{-k(\Omega_i - \Omega_{p_2})} \quad \text{for} \quad \Omega_{p_2} < \Omega_i < \pi; i=1,2,\ldots,L
\]
Figure 2.4 Desired Magnitude Characteristics.
Figure 2.5 Designed Magnitude Characteristics
Figure 2.6 Designed Group Delay Characteristics
Figure 2.7 Designed Impulse Response
THE ZEROS OF THE FILTER ARE:-

ZERO(1) = 3294084E 20 + 5418085E 00
ZERO(2) = 3294084E 20 + 5418085E 00
ZERO(3) = -3989847E 00 + 0000000E 00
ZERO(4) = 1203671E 00 + 7476057E 00
ZERO(5) = 1203671E 00 + 7476057E 00
ZERO(6) = -8963023E 00 + 3264588E 00
ZERO(7) = -8963023E 00 + 3264588E 00
ZERO(8) = 4783893E 00 + 0000000E 00
ZERO(9) = -5808388E 00 + 7498324E 00
ZERO(10) = -5808388E 00 + 7498324E 00
ZERO(11) = 9329610E 00 + 0000000E 00
ZERO(12) = -7336164E 00 + 6195835E 00
ZERO(13) = -7336164E 00 + 6195835E 00
ZERO(14) = 8960102E 00 + 3098137E 00
ZERO(15) = 8960102E 00 + 3098137E 00
ZERO(16) = -4304536E 01 + 0000000E 00

THE POLES OF THE FILTER ARE:-

POLE(1) = 6356730E 00 + 1080559E 01
POLE(2) = 6356730E 00 + 1080559E 01
POLE(3) = -3536178E 00 + 1139319E 01
POLE(4) = -3536178E 00 + 1139319E 01
POLE(5) = 1257123E 01 + 6273603E 00
POLE(6) = 1257123E 01 + 6273603E 00
POLE(7) = -8697155E 00 + 9954215E 00
POLE(8) = -8697155E 00 + 9954215E 00
POLE(9) = -1533266E 00 + 1326754E 01
POLE(10) = -1533266E 00 + 1326754E 01
POLE(11) = -1084559E 01 + 6287196E 00
POLE(12) = -1084559E 01 + 6287196E 00
POLE(13) = 2502317E 00 + 1306255E 01
POLE(14) = 2502317E 00 + 1306255E 01
POLE(15) = 7983543E 00 + 9624366E 00
POLE(16) = 7983543E 00 + 9624366E 00

Figure 2.8 Pole-Zero Positions of the Designed Filter
The filter order was chosen to be 16. A maximum $\xi$ was obtained for group delay value of 13. The value of $\Delta C$ (in the constraint of (3.3.38)) and $k$, chosen for this design are $1 \times 10^{-6}$ and 20.0 respectively. The error measures of the design are as follows:

$$E_1 = 1.6 \times 10^{-2}, \quad E_2 = 1.48$$

The value of $\xi$ (the variable that is maximized in the linear program) is equal to $-7.8 \times 10^{-3}$. The desired magnitude specification is shown in Figure 2.4. The designed filter characteristics with its impulse response and pole-zero positions are shown in Figures 2.5 through 2.8.

**Example 2 - Design of a Differentiator**

The characteristics of an ideal differentiator is given by:

$$H(\Omega) = j \left( \frac{\Omega}{\Omega_n} \right)$$

where $\Omega_n$ is half the sampling frequency in radians. From the magnitude-phase plot of $H(\Omega)$ shown in Figure 2.9(a) and (b), it can be seen that the phase characteristic is discontinuous at one-half the sampling frequency. This characteristic is difficult to realize because of the discontinuity at half the sampling frequency. However, with the addition of a one-half sample delay, this discontinuity can be eliminated [1] as shown in Figure 2.9(c) and (d). The discontinuity at the origin is of no consequence because the magnitude is zero at zero frequency. Thus the desired magnitude and phase characteristics
Figure 2.9 Realization of Differentiator With Half Sample Delay.
Figure 2.10  Desired Magnitude Characteristics.
Figure 2.11 Designed Filter Magnitude Characteristics.
Figure 2.12: Designed Filter Group Delay Characteristics.
Figure 2.13 Error in the Magnitude
Figure 2.14 Impulse Response of the Designed Filter.
THE ZEROS OF THE FILTER ARE:

ZERO(1) = 1E6E000E 01 +J 0000000E 00
ZERO(2) = -56E1700E 00 +J 1726286E 00
ZERO(3) = -56E1700E 00 +J 1726286E 00
ZERO(4) = -56E1700E 00 +J 1726286E 00
ZERO(5) = -56E1700E 00 +J 1726286E 00
ZERO(6) = 250E626E 00 +J 0000000E 00
ZERO(7) = -250E626E 00 +J 0000000E 00
ZERO(8) = -430E282E 00 +J 0000000E 00
ZERO(9) = 430E282E 00 +J 0000000E 00
ZERO(10) = 4431354E 00 +J 3580827E 00
ZERO(11) = 4431354E 00 +J 3580827E 00
ZERO(12) = -2182096E 00 +J 5263062E 00
ZERO(13) = -2182096E 00 +J 5263062E 00
ZERO(14) = 0000000E 01 +J 0000000E 00
ZERO(15) = 1962210E-01 +J 5629674E 00
ZERO(16) = -1962210E-01 +J 5629674E 00
ZERO(17) = -3175194E 01 +J 0000000E 00

THE POLES OF THE FILTER ARE:

POLE(1) = -101E8085E 01 +J 0000000E 00
POLE(2) = -1530243E 01 +J 0000000E 00
POLE(3) = 1760998E 01 +J 0000000E 00
POLE(4) = 1510873E 01 +J 6381251E 00
POLE(5) = 1510873E 01 +J 6381251E 00
POLE(6) = 6182000E 00 +J 1073534E 01
POLE(7) = 6182000E 00 +J 1073534E 01
POLE(8) = 1220490E 01 +J 0000000E 00
POLE(9) = 1220490E 01 +J 0000000E 00
POLE(10) = 7249826E 00 +J 1613375E 01
POLE(11) = 7249826E 00 +J 1613375E 01
POLE(12) = 1624080E 01 +J 0000000E 00
POLE(13) = 1624080E 01 +J 0000000E 00
POLE(14) = 1230965E 01 +J 1194754E 01
POLE(15) = 1230965E 01 +J 1194754E 01
POLE(16) = 7138175E-01 +J 1800103E 01
POLE(17) = 7138175E-01 +J 1800103E 01

Figure 2.15 Pole-Zero Positions of the Designed Filter.
of the differentiator are:

\[ R(\Omega_i) = \frac{\Omega_i}{\pi}; \quad 0 \leq \Omega_i \leq \pi \]

and the phase

\[ \phi(\Omega_i) = 0.5\pi - (\tau_s + 0.5)\Omega_i; \quad 0 \leq \Omega_i \leq \pi \]

\( \tau_s \) is the amount desired delay in samples for linear phase design. The total amount group delay realized is therefore equal to \((\tau_s + 0.5)\).

A filter of order 17 was chosen and \( \Delta C \) was set equal to \( 1 \times 10^{-6} \). A maximum \( \xi \) was obtained for a group delay value of 14. In computing the error measure for group delay, the error at \( \Omega = 0 \) was disregarded, as it is of no consequence as the magnitude of the designed differentiator is almost zero at this frequency. The error measures for this design are as follows:

\[ E_1 = 3.446 \times 10^{-4} \quad \text{and} \quad E_2 = 6.8 \times 10^{-2} \]

The value of \( \xi \) is equal to \( 1.65 \times 10^{-4} \). The desired magnitude of the ideal differentiator is shown in Figure 2.10. The designed response, pole-zero position and various errors are shown in Figures 2.11 through 2.15.

2.4 Two-Dimensional Recursive Filter Design

In this section, the linear programming method of design discussed in the preceding section for the one-dimensional filter is extended to the design of two dimensional recursive digital filters. As in the one dimensional
case, the two dimensional approximation problem is made linear, so as to facilitate use of linear programming in the design. A linear stability constraint, similar to the one dimensional case, is proposed and is used in the design of stable filters.

2.4.a Theory of Approximation

Let \( H(Z_1, Z_2) \) be the transfer function of a two dimensional recursive digital filter. Assume \( H(Z_1, Z_2) \) is of the form:

\[
H(Z_1, Z_2) = \frac{P(Z_1, Z_2)}{Q(Z_1, Z_2)} = \sum_{m=0}^{M_1} \sum_{n=0}^{N_1} a_{mn} Z_1^m Z_2^n
\]

\[
\sum_{m=0}^{M_2} \sum_{n=0}^{N_2} b_{mn} Z_1^m Z_2^n
\]

where,

\[
Z_1 = e^{-j\Omega_1} \quad \text{and} \quad Z_2 = e^{-j\Omega_2}
\]

\(-\pi \leq \Omega_1 \leq \pi \) and \(-\pi \leq \Omega_2 \leq \pi\)

\(\Omega_1\) and \(\Omega_2\) are the normalized frequency variables. The term \(b_{00}\) can be set equal to 1.0 without any loss of generality.

Now given an arbitrary magnitude and phase specification, it is possible, in a manner similar to the one dimensional case, to formulate a linear programming problem such that the constraints of the linear program are in terms of the filter coefficients. Let \( R(\Omega_{1i}, \Omega_{2j}) \) and \( \phi(\Omega_{1i}, \Omega_{2j}) \) be the given magnitude and phase specifications respectively, specified over a frequency grid \( \Omega_{1i}, \Omega_{2j} \).
i = 1, 2, ..., L1, j = 1, 2, ..., L2. As before, the real components \( Y(\Omega_{li}, \Omega_{2j}) \) and the imaginary components \( Y'(\Omega_{li}, \Omega_{2j}) \) of the frequency domain specifications can be written as:

\[
Y(\Omega_{li}, \Omega_{2j}) = R(\Omega_{li}, \Omega_{2j}) \cos(\phi(\Omega_{li}, \Omega_{2j})) \quad (2.4.2)
\]

\[
Y'(\Omega_{li}, \Omega_{2j}) = R(\Omega_{li}, \Omega_{2j}) \sin(\phi(\Omega_{li}, \Omega_{2j})) \quad (2.4.3)
\]

From this point onwards, the mathematical manipulations are similar to the one dimensional design. Thus, in a similar manner, the two dimensional approximation problem can be stated as:

Maximize,

\[ \lambda g = \xi \]

Subject to

\[
Y(\Omega_{li}, \Omega_{2j}) Q_R(\Omega_{li}, \Omega_{2j}) + Y'(\Omega_{li}, \Omega_{2j}) Q_I(\Omega_{li}, \Omega_{2j}) - P_R(\Omega_{li}, \Omega_{2j})
\]

\[ + \xi < 0 \quad (2.4.5) \]

\[
Y'(\Omega_{li}, \Omega_{2j}) Q_R(\Omega_{li}, \Omega_{2j}) - Y(\Omega_{li}, \Omega_{2j}) Q_I(\Omega_{li}, \Omega_{2j}) + P_I(\Omega_{li}, \Omega_{2j})
\]

\[ + \xi < 0 \quad (2.4.6) \]

\[
- Y(\Omega_{li}, \Omega_{2j}) Q_R(\Omega_{li}, \Omega_{2j}) + Y'(\Omega_{li}, \Omega_{2j}) Q_I(\Omega_{li}, \Omega_{2j}) + P_R(\Omega_{li}, \Omega_{2j})
\]

\[ + \xi < 0 \quad (2.4.7) \]

\[
-Y'(\Omega_{li}, \Omega_{2j}) Q_R(\Omega_{li}, \Omega_{2j}) + Y(\Omega_{li}, \Omega_{2j}) Q_I(\Omega_{li}, \Omega_{2j}) - P_I(\Omega_{li}, \Omega_{2j})
\]

\[ + \xi < 0 \; ; \quad (2.4.8) \]

\[ i = 1, 2, 3, ..., L1; \; j = 1, 2, ..., L2 \]
where $Q_R$, $Q_I$ and $P_R$, $P_I$ are the real and imaginary components of $Q$ and $P$ respectively. Also, linear constraints on the denominator coefficients, of the form:

$$
\sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} f_{mn}(\Omega_1, \Omega_2) \leq 0
$$

(2.4.9)

are incorporated to obtain stable designs. Thus the above, completely defines the linear programming design problem for the two dimensional case.

2.4.b Stability Constraints in Two Dimensions

The stability constraint used in the two dimensional design procedure is similar to the constraint used in the one dimensional filter design. It is given by:

$$\text{Re}\{Q(Z_1, Z_2)\} > 0 \text{ for } |Z_1| = 1 \text{ and } |Z_2| = 1 \quad (2.4.10)$$

where $Q(Z_1, Z_2)$ is the denominator polynomial of the two-D transfer function. This constraint is linear in terms of the coefficients $b(m, n)$ of the denominator polynomial $Q(Z_1, Z_2)$ and therefore it can be easily incorporated into the linear programming design procedure.

It should be noted here that the constraint of (2.4.10) is not a general stability constraint; however, it is a sufficient condition for $Q(Z_1, Z_2)$ to be a stable denominator polynomial of a two-D filter transfer function (the proof of the sufficiency is given in Appendix E) and therefore the use of this constraint in the filter design generates a subset of possible stable filters. This is not a drawback when
compared to the existing design methods [2, 3, 4, 5, 6, 7] where the designed filters also belong to a subclass of possible stable filters.

2.4.c Design Procedure

The design procedure is similar to the one dimensional case. As in the one dimensional case, the linear phase characteristic is specified in terms of the spatial delays. As indicated in Chapter I, in the two dimensional case, the spatial delays are defined as:

\[ \tau_1(\Omega_1, \Omega_2) = -\frac{\partial \phi(\Omega_1, \Omega_2)}{\partial \Omega_1} \]  \hspace{1cm} (2.4.11)

and,

\[ \tau_2(\Omega_1, \Omega_2) = -\frac{\partial \phi(\Omega_1, \Omega_2)}{\partial \Omega_2} \]  \hspace{1cm} (2.4.12)

where \( \phi(\Omega_1, \Omega_2) \) is the two dimensional phase characteristic in the frequency domain. Thus if the desired phase characteristics is to be linear, then \( \tau_1(\Omega_1, \Omega_2) \) and \( \tau_2(\Omega_1, \Omega_2) \) are assigned constant values. By setting \( \tau_1(\Omega_1, \Omega_2) \) equal to a constant \( \tau_x \) and \( \tau_2(\Omega_1, \Omega_2) \) equal to a constant \( \tau_y \), the desired linear phase characteristics can be written as:

\[ \phi(\Omega_1, \Omega_2) = - (\tau_x \Omega_1 + \tau_y \Omega_2) \]  \hspace{1cm} (2.4.13)

The specifications can be simplified by setting \( \tau_x = \tau_y = \tau_c \); i.e., realizing the same amount of delay in both directions in the spatial domain. The phase characteristics is then given by:

\[ \tau \]  \hspace{1cm} (2.4.13)

*This is group delay referred to space rather than time.*
\[ \phi(\Omega_1, \Omega_2) = -\tau_c(\Omega_1 + \Omega_2) \] (2.4.14)

Therefore, the real and imaginary components of the desired specifications can be rewritten as:

\[
\begin{align*}
Y(\Omega_{1i}, \Omega_{2j}) &= R(\Omega_{1i}, \Omega_{2j}) \cos \left[ -\tau_c(\Omega_{1i} + \Omega_{2j}) \right] \\
Y'(\Omega_{1i}, \Omega_{2j}) &= R(\Omega_{1i}, \Omega_{2j}) \sin \left[ -\tau_c(\Omega_{1i} + \Omega_{2j}) \right]
\end{align*}
\] (2.4.15)

for \( i=1, 2, \ldots, L_1; \ j=1, 2, \ldots, L_2 \)

Now, with,

\[
Q_R(\Omega_{1i}, \Omega_{2j}) = 1 + \sum_{m=0}^{M_2} \sum_{n=0}^{N_2} b_{mn} \cos(m\Omega_{1i} + n\Omega_{2j}) \] (2.4.16)

\[
Q_I(\Omega_{1i}, \Omega_{2j}) = \sum_{m=0}^{M_2} \sum_{n=0}^{N_2} b_{mn} \sin(m\Omega_{1i} + n\Omega_{2j}) \] (2.4.17)

\[
P_R(\Omega_{1i}, \Omega_{2j}) = \sum_{m=0}^{M_1} \sum_{n=0}^{N_1} a_{mn} \cos(m\Omega_{1i} + n\Omega_{2j}) \] (2.4.18)

\[
P_I(\Omega_{1i}, \Omega_{2j}) = \sum_{m=0}^{M_1} \sum_{n=0}^{N_1} a_{mn} \sin(m\Omega_{1i} + n\Omega_{2j}) \] (2.4.19)

The linear programming design problem for two dimensional filter becomes:

Maximize

\[ g = \xi \]
Subject to

\[ \sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} R(\Omega_{li}, \Omega_{2j}) \cos((m-\tau_c)\Omega_{li} + (n-\tau_c)\Omega_{2j}) \]

\[ m+n \neq 0 \]

\[ \sum_{m=0}^{M1} \sum_{n=0}^{N1} a_{mn} \cos(m\Omega_{li} + n\Omega_{2j}) + \xi \leq -R(\Omega_{li}, \Omega_{2j}) \cos(-\tau_c(\Omega_{li} + \Omega_{2j})) \]

\[ (2.4.20) \]

\[ \sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} R(\Omega_{li}, \Omega_{2j}) \sin((m+\tau_c)\Omega_{li} + (n+\tau_c)\Omega_{2j}) \]

\[ m+n \neq 0 \]

\[ \sum_{m=0}^{M1} \sum_{n=0}^{N1} a_{mn} \sin(m\Omega_{li} + n\Omega_{2j}) + \xi \leq -R(\Omega_{li}, \Omega_{2j}) \sin(-\tau_c(\Omega_{li} + \Omega_{2j})) \]

\[ (2.4.21) \]

\[ \sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} R(\Omega_{li}, \Omega_{2j}) \cos((m-\tau_c)\Omega_{li} + (n-\tau_c)\Omega_{2j}) \]

\[ m+n \neq 0 \]

\[ \sum_{m=0}^{M1} \sum_{n=0}^{N1} a_{mn} \cos(m\Omega_{li} + n\Omega_{2j}) + \xi \leq R(\Omega_{li}, \Omega_{2j}) \cos(-\tau_c(\Omega_{li} + \Omega_{2j})) \]

\[ (2.4.22) \]

\[ \sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} R(\Omega_{li}, \Omega_{2j}) \sin((m-\tau_c)\Omega_{li} + (n-\tau_c)\Omega_{2j}) \]

\[ m+n \neq 0 \]

\[ \sum_{m=0}^{M1} \sum_{n=0}^{N1} a_{mn} \sin(m\Omega_{li} + n\Omega_{2j}) + \xi \leq R(\Omega_{li}, \Omega_{2j}) \sin(-\tau_c(\Omega_{li} + \Omega_{2j})) \]

\[ (2.4.23) \]
and the stability constraint,

$$1 + \sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} \cos(m\Omega_{1i} + n\Omega_{2j}) \geq \Delta C$$  \hspace{1cm} (2.4.24)

where $\Delta C$ is a small positive quantity.

The design procedure, which is similar to the one dimensional case, can be described as follows:

1) Specify the desired magnitude characteristics:

$$R(\Omega_{1i}, \Omega_{2j}) ; i=1,2,\ldots,L1 ; j=1,2,\ldots,L2$$

2) Choose an order for the filter, say $N$; and choose a range for $\tau_C$.

3) Solve the linear programming problem for each $\tau_C$ from its upper limit (in descending order) until a maximum $\xi$ is obtained. Denote the corresponding $\tau_C$ as $\tau_{C_{\text{max}}}$

4) Choose coefficients of the filter for which maximum $\xi$ was obtained and compute frequency response and error measures.

The range of $\tau_C$ in Step 2 is chosen based on $N$. The larger the range for $\tau_C$, the higher is the computational time for the design of the filter. A suitable range has been found to be $(N-2) \leq \tau_C \leq (N+1)$.

The error measures in Step 4 are: a) squared sum of magnitude errors $E_1$, and is given by:

$$E_1 = \sum_{i=1}^{L1} \sum_{j=2}^{L2} \left| R(\Omega_{1i}, \Omega_{2j}) - H(\Omega_{1i}, \Omega_{2j}) \right|^2$$  \hspace{1cm} (2.4.25)

+ If $(N-2)$ is less than or equal to zero, the lower limit is to be set equal to 1.
b) the squared error sum in spatial delay $\tau_1(\Omega_1, \Omega_2)$ in the pass band of the filter, $E_2$, and is given by:

$$(\text{Pass Band}) \quad E_2 = \sum \sum \{\tau_{c_{\text{max}}} - \tau_1(\Omega_{1i}, \Omega_{2j})\}^2 \quad (2.4.26)$$

and c) the squared error sum in spatial delay $\tau_2(\Omega_1, \Omega_2)$ in the pass band of the filter $E_3$, which is given by:

$$(\text{Pass Band}) \quad E_3 = \sum \sum \{\tau_{c_{\text{max}}} - \tau_2(\Omega_{1i}, \Omega_{2j})\}^2 \quad (2.4.27)$$

3.4.d Computational Considerations

The computational simplifications, as well as the storage reductions in the two dimensional case are similar to the one dimensional case. The linear programming problem of two dimensional filter design, given in Section 2.4.c, can be written in a matrix form as:

$$\begin{bmatrix} b_{01}, b_{02}, \ldots, b_{0N_2}, b_{10}, b_{12}, \ldots, b_{1N_2}, \ldots, b_{M2N2}, a_{00}, a_{01}, \ldots, a_{M1N1}, \xi \end{bmatrix} \begin{bmatrix} D \end{bmatrix} \leq \begin{bmatrix} c \end{bmatrix} \quad (2.4.28)$$

such that,

$$g = \begin{bmatrix} b_{01}, \ldots, b_{M2N2}, a_{00}, \ldots, a_{M1N1}, \xi \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \quad (2.4.29)$$
is maximized. The matrix $D$ consists of several submatrices as given below:

$$
D = \begin{bmatrix}
D_1 & D_2 & D_5 & D_6 & D_9 \\
D_3 & D_4 & D_7 & D_8 & D_{10} \\
D_{11} & D_{12} & D_{13}
\end{bmatrix} \quad (2.4.30)
$$

Matrices $D_1$ and $D_2$ are also made up of submatrices:

$$
D_1 = \begin{bmatrix}
F_{1,1} & F_{1,2} & \cdots & F_{1,L_2} \\
F_{2,1} & & & \\
& & & \\
& & & \\
F_{(M_2+1),1} & \cdots & \cdots & F_{(M_2+1),L_2}
\end{bmatrix} \quad (2.4.31)
$$

$$
D_2 = \begin{bmatrix}
G_{1,1} & G_{1,2} & \cdots & G_{1,L_2} \\
G_{2,1} & & & \\
& & & \\
& & & \\
G_{(M_2+1),1} & \cdots & \cdots & G_{(M_2+1),L_2}
\end{bmatrix} \quad (2.4.32)
$$

The elements in the submatrices of $D_1$ are given by:

$$
F_{i,j_{kl}} = Y(\Omega_{1j}, \Omega_{2k}) \{ \cos (i-1)\Omega_{1j} + m\Omega_{2k} \}
$$

$$
+ Y'(\Omega_{1j}, \Omega_{2k}) \{ \sin (i-1)\Omega_{1j} + m\Omega_{2k} \} \quad (2.4.33)
$$
where,
\[ i = 1, 2, \ldots, (M+2) \]
\[ m = k \quad \text{and} \quad k = 1, 2, \ldots, N+2 \quad \text{for} \quad i = 1 \]
and
\[ m = (k-1) \quad \text{and} \quad k = 1, 2, \ldots, (N+2) \quad \text{for} \quad i \neq 1 \]
and
\[ j = 1, 2, \ldots, L+2; \quad \ell = 1, 2, \ldots, L+1 \]

The elements of the submatrices of \( D_2 \) are given by:
\[
G_{i,j,k,\ell} = Y'(\Omega_{1j},\Omega_{2\ell}) \{ \cos [(i-1)\Omega_{1j} + m\Omega_{2\ell}] \} \\
- Y'(\Omega_{1j},\Omega_{2\ell}) \{ \sin [(i-1)\Omega_{1j} + m\Omega_{2\ell}] \} \quad (2.4.35)
\]

where \( i, j, k, \ell, m \) are as given in (2.4.34).

The submatrix \( D_9 \) is also made up of submatrices as follows:

\[
D_9 = \begin{bmatrix}
H_{1,1} & H_{1,2} & \cdots & H_{1,L+2} \\
H_{2,1} & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots \\
H_{(M+2),1} & \cdots & \cdots & H_{(M+2),L+2}
\end{bmatrix} \quad (2.4.36)
\]

and the elements of the submatrices \( \{ H_{i,j} \} \), which corresponds to the stability constraint given by (2.4.24), are:

\[
H_{i,j,k,\ell} = -\cos [(i-1)\Omega_{1j} + m\Omega_{2\ell}] \quad (2.4.37)
\]

where \( i, j, k, \ell, m \) are as given in (2.4.34).
From the above, it can be seen that each of the submatrices, \( D_1 \), \( D_2 \) and \( D_9 \), are of size \((M2+1)(N2+1)-1 \times (L1\cdot L2)\). Further, the first row of submatrices of \( D_1 \), \( D_2 \) and \( D_9 \) are of size \( N2 \times L1 \), where as the rest of the submatrices of \( D_1 \), \( D_2 \) and \( D_9 \) are of size \((N2+1) \times L1\). The submatrices \( D_3 \) and \( D_4 \) of \( D \) are further made up of submatrices as follows:

\[
D_3 = \begin{bmatrix}
I_{1,1} & I_{1,2} & \ldots & I_{1,L2} \\
I_{2,1} & & & \\
& \ddots & & \\
& & \ddots & I_{(M1+1),L2}
\end{bmatrix} \quad (2.4.38)
\]

\[
D_4 = \begin{bmatrix}
J_{1,1} & J_{1,2} & \ldots & J_{1,L2} \\
J_{2,1} & & & \\
& \ddots & & \\
& & \ddots & J_{(M1+1),L2}
\end{bmatrix} \quad (2.4.39)
\]

where elements of each of submatrices \( I_{i,j} \)'s are:

\[
I_{i,j}_{k\ell} = -\cos\left[ (i-1)\Omega_{1j} + (k-1)\Omega_{2\ell} \right] \quad (2.4.40)
\]

and the elements of \( J_{i,j} \)'s are:

\[
J_{i,j}_{k\ell} = \sin\left[ (i-1)\Omega_{1j} + (k-1)\Omega_{2\ell} \right] \quad (2.4.41)
\]

and \( k=1,2,\ldots,(N1+1) \); \( \ell=1,2,\ldots,L1 \)
The submatrix \( D_{11} \) is as follows:

\[
D_{11} = \begin{bmatrix}
1 & 1 & \ldots & 1
\end{bmatrix};
\]

(2.4.42)

and is of size \( 1 \times (2 \cdot L1 \cdot L2) \). The remaining submatrices of \( D \) are as follows:

\[
D_5 = -D_1; \quad D_6 = -D_2; \quad D_7 = -D_3; \quad D_8 = -D_4
\]

\[
D_{10} = 0; \quad D_{12} = D_{11}; \quad D_{13} = 0
\]

(2.4.43)

The row matrix, \( C \), on the right hand side of (2.4.27) can be split into four submatrices, as shown below:

\[
C = \begin{bmatrix}
C_1 & C_2 & C_3 & C_4 & C_5
\end{bmatrix}
\]

(2.4.44)

\( C_1 \) and \( C_2 \) are further made up of submatrices as follows:

\[
C_1 = \begin{bmatrix}
S_1, S_2, \ldots, S_{L2}
\end{bmatrix}; \quad C_2 = \begin{bmatrix}
T_1, T_2, \ldots, T_{L2}
\end{bmatrix}
\]

(2.4.45)

The elements of the submatrices \( S_i \)'s and \( T_i \)'s are:

\[
S_{ij} = -Y(\Omega_{1i}, \Omega_{2j}); \quad T_{ij} = -Y'(\Omega_{1i}, \Omega_{2j})
\]

(2.4.46)

\( j = 1, 2, \ldots, L1 \)

The elements of \( C_5 \) are as follows:

\[
C_5 = \begin{bmatrix}
(1-\Delta C), \ldots, (1-\Delta C)
\end{bmatrix}
\]

(2.4.47)

and it is of size \( 1 \times (L1 \cdot L2) \). The remaining submatrices of \( C \) are:

\[
C_3 = -C_1; \quad C_4 = -C_2
\]

(2.4.48)
It is clear from the foregoing explanation that there is redundancy in the coefficient matrix, and in matrix C. Because of this redundancy, the storage requirement can be almost reduced by half. Thus, instead of storing entire D and C matrices, it is required to store only submatrices \( D_1, D_2, D_3, D_4, D_{11}, D_9, C_1, C_2 \) and \( C_5 \). As in the one dimensional case, the solution is obtained by solving the dual of the linear programming problem; this involves a comparatively smaller number of variables compared to the primal problem.

Finally, a procedure similar to the one dimensional case, is adopted in order to choose the number of sample frequencies over the right half plane of the frequency domain. Thus, given the denominator of the transfer function \( H(z_1, z_2) \), i.e.,

\[
Q(z_1, z_2) = \sum_{m=0}^{M_2} \sum_{n=0}^{N_2} b_{mn} z_1^m z_2^n.
\]

the frequency domain specifications are specified over \( L_1 \) and \( L_2 \) number of equally spaced points in the \( \Omega_1 \) and \( \Omega_2 \) frequency axes respectively, such that \( L_1 \geq 8 \times M_2 \) and \( L_2 \geq 4 \times N_2 \). The stability constraint is also specified over the same (or larger) number of points in order to reasonably ensure stability of the designed filter.

2.4.ε Design Examples

In the design procedure, the specifications in the frequency domain are specified over equally spaced frequency points in the right half of \( \Omega_1 - \Omega_2 \) plane. The value of \( \Delta C \) in the constraint equation of (2.4.24) was set equal to \( 1 \times 10^{-3} \).
Example 1. Low Pass Filter Design

The specifications are as follows:

\[ R(\rho) = 1.0 \quad \text{for} \quad 0 \leq \rho \leq 0.2\pi \]

\[ R(\rho) = e^{-k(\rho - 0.2\pi)} \quad \text{for otherwise} \]

where,

\[ \rho = \sqrt{\Omega_1^2 + \Omega_2^2} \]

The transition and stop band characteristics are gaussian and the value of \( k \) is set at 5.0. The order of the filter was chosen such that \( M1 = M2 = N1 = N2 = 3 \). The initial value of \( \tau_c \) was set equal to 3 for which the design had maximum value of \( \xi \). The approximation was carried out over a 32 x 17 grid of equally spaced points in the frequency domain. The error measures for this design were as follows:

\[ E_1 = 0.9192 \; ; \; E_2 = 0.92 \; ; \; E_3 = 0.93 \; ; \; |\xi| = 3.12 \times 10^{-2} \]

The desired magnitude and the designed magnitude and group delays \( \tau_1(\Omega_1, \Omega_2), \tau_2(\Omega_1, \Omega_2) \) are shown in Figure 2.16. Also Figure 2.17 shows the first 16 x 16 pixels of the impulse response and the coefficients of the designed filter.

Example 2. Band Pass Filter

The specifications for the band pass filter are as follows:

\[ R(\rho) = 1.0 \quad \text{for} \quad 0.3\pi \leq \rho \leq 0.4\pi \]

\[ R(\rho) = e^{-k(0.3 - \rho)^2} \quad \text{for} \quad 0 \leq \rho \leq 0.3\pi \]

\[ R(\rho) = e^{-k(\rho - 0.4\pi)^2} \quad \text{for otherwise} \quad (\text{value of } k = 5.0) \]
Figure 2.16  a) Desired Magnitude Specifications
b) Designed Magnitude Response
c) & d) Designed Group Delays $\tau_1$ And $\tau_2$
Figure 2.17 (a) First 16 x 16 Pixels of the Impulse Response
(b) Coefficients of the Designed Filter

Each square area of constant intensity represents one pixel.
Figure 2.18  

a) Desired Magnitude Specifications  
b) Designed Magnitude Response  
c) & d) Designed Group Delays $\tau_1$ and $\tau_2$
Figure 2.19  
a) First 16 x 16 Pixels of the Impulse Response*  
b) Coefficients of the Designed Filter.  

*Each square area of constant intensity represents one pixel.
The order of the filter is such that $M_1 = M_2 = N_1 = N_2 = 3$. The initial value of $\tau_c$ was set at 3 and a lower limit of 1. A good design was obtained for $\tau_c = 3$ for which the value of $\xi$ was a maximum. The error measures for this design are as follows:

$$E_1 = 2.27; \quad E_2 = 0.81; \quad E_3 = 0.67; \quad |\xi| = 9.29 \times 10^{-2}$$

Figure 3.18 shows the desired magnitude and the designed magnitude, group delay responses. Figure 2.19 shows the first $16 \times 16$ pixels of the impulse response and the coefficients of the designed filter.

**Example 3: High Pass Filter Design**

The magnitude specifications for this design are as follows:

$$R(\rho) = e^{-k(0.5\pi - \rho)^2} \quad \text{for} \ 0 < \rho < 0.5\pi$$

$$R(\rho) = 1.0 \quad \text{for otherwise (value of} \ k = 5.0)$$

The range of $\tau_c$ chosen was such that $1 < \tau_c < 3$. The order of the filter is such that $M_1 = N_1 = M_2 = N_2 = 3$. The design corresponding to $\tau_c = 2$ resulted in a maximum in the value of $\xi$. The error measures are as follows:

$$E_1 = 0.447; \quad E_2 = 0.613; \quad E_3 = 0.57; \quad |\xi| = 6.19 \times 10^{-2}$$

Figure 2.20 shows the desired magnitude and designed magnitude, group delay characteristics. Figure 2.21 shows the first $8 \times 8$ pixels of the impulse response and the coefficients of the designed filter.
Figure 2.20  

a) Desired Magnitude Specifications.  
b) Designed Magnitude Response  
c) & d) Designed Group Delays $\tau_1$ and $\tau_2$.  

Figure 2.21  (a) First 16 x 16 Pixels of Impulse Response
(b) Coefficients of the Designed Filter.

*Each square area of constant intensity represents one pixel.*
2.5 **Summary**

A linear programming method has been presented for the design of one and two dimensional recursive digital filters to approximate magnitude and phase characteristics. The variables of the linear program are the coefficients of the desired filter and the linear programming problem is set up such that they minimize the real and imaginary components of the complex weighted error.

The types of stability constraints, which are linear in form, are also indicated and are incorporated in the method to design stable filters. The constraints used are sufficient conditions for stability and proofs of their sufficiency are also presented.

The filters are designed in the direct form and examples of design are also presented.
CHAPTER III

RECURSIVE DIGITAL FILTER APPLICATIONS IN IMAGE PROCESSING

3.1 Introduction

Although recursive digital filters are computationally advantageous when compared to most convolutional methods of filtering, (specifically, convolution via FFT), little has been reported regarding the application of these filters in image processing. Lack of application of these filters in the past was primarily due to design and stability problems associated with these filters. In recent years, however, many of the stability and design problems have been overcome, specifically in the case of quarter plane filters and at the present time there are a large number of design methods available, including the one presented in Chapter II, and therefore this chapter considers in detail the applications of quarter plane recursive digital filters to image processing problems in the areas of enhancement and restoration.

A detailed analysis of the usefulness of various two dimensional recursive filter design techniques, indicated in Section 1.2, with regard to their ability to design recursive filters for image processing applications, will not be attempted here, since the task is beyond the scope of this thesis. However, a brief analysis is provided in the following paragraph.

As discussed in Section 1.2, in situations where simultaneous linear phase and arbitrary magnitude approximation are required, the linear programming technique, presented in Chapter II, is more advantageous to use than the group delay
(phase) equalization technique of [3], which uses non-linear optimization for approximation. In situations where approximation of arbitrary specifications to the real part of a transfer function or to the magnitude squared transfer function, the non-linear optimization design procedures are more useful than the design techniques of [9] and the technique of Chapter II, which use linear programming. Considering magnitude squared approximations, the linear programming technique of Chapter II simplifies to the design technique of Dudgeon [8]. As discussed in [32], there are severe problems in implementation of the magnitude squared transfer function designed using the method of [8]. In the case of an approximation to the real part of the transfer function, $\text{Re} \, H(Z_1, Z_2)$, the coefficients of the numerator of $\text{Re} \, H(Z_1, Z_2)$ are in terms of the product of numerator and denominator coefficients of the original transfer function $H(Z_1, Z_2)$. Such an approximation is therefore, not in a linear form, and cannot be performed using linear programming techniques.

All of the non-linear optimization design techniques discussed in Section 1.2, design filters in cascades of lower order sections and therefore the filters obtained due to these design techniques belong to a subclass of all possible stable realizations.† Among these techniques, the most recent and the most improved, is the design procedure

---

† $\text{Re} \, H(Z_1, Z_2)$ refers to real part of filter transfer function, $H(Z_1, Z_2)$.

† This is due to the fact that the fundamental theorem of algebraic factorization does not exist in two dimensions.
of [4]. Compared to the remainder of the similar techniques, the technique of [4] designs filters with the general class of low order stable filter sections.

From the brief analysis above, it can be seen that suitable techniques for designing filters for image processing applications are the linear programming technique of Chapter II and the non-linear optimization design procedure of [4]. In the following sections on applications of recursive digital filters to image processing, the linear programming design technique is used in designing filters to simultaneous specification of magnitude and linear phase and the non-linear optimization procedure of [4], which is described in Appendix F, is made use of in approximating to the real part of a transfer function or to the magnitude squared transfer function.

3.2 Application of Recursive Digital Filters to Image Enhancement

In image processing, the familiar enhancement problems are: high frequency emphasis or crispening and edge enhancement. The filters required for these applications have magnitude characteristics that are generally high pass in nature and linear phase characteristics over most of the frequency domain. As indicated earlier, the suitable design technique is the linear programming design procedure, with which the desired specifications can be approximated simultaneously. In the following subsections, the usefulness of this design technique is illustrated by filters designed to prescribed specifications and applied to image processing
problems.

3.2.a **High Frequency Emphasis or Crispening**

Oftentimes it is required to make an image appear sharper. One way of achieving this objective is to pass the image through a high pass filter and thus emphasize the high frequency components of the image. Consider the image of Figure 3.1(a), which is sampled using the flying spot scanner. Referring to Appendix G (second section), the intensity of each pixel of the sampled image is the convolution of the original image with the CRT (cathode ray tube) beam spot of the flying spot scanner. The sampled image can also be thought of as the output of a filter, with the original image as the input and the transfer function of the filter equal to the fourier transform of CRT spot of light incident on the film. The filter transfer function is low pass in characteristics, as the intensity profile of CRT spot is gaussian in nature. The bandwidth of this filter is directly related to the aperture of the CRT spot. If the CRT beam is out of focus, then the aperture of the spot incident on the film is larger than normal and therefore the bandwidth of the low pass filter becomes very small. This causes severe attenuation of mid and higher frequency components of the image and results in a blurred image as shown in Figure 3.1(a).

Blurred images similar to the one shown in Figure 3.1(a) can be enhanced by passing the image through a high pass filter, which compensates for the attenuation of higher frequency components. For the image of Figure 3.1(a), a high pass
Figure 3.1 a) Original Image  
b) Enhanced Image.
Figure 3.2  
a) Designed High Pass Filter Magnitude Response  
b) & c) Designed Group Delay Responses $\tau_1$ and $\tau_2$  
d) First 8 x 8 Pixels of the Impulse Response$^+$  

$^+$Each square area of constant intensity represents one picture element.
\[
\begin{align*}
a_{00} &= -1.5575530 \times 10^{-1} \\
a_{01} &= 0.46834400 \times 10^{-1} \\
a_{02} &= -3.9941100 \times 10^{-2} \\
a_{10} &= 0.49515360 \times 10^{-1} \\
a_{11} &= -2.1031310 \times 10^{-1} \\
a_{12} &= -2.2371150 \\
a_{20} &= 0.41128100 \times 10^{-2} \\
a_{21} &= -2.3362350 \\
a_{22} &= 0.70751300
\end{align*}
\]
\[
\begin{align*}
b_{00} &= 1.0000000 \\
b_{01} &= 0.22357600 \\
b_{02} &= 0.71496190 \times 10^{-1} \\
b_{10} &= 0.22108698 \\
b_{11} &= 0.15445120 \\
b_{12} &= 0.10571910 \\
b_{20} &= 0.91730540 \times 10^{-1} \\
b_{21} &= 0.10200290 \\
b_{22} &= 0.15625000 \times 10^{-1}
\end{align*}
\]

Figure 3.2(e) Coefficients of the Designed Filter.
filter was designed with arbitrarily chosen specifications. The gain of the filter was specified to be equal to 1 beyond the radial frequency of 0.7\(\pi\), and 0.1 below the radial frequency of 0.3\(\pi\). The transition band in between 0.3\(\pi\) and 0.7\(\pi\) was gaussian. The magnitude, group delay and the impulse response of the designed filter are shown in Figure 3.2. The coefficients of the designed filter is shown in Figure 3.3. The high pass filtered image of 3.1(a) is shown in Figure 3.1(b). From this figure, it is clear that the image is much sharper compared to the original and the details which are blurred in the original are much clearer in the filtered version.

3.2.b  **Edge Enhancement**

One of the approaches to picture segmentation is based on the detection of discontinuities; i.e., lines along which there is an abrupt change in gray level, indicating the end of one region and the beginning of another. Such discontinuities are called edges. Prior to the detection of these edges, it is required that they be enhanced. The filter that is often used for edge enhancement is the Laplacian [36,37]. In continuous space, it is defined as:

\[
\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{3.2.1}
\]

where \(f(x,y)\) is the image intensity which is a function of spatial coordinates \(x\) and \(y\). The fourier transform of (3.2.1) is given by:

\[
L(w_1,w_2) = -(w_1^2 + w_2^2) \cdot F(w_1,w_2) \tag{3.2.2}
\]

where \(F(w_1,w_2)\) is the fourier transform of the image \(f(x,y)\) and \(w_1\) and \(w_2\) are continuous frequency variables. Thus, the
laplacian operator can be represented in terms of a linear shift invariant system possessing a transfer function:

\[ H(w_1, w_2) = -(w_1^2 + w_2^2) \]  \hspace{1cm} (3.2.3)

Assuming \( w_s \) to be the sampling frequency at which the image is sampled along both the spatial directions, the frequency response of the laplacian filter can be written as:

\[ H(\Omega_1, \Omega_2) = -(\Omega_1^2 + \Omega_2^2) \]  \hspace{1cm} (3.2.4)

where \( \Omega_1 = w_1 \pi / (w_s/2) \) and \( \Omega_2 = w_2 \pi / (w_s/2) \) are the normalized frequency variables. The negative sign in (3.2.4) can be neglected, since it simply involves multiplying the input or the output by -1. Therefore the edge enhancement filter can be written as:

\[ H(\Omega_1, \Omega_2) = (\Omega_1^2 + \Omega_2^2) \]  \hspace{1cm} (3.2.5)

The phase characteristics associated with the transfer function \( H(\Omega_1, \Omega_2) \) of (3.2.5) are zero or linear phase. A filter was designed to approximate the magnitude characteristics of (3.2.5) with linear phase using the linear programming technique. The desired magnitude characteristics are shown in Figure 3.3(a). The designed filter magnitude, group delay and impulse response characteristics and the coefficients are shown in Figure 3.3. This filter was then applied to enhance the edges of the image shown in Figure 3.4(a). The result of enhancing the edges is shown in Figure 3.4(b). It is obvious that there is indiscriminate enhancement of all edges of the image; the desired result.

For the purpose of comparison, a recursive filter
Figure 3.3

(a) Desired Magnitude Specifications
(b) Designed Magnitude Response
(c) & (d) Designed Group Delays, $\tau_1$ and $\tau_2$
Figure 3.3  (e)  **First 8 x 8 Pixels of the Designed Filter Impulse Response**

(f)  **Coefficients of the Designed Filter**

+ Each square area of constant intensity represents one pixel.
Figure 3.4  
(a) Original Image  
(b) Edge Enhanced Image with Linear Phase Filter  
(c) Edge Enhanced Image with Non-Linear Phase Filter.
Figure 3.5  

(a) Magnitude Response  
(b) & (c) Group Delays $\tau_1$ and $\tau_2$  
(d) Impulse Response

*Each square area of constant intensity represents one pixel.*
\begin{align*}
a_{00} &= 1.000000 \\
a_{01} &= 0.33264320 \\
a_{02} &= -0.14728560 \\
a_{10} &= 0.33264320 \\
a_{11} &= -0.44744180E-01 \\
a_{12} &= -0.11984560 \\
a_{20} &= -0.14728560 \\
a_{21} &= -0.11984560 \\
a_{22} &= -0.63393340E-02 \\
b_{00} &= 1.000000 \\
b_{01} &= 0.93398490 \\
b_{02} &= 0.17348960 \\
b_{10} &= 0.93398490 \\
b_{11} &= 0.90195800 \\
b_{12} &= 0.18074860 \\
b_{20} &= 0.17348960 \\
b_{21} &= 0.18074860 \\
b_{22} &= 0.42728630E-01
\end{align*}

Figure 3.5(e) Coefficients of the Filter.
was designed in which the magnitude alone was given consideration. The characteristics of the designed filter and the coefficients are shown in Figures 3.5. Using this filter, the image of Figure 3.4(a) was filtered to obtain an enhanced image of 3.4(c). The consequences of neglecting the phase characteristics in the design are obvious in the enhanced image.

3.3 Application of Recursive Digital Filters in Image Restoration

The problem of image restoration of degraded images deals with the removal of the sources of degradation. The type of degradation can be spatially varying or spatially invariant. In this section, we will consider using recursive filter implementations for image restoration, where image degradations are of the latter type. The most common types of degradations are:

a) Motion Blur
b) Focus Blur
c) Atmospheric Turbulance Blur

With the assumption of spatial invariance, the degrading system, corresponding to the three types of blur can be modeled as shown in Figure B2 of Appendix B. In this figure, \( g(x,y) \) is the blurred image and \( h(x,y) \) is the point spread function corresponding to the type of blur in question.

In order to restore blurred images, the first task is to identify the blur and the second to generate the corresponding restoration or inverse filter transfer function. The details of the blur point spread function identi-
fication will not be given here, since a complete description of identification of the three different types of blurs is found in the work of Cannon [23]. The generation of restoration filter response is presented in the next section. In the work reported by Cannon and others, the inverse filtering is carried out by using non-recursive methods, specifically, by the use of fast fourier transform (FFT). Here, we consider performing inverse filtering using recursive digital filters whose response has been approximated to the desired restoration filter characteristics. For the purpose of comparison, the inverse filtering was also carried out via convolution using the FFT. The restoration filter (i.e., inverse filter) impulse response was windowed by a two dimensional Hamming window, thus restricting the convolution kernal to the chosen size of the window. For the extent of blurs considered in all the examples in this thesis, a kernal size of 32 x 32 was found to be appropriate. Larger kernal sizes did not appear to improve the restorations significantly. However, if the extent of blur is to be increased to larger values than what is considered here, the kernal sizes would have to be increased further. The convolution via the FFT was implemented using the overlap and save method [1], with an FFT array size equal to 64 x 64. Since the input to the FFT is complex, the convolution can be carried out for two 32 x 32 sections of image data at a time.

The original images used in the examples were sampled from photographs. The blurring of the images was performed
by the computer, from known blurring functions and computer
generated noise (wide band gaussian) was added to the blurred
images, with a specified signal to noise ratio (SNR). The
two dimensional recursive digital filters used in the inverse
filtering of the blurred images were designed using either
the cascaded design procedure (described in Appendix F) or
the linear programming technique described in Chapter II,
depending on the type of approximation at hand.

3.3a Restoration Filter Transfer Functions

Consider, the block diagram of Figure B2 in Appendix B.
Let \( b(x,y) \) be the intensity of the blurred image. Therefore,
from B3:

\[
b(x,y) = h(x,y) \odot i(x,y) + n(x,y)
\]  \hspace{1cm} (3.3.1)

where \( h(x,y) \) is the impulse response of the blurring system.
The fourier transform of the blurred image can therefore be
written as:

\[
B(w_1, w_2) = H(w_1, w_2) \cdot I(w_1, w_2) + N(w_1, w_2)
\]  \hspace{1cm} (3.3.2)

where \( w_1 \) and \( w_2 \) are the frequency variables in the two
dimensional frequency domain. \( B, H, I \) and \( N \) are the fourier
transforms of \( b, h, i \) and \( n \) respectively. If \( w_s \) is the
frequency at which the image is sampled along both the
spatial directions, then (3.3.2) can be written as:

\[
B(\Omega_1, \Omega_2) = H(\Omega_1, \Omega_2) \cdot I(\Omega_1, \Omega_2) + N(\Omega_1, \Omega_2)
\]  \hspace{1cm} (3.3.3)

where \( \Omega_1 = \frac{w_1 \pi}{w_s/2} \) and \( \Omega_2 = \frac{w_2 \pi}{w_s/2} \) are the normalized
frequency variables. From (3.3.3), the power spectrum of \( b(x,y) \) can be written as:

\[
P_b(\Omega_1, \Omega_2) = |H(\Omega_1, \Omega_2)|^2 \cdot P_I(\Omega_1, \Omega_2) + P_N(\Omega_1, \Omega_2)
\]

(3.3.4)

where \( P_B \), \( P_I \) and \( P_N \) are the power spectra of \( b \), \( i \) and \( n \) respectively.

Consider now, a blurred image that is free of noise. For this case, \( N(\Omega_1, \Omega_2) \) in (3.3.3) is equal to zero. Therefore, the spectrum of the original image can be obtained in a straightforward manner, as:

\[
I(\Omega_1, \Omega_2) = B(\Omega_1, \Omega_2)/H(\Omega_1, \Omega_2)
\]

(3.3.5)

provided \( H(\Omega_1, \Omega_2) \) is known and invertible. Thus the simple restoration, or inverse filter, would be:

\[
H_R(\Omega_1, \Omega_2) = \frac{1}{H(\Omega_1, \Omega_2)}
\]

(3.3.6)

With the knowledge of the type of blur, the simple restoration filter \( H_R(\Omega_1, \Omega_2) \) can be computed and in the ideal situation (when the blurred image is noise free), the original image can be obtained from the blurred image. However, in the presence of noise, the simple restoration filter would yield an inferior restoration as it tends to amplify the noise which dominates the higher frequency components of the image. Thus, one has to consider restoration filters which take into account the presence of noise in the image.

In the literature, the restoration filters, for
deblurring blurred noisy image, have been derived based on two criteria. These are the minimum mean squared error criteria (MMSE) and the power spectrum equalization criteria (PSEC). The former criterion can be written as:

Minimize

\[ E \{ i(x,y) - \hat{i}(x,y) \}^2 \quad \text{over all } x \text{ and } y \quad (3.3.7) \]

where \( i(x,y) \) is the original image and \( \hat{i}(x,y) \) is the restored image estimate. A restoration filter based on this criterion is given by [38,39].

\[
H_{R_1}(\Omega_1, \Omega_2) = \frac{H^*(\Omega_1, \Omega_2) \cdot P_I(\Omega_1, \Omega_2)}{|H(\Omega_1, \Omega_2)|^2 \cdot P_I(\Omega_1, \Omega_2) + P_N(\Omega_1, \Omega_2)} \quad (3.3.8)
\]

where \( P_I \) and \( P_N \) are the power spectra of the original image and noise. The second criterion suggested and used by Cannon and Stockham [23,39], can be stated as:

\[
P_B(\Omega_1, \Omega_2) \cdot |H_{R_2}(\Omega_1, \Omega_2)|^2 = P_I(\Omega_1, \Omega_2) \quad (3.3.9)
\]

where \( H_{R_2} \) is the desired restoration filter and \( P_B \) and \( P_I \) are the power spectra of the blurred and original images respectively. The restoration filter can then be written as:

\[
H_{R_2}(\Omega_1, \Omega_2) = \sqrt{\frac{P_I(\Omega_1, \Omega_2)}{P_B(\Omega_1, \Omega_2)}} \cdot e^{-j\phi}
\]

\[
\phi = \tan^{-1}\left(\frac{\text{Imag } H(\Omega_1, \Omega_2)}{\text{Real } H(\Omega_1, \Omega_2)}\right) \quad (3.3.10)
\]

\( \hat{E}\{ \} \) refers to the expected value of the quantity inside the parenthesis.

* denotes conjugation.
and as indicated earlier, \( P_B \) is given by:

\[
P_B(\Omega_1, \Omega_2) = P_I(\Omega_1, \Omega_2) \cdot |H(\Omega_1, \Omega_2)|^2 + P_N(\Omega_1, \Omega_2)
\]

In (3.3.8) and (3.3.10), the function \( H(\Omega_1, \Omega_2) \) represents the blurring transfer function. Also, it should be noted that in (3.3.8) and (3.3.10) the phase response of \( H_{R_1}(\Omega_1, \Omega_2) \) and \( H_{R_2}(\Omega_1, \Omega_2) \) are identical to the phase response of \( H(\Omega_1, \Omega_2) \).

In the examples considered in this thesis, since the image blurrings are performed by the computer, the quantities on the right hand side of (3.3.8) and (3.3.10) are readily available to compute the restoration filters \( H_{R_1} \) and \( H_{R_2} \). In actual situation, where only the blurred image is available, the blurring transfer function \( H(\Omega_1, \Omega_2) \) has to be determined from the blurred image itself and the details of this procedure can be found in [23]. Also the power spectrum \( P_I(\Omega_1, \Omega_2) \) of the original image is required in computing \( H_{R_1} \) and \( H_{R_2} \), and is unknown, when only the blurred image is available. However, it has been suggested, and shown, by Cannon [23], that good restorations can be obtained where the power spectrum of an image, which is statistically similar to the original image, is substituted for \( P_I(\Omega_1, \Omega_2) \). It has also been shown by Cannon [23], that the restoration filter \( H_{R_2} \) (based on PSEC) provides restorations that are somewhat superior to that of \( H_{R_1} \) (based on MMSEC), for the case of motion and focus blurs. However, in the case of atmospheric turbulence blur, \( H_{R_1} \) has been shown to handle the situation more effectively compared to \( H_{R_2} \). Therefore,
in the application examples shown, $H_{R_2}$ has been used for motion and focus blur restorations and $H_{R_1}$ for atmospheric turbulence blur restorations.

3.3.b Uniform Motion Blur

Subjecting a point source of light to a camera which is operated while in uniform motion produces a streak in the resulting picture. Plotting the intensity of this image, as a function of spatial coordinates, results in a rectangular function in the direction of the blur; as shown in Figure 3.6. This function is the impulse response (point spread function) of the blurring system and is given by:

$$h(r, \phi) = \begin{cases} \dfrac{1}{d} & \text{for } r < \dfrac{d}{2} \text{ and } \phi = 0 \\ \end{cases}$$

$$r = \sqrt{x^2 + y^2}$$

(3.3.11)

= 0 otherwise

where $x$ and $y$ are spatial coordinates and $\theta$ is the angle with respect to the $x$ axis along which the camera motion has taken place. The fourier transform of this function has the form of a $\sin x / x$ function in the direction of blur and is constant in the direction perpendicular to it. The fourier transform is given by:

$$H_M(w_1, w_2) = \frac{\sin (wd/2)}{(wd/2)} \quad w = w_1 \cos \theta + w_2 \sin \theta$$

(3.3.12)

where $w_1$ and $w_2$ are spatial frequency variables. If we let $\Delta$ be the sampling interval, chosen for sampling the images along both the spatial directions, then the blur length $d$ can be expressed in terms of $\Delta$ as:
Figure 3.6  
(a) Motion Blur Spatial Response  
(b) Motion Blur Transfer Function  
(c) Amount of Phase Shift Introduced by (b)
\[ d = k\Delta \quad (3.3.13) \]

where \( k \) is any positive number. Let the sampling frequency in radians be \( w_s \), so that:

\[ w_s = \frac{2\pi}{\Delta} \quad (3.3.14) \]

Therefore (3.3.2) can be rewritten as:

\[ H_M(\Omega_1, \Omega_2) = \frac{\sin \frac{\Omega k/2}{\Omega k/2}}{\Omega_1} ; \quad \Omega = \Omega_1 \cos \theta + \Omega_2 \sin \theta \quad (3.3.15) \]

where \( \Omega_1 = \frac{w_1\pi}{(w_s/2)} \) and \( \Omega_2 = \frac{w_2\pi}{(w_s/2)} \) are the normalized frequency variables. The plot of \( H_M(\Omega_1, \Omega_2) \) along the direction of the blur is shown in Figure 3.6(b). In (3.3.13), \( k \) is referred to as the blur length in pixels. The motion blurring transfer function of (3.3.15) is purely real and it attenuates the higher frequency components of the image. Also, the alternate side lobes of the transfer function are negative and this introduces a phase shift of \( \pi \) radians to the frequencies that lie in those negative regions, as shown in Figure 3.6(c).

Figure 3.7(a) shows the computer generated motion blurring transfer function for a blur length of 4 pixels. The blur is along a direction which makes an angle \( \theta = 45^\circ \) with respect to the x axis. An original image, shown in Figure 3.7(b) is blurred using the above transfer function. The blurring is performed by convolution via the FFT, with kernel size of 32 x 32 pixels. Noise is then added to the blurred image such that the SNR is equal to 30 db. The resulting
Figure 3.7  
(a) Blurring Transfer Function. Blur Length=4 Pixels, Angle=45 degrees w.r.t. Horizontal.  
(b) Original Image  
(c) Image Blurred by (a), With Noise, SNR=30 db.
blurred, noisy image is shown in Figure 3.7(c). For this blurred image, a restoration filter response $H_{MR_2}(\Omega_1, \Omega_2)$ was computed based on the power spectrum equalization criteria. The response is purely real and is shown in Figure 3.8(a). An approximate form of the response $H_{MR_2}(\Omega_1, \Omega_2)$, of the restoration filter along the direction $\theta = 45^\circ$ is shown in Figure 3.8(b). In Figure 3.8(a), the intense dark regions and intense bright regions correspond to the most positive and most negative values in Figure 3.8(b). The restoration filter magnitude characteristics is simply the absolute value of $H_{MR_2}(\Omega_1, \Omega_2)$ and the phase characteristics is as shown in Figure 3.8(c), which is identical to .6(c).

From Figure 3.8(c), it is clear that the phase characteristics is discontinuous in nature; therefore, a two dimensional recursive filter implementation having causal impulse response cannot be used to realize the motion blur restoration filter. This would become clear if the spatial delay characteristics corresponding to the desired phase response are considered. From Figure 3.8(c), at points where the phase characteristics is discontinuous, the corresponding spatial delay would be an impulse and such a delay characteristics cannot be realized by a two dimensional recursive filter. A more successful approach, to realize the desired restoration filter response of Figure 3.8(a) by a recursive filter is to use the implementation of Figure 3.9(a) where $H_1(Z_1, Z_2)$ is a two dimensional recursive digital filter transfer function. The overall frequency response;
Figure 3.8  
(a) Restoration Filter Response (Origin is at the Centre of the Image)  
(b) Restoration Filter Response Along $\theta = 45^\circ$  
(c) Phase Shift Introduced by (b)
\( H(\Omega_1, \Omega_2) \) of this implementation is given by:

\[
H(\Omega_1, \Omega_2) = H_1(z_1, z_2) + H_1^*(z_1, z_2) \quad (3.3.16)
\]

\[
= 2 \text{ Real}(H_1(z_1, z_2)); \quad (3.3.17)
\]

\[
z_1 = e^{-j\Omega_1}, \quad z_2 = e^{-j\Omega_2}
\]

\( H(\Omega_1, \Omega_2) \) is purely real and can assume both positive and negative values; therefore, it is now possible to approximate the desired restoration filter response by the recursive filter implementation of Figure 3.9(a). In (3.3.16), the impulse response of \( H_1(z_1, z_2) \) is causal and the impulse response of \( H_1^*(z_1, z_2) \) is the same as that of \( H_1(z_1, z_2) \), but non-causal. Therefore, the overall impulse response of \( H(\Omega_1, \Omega_2) \) is non-causal.

The restored image, \( i \), is the sum of the outputs \( i_1 \) and \( i_2 \) of \( H_1(z_1, z_2) \) and \( H_1^*(z_1, z_2) \), respectively. The output array, \( i_1 \), is obtained by filtering the input array, \( b \), with \( H_1(z_1, z_2) \), starting from the upper left hand corner of the input array. As shown in Chapter I, the filtering operation is carried out via the difference equation relating to \( H_1(z_1, z_2) \). The input array, \( i_2 \), is also obtained by filtering the input \( b \) by \( H_1(z_1, z_2) \), but starting from the diagonally opposite corner of the input array. The \( i_2 \) thus obtained is

\[\text{** denotes conjugation}\]
Figure 3.9  (a) Recursive Filter Implementation for Motion Deblur
(b) Frequency Response of (a) (Origin is at the Centre).
equivalent to filtered output of \( H_1^v(Z_1, Z_2) \).

Figure 3.9(b) shows the approximation of \( H(\Omega_1, \Omega_2) \) to the desired restoration filter response of Figure 3.8(a). The impulse response corresponding to the desired restoration filter is shown in Figure 3.10(b). Figure 3.10(c) shows the coefficients of the two-dimensional recursive digital filter \( H_1(Z_1, Z_2) \). The filter was designed using the cascaded design procedure (the design approach is described in Appendix F). The linear programming method, however, is not suitable in this situation because of the type of approximation involved.

The result of using the recursive filter implementation to restore the motion blurred, noisy image is shown in Figure 3.11. Figures 3.11(a) and (b) are the original and blurred images shown earlier in Figures 3.7(b) and (c). Figure 3.11(d) shows the restoration from motion blur, where the inverse

\[ I_2(Z_1, Z_2) = B(Z_1, Z_2) \cdot H_1^v(Z_1, Z_2) = B^*(Z_1, Z_2) \cdot H_1(Z_1, Z_2)^* \]

Let \( I_A(Z_1, Z_2) = B^*(Z_1, Z_2) \) and let \( i_A(m,n) \) be the inverse Z transform of \( I_A(Z_1, Z_2) \). The inverse Z transform of \( B^*(Z_1, Z_2) \) is an array \( b_1(m,n) \), such that:

\[ b_1(m,n) = b(M-m, N-n) \quad m=0,1,\ldots, M \quad n=0,1,\ldots, N \]

From the above it can be seen that \( b_1 \) is obtained by rotating \( b \) through 180° about its minor diagonal. \( i_A \) is now the filtered output of \( H_1 \), with \( b_1 \) as the input. As before \( i_2 \) is obtained by rotating \( i_A \) through 180° about its minor diagonal.

\[ \dagger \text{The impulse responses are shifted from the origin by 16 pixels in both spatial directions.} \]
$\$ \text{OF 1ST ORD CASCADES} = 1$, $\$ \text{OF 2ND ORD CASCADES} = 1$

$A_0 = 0.264022905158155 \times 0.01$

CASCAD SECTION 1
\[ A(0.0) = \begin{bmatrix} 1.000000000000000000000000 \end{bmatrix} \times 0.01 \]
\[ A(1.0) = \begin{bmatrix} -1.000000000000000000000000 \end{bmatrix} \times 0.01 \]
\[ A(1.1) = \begin{bmatrix} -1.678596599355020000 \end{bmatrix} \times 0.01 \]

CASCAD SECTION 2
\[ C(2.0) = \begin{bmatrix} 1.000000000000000000000000 \end{bmatrix} \times 0.01 \]
\[ C(2.1) = \begin{bmatrix} 0.113481494867157000 \end{bmatrix} \times 0.01 \]
\[ C(2.2) = \begin{bmatrix} 0.307356858703432000 \end{bmatrix} \times 0.01 \]
\[ C(1.0) = \begin{bmatrix} -0.250315303775457000 \end{bmatrix} \times 0.01 \]
\[ C(1.1) = \begin{bmatrix} 0.143236633523622000 \end{bmatrix} \times 0.01 \]
\[ C(1.2) = \begin{bmatrix} 0.149356646646766500 \end{bmatrix} \times 0.01 \]
\[ C(2.0) = \begin{bmatrix} -0.283378331748269000 \end{bmatrix} \times 0.01 \]
\[ C(2.1) = \begin{bmatrix} 0.403632413971472000 \end{bmatrix} \times 0.01 \]
\[ C(2.2) = \begin{bmatrix} -0.271511318415500700 \end{bmatrix} \times 0.01 \]

(c)

Figure 3.10 (a) Impulse Response of Desired Restoration Filter
(b) Impulse Response of Recursive Filter Implementation
(c) Coefficients of the Recursive Filter $H_1(z_1, z_2)$

† Origin is at the upper left hand corner. Size = 64x64 pixels.
Figure 3.11  
(a) Original Image  
(b) Blurred Image with Noise; Blur Length=4, Angle=45°, SNR=30 db  
(c) Image Restored Using FFT  
(d) Image Restored Using Recursive Implementation of Figure 3.9(a).
filtering was carried out using the recursive filter implementation of Figure 3.9(a). The restoration that was carried out via convolution using the FFT is shown in Figure 3.11(c). In another example considered, the blur length is increased to 8 pixels and the angle of blur is set equal to 22.5° with respect to the horizontal. The corresponding blurring transfer function frequency response is shown in Figure 3.12(a). The original image and the motion blurred noisy image, where the blurring is performed using the transfer function of Figure 3.12(a), is shown in Figure 3.15(a) and (b) respectively. The restoration filter specifications, computed based on the power spectrum equalization criteria, is shown in Figure 3.12(b). Figure 3.12(c) shows the frequency response $H(\Omega_1, \Omega_2)$ of the recursive filter implementation of Figure 3.9(a) after the approximation. The impulse responses corresponding to Figures 3.12(b) and (c) are shown in Figures 3.13(a) and (b) respectively. The coefficients of the designed recursive filter are shown in Figure 3.14. The results of using the recursive filter implementation for deblurring the motion blurred image of Figure 3.15(b) is shown in Figure 3.15(d). For the purpose of comparison, the restoration carried out by convolution via the FFT is shown in Figure 3.15(c).

3.3.c Focus Blur

When a camera lens system is out of focus, then the image of a point source of light is not a point; instead, it is a circular disc of constant intensity. This is the

†the impulse responses are shifted from origin by 16 pixels in each spatial direction.
Figure 3.12  
(a) Motion Blurring Transfer Function; Blur Length=8 pixels, Angle=22.5° with respect to horizontal 
(b) PSEC Restoration Filter Corresponding to (a) and SNR=30 db. 
(c) Recursive Filter Approximation to (b).
Figure 3.13  (a) Impulse Response of Desired Restoration Filter
(b) Impulse Response of Recursive Filter Implementation

†Origin is at the upper left hand corner. Size=64x64 pixels.
Figure 3.14 Coefficients of the Recursive Filter $H_1(Z_1, Z_2)$.
Figure 3.15  
(a) Original Image  
(b) Motion Blurred Image with Noise; Blur Length=8 pixels; Angle=22.5°, with respect to horizontal; SNR=30 db  
(c) Restoration Using FFT  
(d) Restoration Using Recursive Filter Implementation of Figure 3.9(a).
point spread function of a focus blurring system. Although the true point spread function is actually related to the fourier transform of the aperture of the lens system, a cylindrical approximation is a good one and is also mathematically tractable. Thus, the focus blurring point spread function can be written as:

\[
\begin{align*}
    h(r) &= 0 \quad ; \quad r > R \\
    h(r) &= \frac{1}{(\pi R^2)} \quad ; \quad r \leq R \quad ; \quad r = \sqrt{x^2 + y^2}
\end{align*}
\] (3.3.18)

The plot of (3.3.18) is shown in Figure 3.16(a). The fourier transform of this point spread function is a bessel function of first order and it is of the form:

\[
H_F(w) = 2 \cdot \frac{J_1(Rw)}{(Rw)} \quad ; \quad w = \sqrt{\frac{w_1^2}{\Delta^2} + \frac{w_2^2}{\Delta^2}}
\] (3.3.19)

where \(w_1\) and \(w_2\) are the two dimensional frequency variables. As in Section 3.3.b, let \(\Delta\) and \(w_s\) be the sampling interval and the sampling frequency respectively. The radius \(R\) of the blur point spread function can then be written as:

\[
R = k\Delta
\] (3.3.20)

where \(k\) is referred to as the radius of blur in pixels. Therefore, (3.3.19) can be rewritten as:

\[
H_F(\Omega) = 2 \cdot \frac{J_1(k\Omega)}{(k\Omega)} \quad ; \quad \Omega = \sqrt{\frac{\Omega_1^2}{\Delta^2} + \frac{\Omega_2^2}{\Delta^2}}
\] (3.3.21)

where \(\Omega_1 = \frac{w_1\pi}{(w_s/2)}\) and \(\Omega_2 = \frac{w_2\pi}{(w_s/2)}\) are the normalized frequency variables. \(H_F(\Omega)\) is a radially symmetric function with alternating positive and negative side lobes. An image that is filtered through \(H_F(\Omega)\), will not only experience an attenuation
Figure 3.16  
(a) Focus Blur Point Spread Function  
(b) Transfer Function of Focus Blur  
(c) Phase Shift Introduced by (b)
of higher frequency components, but also the frequencies in
the negative side lobe regions will experience a phase shift
of $\pi$ radians. The plot of $H_F(\Omega)$ along any radial direction
is shown in Figure 3.16(b). Figure 3.16(c) shows the phase
shift experienced by various frequency components of the
input.

In Figure 3.17(a) is shown the transfer function cor-
responding to focus blur, where the radius of the blur is 4
pixels. Using this transfer function, the original image of
Figure 3.17(b) is blurred and noise is added to the image
such that the SNR is equal to 30 db. The blurred noisy image
is as shown in Figure 3.17(c). For this blurred image, a
restoration filter response $H_{FR2}(\Omega_1,\Omega_2)$ was computed based on
the power spectrum equalization criterion. The response
is purely real and it is as shown in Figure 3.18(a). An
approximate form of the response along a radial direction of
45° with respect to the horizontal is shown in Figure 3.18(b).
In Figure 3.18(a), the intense dark regions and the intense
bright regions correspond to the most negative and most posi-
tive values, respectively of the function in Figure 3.18(b).
The restoration filter magnitude characteristic is the abso-
lute value of $H_{FR2}(\Omega_1,\Omega_2)$ and the phase characteristics is
as shown in Figure 3.18(c).

As in the case of motion blur, the desired restoration
filter phase characteristics is discontinuous in nature.
Such a response cannot be realized by a two dimensional re-
cursive digital filter having causal impulse response.
However, a possible recursive filter implementation to realize
Figure 3.17  
(a) Focus Blurring Transfer Function†, Blur Radius=4 Pixels 
(b) Original Image (Size 128x128 Pixels) 
(c) Focus Blurred Noisy Image; SNR=30db 

†Origin is at the centre of the image.
Figure 3.18  
(a) Restoration Filter Response
(b) Restoration Filter Response Along 45° Radial Direction
(c) Phase Shift Introduced by (b)

†Origin is at the centre.
the desired response is as shown in Figure 3.19(a). The implementation consists of two recursive filters in parallel having non-causal impulse response. The overall frequency response \( H(\Omega_1, \Omega_2) \) of the recursive filter implementation is given by:

\[
H(\Omega_1, \Omega_2) = |H_1(Z_1, Z_2)|^2 - |H_2(Z_1, Z_2)|^2\cdot
\]

\[
Z_1 = e^{-j\Omega_1}, \quad Z_2 = e^{-j\Omega_2}
\]

where

\[
|H_1(Z_1, Z_2)|^2 = H_1(Z_1, Z_2) \cdot H_1^*(Z_1, Z_2) \tag{3.3.23}
\]

\[
|H_2(Z_1, Z_2)|^2 = H_2(Z_1, Z_2) \cdot H_2^*(Z_1, Z_2)
\]

The impulse response of the overall filter \( H(\Omega_1, \Omega_2) \) is non-causal since \( H_1^*(Z_1, Z_2) \) and \( H_2^*(Z_1, Z_2) \) are filters with non-causal impulse response. The filtering by \( H_1^*(Z_1, Z_2) \) can be carried out in a manner described in Section 3.3.2b. The frequency response realized by \( H(\Omega_1, \Omega_2) \), after the approximation, is shown in Figure 3.19(b). The coefficients of the filter are shown in Figures 3.20 and 3.21. The filter was designed using the cascaded design procedure (described in Appendix F).

The result of restoration via convolution using FFT is shown in Figure 3.22(a) and Figure 3.22(b) shows the restoration using the recursive filter implementation. It is obvious that the restoration due to recursive filter implementation is a failure. The cause of this failure can
Figure 3.19  (a) Recursive Filter Implementation for Focus Deblur.
(b) Frequency Response of (a) (Origin Is At Centre).
Figure 3.20 Coefficients of Two Dimensional Filter $H_1(Z_1, Z_2)$. 
Figure 3.21 Coefficients of Two Dimensional Filter $H_2(z_1, z_2)$. 

<table>
<thead>
<tr>
<th>$A(0,0)$</th>
<th>.1000000000000000D+01</th>
<th>$B(0,0)$</th>
<th>.1000000000000000D+01</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A(0,1)$</td>
<td>.2554556675258177D-01</td>
<td>$B(0,1)$</td>
<td>.4759741951718458D-01</td>
</tr>
<tr>
<td>$A(1,0)$</td>
<td>.2554556675258177D-01</td>
<td>$B(1,0)$</td>
<td>.4759741951718458D-01</td>
</tr>
<tr>
<td>$A(1,1)$</td>
<td>.4597080653745180D+00</td>
<td>$B(1,1)$</td>
<td>.7001786814366320D+00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A(0,0)$</th>
<th>.1000000000000000D+01</th>
<th>$B(0,0)$</th>
<th>.1000000000000000D+01</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A(0,1)$</td>
<td>.548269821489491D-01</td>
<td>$B(0,1)$</td>
<td>.2912752161940441D-00</td>
</tr>
<tr>
<td>$A(1,0)$</td>
<td>.548269821489491D-01</td>
<td>$B(1,0)$</td>
<td>.2912752161940441D-00</td>
</tr>
<tr>
<td>$A(1,1)$</td>
<td>.457426178074469D+00</td>
<td>$B(1,1)$</td>
<td>.6901991752239320D+00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C(0,0)$</th>
<th>.1000000000000000D+01</th>
<th>$D(0,0)$</th>
<th>.1000000000000000D+01</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(0,1)$</td>
<td>.1545887255426392D+00</td>
<td>$D(0,1)$</td>
<td>.1721814622456480D+00</td>
</tr>
<tr>
<td>$C(0,2)$</td>
<td>.30851556994695D+00</td>
<td>$D(0,2)$</td>
<td>.5941255262595205D+00</td>
</tr>
<tr>
<td>$C(1,0)$</td>
<td>.1545887255426392D+00</td>
<td>$D(1,0)$</td>
<td>.1721814622456480D+00</td>
</tr>
<tr>
<td>$C(1,1)$</td>
<td>.209542969733181D-01</td>
<td>$D(1,1)$</td>
<td>.1672262759361820D+00</td>
</tr>
<tr>
<td>$C(1,2)$</td>
<td>.408658086468211D-00</td>
<td>$D(1,2)$</td>
<td>.1899755691485900D+00</td>
</tr>
<tr>
<td>$C(2,0)$</td>
<td>.30851556994695D+00</td>
<td>$D(2,0)$</td>
<td>.5941255262595205D+00</td>
</tr>
<tr>
<td>$C(2,1)$</td>
<td>.408658086468211D-00</td>
<td>$D(2,1)$</td>
<td>.1899755691485900D+00</td>
</tr>
<tr>
<td>$C(2,2)$</td>
<td>.1955700259014226D-00</td>
<td>$D(2,2)$</td>
<td>.5001965714888430D+00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C(0,0)$</th>
<th>.1000000000000000D+01</th>
<th>$D(0,0)$</th>
<th>.1000000000000000D+01</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(0,1)$</td>
<td>.9585811897624177D+00</td>
<td>$D(0,1)$</td>
<td>.4281989508218660D+00</td>
</tr>
<tr>
<td>$C(0,2)$</td>
<td>.1016472074156251D+00</td>
<td>$D(0,2)$</td>
<td>.2017354037914430D+00</td>
</tr>
<tr>
<td>$C(1,0)$</td>
<td>.9585811897624177D+00</td>
<td>$D(1,0)$</td>
<td>.4281989508218660D+00</td>
</tr>
<tr>
<td>$C(1,1)$</td>
<td>.104594496195385D+00</td>
<td>$D(1,1)$</td>
<td>.5655793276443980D+00</td>
</tr>
<tr>
<td>$C(1,2)$</td>
<td>.9875815690262747D+00</td>
<td>$D(1,2)$</td>
<td>.1749242814554080D+00</td>
</tr>
<tr>
<td>$C(2,0)$</td>
<td>.1016472074156251D+00</td>
<td>$D(2,0)$</td>
<td>.2017354037914430D+00</td>
</tr>
<tr>
<td>$C(2,1)$</td>
<td>.9875815690262747D+00</td>
<td>$D(2,1)$</td>
<td>.1749242814554080D+00</td>
</tr>
<tr>
<td>$C(2,2)$</td>
<td>.1168059244349930D+00</td>
<td>$D(2,2)$</td>
<td>.4855593858039680D+00</td>
</tr>
</tbody>
</table>
Figure 3.22 (a) Restoration by Convolution Via FFT. (b) Restoration Using Recursive Filter Implementation of Figure 3.19(a).
be attributed to the recursive filters $H_1(z_1, z_2)$ and $H_2(z_1, z_2)$, which are quarter plane recursive filters. As pointed out by Ekstrom and Woods [40], this class of filters cannot realize general magnitude characteristics. As can be seen from Figure 3.19(b), the approximation of the recursive filter implementation to the desired response is inadequate.

The class of filters, called half plane filters, appear to be more suitable for this type of application, and their use is suggested for further work in this area. Design techniques for half plane filters were not available at the time this work was performed and therefore the application of this class of filters is not considered here. At the time of completion of this thesis, several techniques for half plane filter design have been demonstrated in the literature, [41,42] which it is suggested, can be used to adequately approximate the required focus deblurring filter response.

3.3.3 Atmospheric Turbulence Blur

The cause of this type of blur is attributed to the variation in the refractive index of the atmosphere. Since the atmosphere is thermally non-uniform, its refractive index varies as function of both time and space. Therefore, in a strict sense, the point spread function corresponding to atmospheric turbulence blur is not only space variant, but also a function of time. However, it has been shown by Horner [43], that the image of a point source of light coming through atmosphere, when averaged over a period of time, has the form of a gaussian function and therefore, one can
write the point spread function corresponding to atmospheric
turbulence blur as:

\[ h(r) = e^{-kr^2} ; \quad r = \sqrt{x^2 + y^2} \]  \hspace{1cm} (3.3.24)

where \( x \) and \( y \) are spatial coordinates and \( k \) is a constant.
The fourier transform of the point spread function is also
gaussian and it is given by:

\[ H_A(w) = \frac{\pi}{k} e^{-\left(\frac{w^2}{4k}\right)} ; \quad w = \sqrt{w_1^2 + w_2^2} \]  \hspace{1cm} (3.3.25)

where \( w_1 \) and \( w_2 \) are spatial frequency variables. As before, choosing \( w_s \) as the sampling frequency in both spatial direc-
tions, the atmospheric turbulence blur transfer function
can be written as:

\[ H_A(\Omega) = \frac{\pi}{k} e^{-\left(\frac{\Omega^2}{4k}\right)} ; \quad \Omega = \sqrt{\Omega_1^2 + \Omega_2^2} \]  \hspace{1cm} (3.3.26)

where \( \Omega_1 = \left(\frac{w_1\pi}{w_s/2}\right) \) and \( \Omega_2 = \left(\frac{w_2\pi}{w_s/2}\right) \) are the normalized frequency
variables. The computer generated atmospheric turbulence blur
transfer function for a value of \( k = 2 \) is shown in Figure
3.23(a) and a plot of the cross section of the transfer func-
tion is shown in Figure 3.23(b). This transfer function is
used to blur the original image, shown in Figure 3.27(a).
Noise is added to the blurred image such that the signal to
noise ratio is 30 db. The noisy blurred image is shown in
Figure 3.27(b).

For the blurred image of 3.27(b), a restoration filter,
\( H_{AR}(\Omega) \), was computed based on the minimum mean squared error
criterion. The intensity plot of the specifications is shown
in Figure 3.24(a). An approximate variation of restoration
Figure 3.23 (a) Atmospheric Turbulence Blur Transfer Function
   (b) One Dimensional Plot of the Transfer Function Along Any Radial Direction

†Origin is at the centre.
filter specifications, along 45° radial direction is shown in Figure 3.24(b). The intense dark and the intense bright regions in 3.24(a) correspond to the zero and most positive values of the function shown in Figure 3.24(b). Since the blurring transfer function has zero or linear phase characteristics, the restoration filter also has zero or linear phase characteristics. Therefore, the linear programming technique was used in designing a causal recursive digital filter to the desired restoration filter magnitude and linear phase characteristics. The magnitude characteristics of the designed filter, and its coefficients, are shown in Figure 3.25(a) and (b). The impulse responses corresponding to the restoration filter frequency domain specifications and the recursive digital filter, designed using linear programming, are shown in Figure 3.26(a) and (b). The result of using the recursive filter for the purpose of restoration is shown in Figure 3.27(d). The restoration performed via convolution using FFT is shown in Figure 3.27(c).

One more example of atmospheric turbulence blur is considered, in which the image is blurred using the transfer function of (3.3.26) with \( k = 0.5 \). The SNR in the blurred noisy image is 20db. The original and the blurred noisy images are shown in Figures 3.30(a) and (b). In this example, a recursive filter implementation whose impulse response is non-causal, is used as opposed to the causal implementation of the previous example, in the restoration of the blurred image. The recursive filter imple-
Figure 3.24  
(a) Restoration Filter Response for Atmospheric Blur

(b) Plot of Magnitude of (a) Along 45° Radial Direction

*Origin is at the centre of the image.*
Figure 3.25  (a) Magnitude Response of the Designed Filter  
(b) Coefficients of the Designed Filter  

The array N corresponds to numerator coefficients and array D corresponds to denominator coefficients.
Figure 3.26  
(a) Impulse Response Corresponding to Desired Magnitude Specifications (16x16 pixels)  
(b) Impulse Response of the Designed Filter (the first 16x16 pixels)  

Note: For (a) the origin is at the centre of the image; for (b) the origin is at upper left hand corner. In (a) and (b) each square area of constant intensity represents one pixel.
Figure 3.27  (a) Original Image  
(b) Blurred Noisy Image; SNR=30 db  
(c) Restoration Using FFT  
(d) Restoration Using Recursive Filter
Figure 3.28  
(a) Recursive Filter Implementation for Atmospheric Turbulence Deblur  
(b) Desired Restoration Filter Frequency Response  
(c) Frequency Response Realized by (a)
Figure 3.29  (a) Impulse Response Corresponding to Figure 3.28(b)* (64 x 64 Pixels)
(b) Impulse Response Corresponding to Figure 3.28(c)* (64 x 64 Pixels)

* Origin is at upper left hand corner.
$a_0 = 0.1227548808826765D-01$

**CASCADE SECTION $++1$**

<table>
<thead>
<tr>
<th>$C(0,0)$</th>
<th>$1.000000000000000D+01$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(0,1)$</td>
<td>$3.786583422245172D+01$</td>
</tr>
<tr>
<td>$C(0,2)$</td>
<td>$9.246595737916343D+01$</td>
</tr>
<tr>
<td>$C(1,0)$</td>
<td>$3.639675385119787D+01$</td>
</tr>
<tr>
<td>$C(1,1)$</td>
<td>$1.225379673623823D+01$</td>
</tr>
<tr>
<td>$C(1,2)$</td>
<td>$4.705732027755850D+01$</td>
</tr>
<tr>
<td>$C(2,0)$</td>
<td>$3.005762355529315D+01$</td>
</tr>
<tr>
<td>$C(2,1)$</td>
<td>$3.618879697731638D+01$</td>
</tr>
<tr>
<td>$C(2,2)$</td>
<td>$1.181422828981420D+01$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$D(0,0)$</th>
<th>$1.000000000000000D+00$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D(0,1)$</td>
<td>$-9.309287746238103D+03$</td>
</tr>
<tr>
<td>$D(0,2)$</td>
<td>$-3.720848943959635D+03$</td>
</tr>
<tr>
<td>$D(1,0)$</td>
<td>$-1.001351756984938D+01$</td>
</tr>
<tr>
<td>$D(1,1)$</td>
<td>$-9.110536675183730D+03$</td>
</tr>
<tr>
<td>$D(1,2)$</td>
<td>$-3.245609230136470D+00$</td>
</tr>
<tr>
<td>$D(2,0)$</td>
<td>$-4.551665866647211D+00$</td>
</tr>
<tr>
<td>$D(2,1)$</td>
<td>$-4.086692213601412D+00$</td>
</tr>
<tr>
<td>$D(2,2)$</td>
<td>$-1.435237908845550D+00$</td>
</tr>
</tbody>
</table>

**Figure 3.29 (c) Coefficients of $H_1(z_1, z_2)$**
Figure 3.30  
(a) Original Image  
(b) Atmospheric Turbulance Blurred Image With Noise  
(c) Restoration Using FFT  
(d) Restoration Using Recursive Filter Implementation of Figure 3.28(a)
mentation is shown in Figure 3.28(a). The overall frequency response \( H(\Omega_1, \Omega_2) \) of the implementation is:

\[
H(\Omega_1, \Omega_2) = |H_1(z_1, z_2)|^2 = H_1(z_1, z_2) \cdot H_1^*(z_1, z_2); z_1 = 3^{-j\Omega_1}, z_2 = e^{-3j\Omega_2} \tag{3.3.27}
\]

\( H(\Omega_1, \Omega_2) \) is purely real and positive and has zero phase characteristics, and therefore, it can be used to approximate the desired restoration filter response. The filtering of data by \( H_1^*(z_1, z_2) \) is performed in a manner described in Section 3.3.b.

The desired restoration filter response as shown in Figure 3.28(b) and Figure 3.28(c) shows the response realized by the recursive filter implementation, after the approximation. The recursive filter was designed using the cascaded design procedure. The impulse responses corresponding to the desired restoration filter specifications, and the recursive filter implementation is shown in Figure 3.29(a) and (b). The coefficients of the recursive filter \( H_1(z_1, z_2) \) are shown in Figure 3.29(c). The restoration achieved using the recursive filter implementation is shown in Figure 3.30(d). The restoration that was carried out via convolution using the FFT is shown in Figure 3.30(c).

3.4 Summary

This chapter has presented the application of quarter plane recursive digital filters to problems in image processing. The applications considered are in the areas of image

+ The impulse responses are shifted by 16 pixels in each direction from the origin.
enhancement and image restoration. The problems of enhancement considered were high frequency emphasis and edge enhancement. In the case of restoration, the use of recursive filters are examined for motion, focus and atmospheric turbulence blurs. In the case of focus blur, the restoration appears to be a failure due to the inability of quarter plane filters to approximate the required filter response. However, in the case of motion and atmospheric turbulence blurs, it is shown that successful restorations can be obtained with recursive filter implementations.
CHAPTER IV
DISCUSSIONS OF RESULTS AND EXTENSIONS

4.1 Introduction

In Chapter II, the design of both one dimensional and quarter plane two dimensional recursive digital filters, using linear programming, was presented where a simultaneous approximation is carried out to both the desired magnitude and linear phase characteristics. The application of quarter plane two dimensional recursive digital filters to image processing problems were considered in Chapter III. This chapter presents a discussion of some of the problems and the advantages associated with the linear programming design method for recursive digital filters and also the problems and the computational efficiencies associated with restorations of blurred images using recursive digital filter implementations. Also some extensions of the research work are discussed towards the end of the chapter.

4.2 Discussion of the Filter Design Method

The design examples presented earlier have shown that linear programming can be successfully used to design recursive filters which approximate given frequency domain magnitude response with linear phase characteristic. However, as in the case of many other linear programming methods, there are some problems that limit the range of filters that can be designed using this method. The type of stability constraints given by (2.3.34) and (2.4.10), used in the one and two dimensional recursive filter design method respectively, are not general

131
stability constraints. Therefore, the filters that can be designed using this method belong to a subclass of all possible stable filter realizations. This is not a drawback in the case of two dimensional filters, since for many of the existing methods, the designed filters are also of a subclass of all possible stable filter realizations. However, the storage requirements for linear programming design of two dimensional filters are very large. Also, the computation times are quite high because of the large number of constraints involved in the linear program. Because of the high storage requirement, the range of the order of the two dimensional filters that can be designed using this method is limited. Thus, in this work, with 32 x 17 points of frequency domain specifications, the largest possible filter order that can be realized is M1=N2=N1=N2=3, (i.e., 16 coefficients in the numerator and 16 in the denominator). However, in the case of one dimension, this is not a problem, and filters of order 16 have been designed using this design procedure.

The design time for both one and two dimensional filters vary with the number of desired specification points and also the order of the filter. On the average, the time taken for one trial value of $\tau_s$ in the one dimensional filter design is approximately one minute. In the case of two dimensional filter design, the average time taken for one trial value of $\tau_c$ is approximately two minutes for specifications over a grid of points less than or equal to 21 x 11. However, for specifications over the grid points 32 x 17, the time for one trial value of $\tau_c$ is quite high (about ten minutes). All the designs presented in the examples were
carried out using an IBM 360/65 computer in single precision arithmetic.

In spite of the problems mentioned earlier, the method has advantages compared to non-linear optimization and some of the other linear programming design methods. In the case of methods that use non-linear optimization \([2,3,4,5,6,7,9]\) they require the specification of initial values of the parameters, to start the optimization, and also the final design is dependent on the initial values of the parameters. In addition, the convergence of the optimization is not guaranteed and even when the procedure converges, it generally converges to a local optimum. In comparison, the linear programming method of design does not require the specification of initial parameter values and also the optimum obtained is the absolute optimum consistent with the constraints of the problem. Compared to many of the linear programming methods \([8,17,18,19,20,21]\), where it is possible to approximate only a magnitude squared specifications, this method can approximate to both magnitude and linear phase specifications.

4.3 Discussion on the Applications of Recursive Digital Filters to Image Processing

4.3.a Image Enhancement Applications

In image enhancement applications, such as high frequency emphasis and edge enhancement, the filter phase characteristics are zero or linear over most of the spatial frequency domain. In Chapter III, it is demonstrated that the linear programming design can be successfully employed in designing the desired filters with linear phase character-
istics. It is also demonstrated that the linear phase characteristics are critical in edge enhancement applications. It should be noted that zero phase can also be achieved in the filters used for enhancement, by using the magnitude squared transfer function (non-causal) implementation shown in Figure 3.28(a). However, this results in almost twice the number of computations than that of the causal filter implementation.

4.3.b Image Restoration Applications

In Chapter III, suitable recursive filter implementations are shown for restorations due to motion, focus and atmospheric turbulence blurs. It is shown that the phase characteristics of the restoration filters are discontinuous in nature, in the case of motion and focus blurs, and therefore a non-causal impulse response filter implementation is required. However, in the case of atmospheric turbulence blur, the phase characteristics are either zero or linear phase and both causal and non-causal impulse response implementations are possible.

Prior to using recursive filter implementations for restoration purposes, the restorations were also carried out by convolution via FFT. The restorations so obtained, were used in judging the performance of the recursive filter implementations. In the case of motion blur, it can be seen from the examples provided that the restorations obtained are successful and are comparable to those obtained by convolution via FFT. In the case of atmospheric turbulence, it can be observed from the examples that both causal and non-
causal impulse response recursive filter implementations provide good restorations, and are almost equivalent to those obtained by convolution via FFT. The usefulness of the linear programming design method is also demonstrated in designing the restoration filter for causal impulse response recursive filter implementation. In addition to providing good restorations, the recursive filter implementations also provide significant computational savings compared to convolution via FFT. A brief comparison of the two types of implementations follows.

The size of kernel used in the restoration by convolution via the FFT is restricted to 32 x 32. Since the FFT array size is set to 64 x 64, the filtering can be performed for two 32 x 32 sections of the image at a time, where the input for the FFT is complex. Hence, according to 3.1, the number of complex operations involved for filtering an image of size 256 x 256 pixels is equal to 6,422,528 complex operations.\(^+\) In comparison, the number of real operations involved for the recursive filter implementation used, in the case of motion and atmospheric turbulence blurs, where successful restorations were obtained, are as in Table 4.1. It can be seen from this table that the number of computations required for recursive filtering is very much less than that required for the convolution via FFT. The average time required for filtering image data of size 256 x 256 using FFT was approximately equal to 32 minutes compared to an average of 8 minutes for the recursive filter implementations, using the Data

\(^+\) One complex operation is defined as one complex multiplication and an addition.
<table>
<thead>
<tr>
<th>Type of Blur</th>
<th>Order of the Two Dimensional Recursive Digital Filter</th>
<th>Number of Real Operations for Filtering</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOTION BLUR</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Example 1:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blur Length=4 Pixels</td>
<td>One 1st and One 2nd Order Filter Sections.</td>
<td>3,145,728</td>
</tr>
<tr>
<td>Angle=45° w.r.t. Horizontal</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Example 2:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blur Length=8 Pixels</td>
<td>Two 1st and One 2nd Order Filter Sections.</td>
<td>4,063,232</td>
</tr>
<tr>
<td>Angle=22.5° w.r.t. Horizontal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATMOSPHERIC TURBULANCE BLUR</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Example 1:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>k=2 In Equation (4.4.26)</td>
<td>One 3rd Order Filter</td>
<td>2,162,688</td>
</tr>
<tr>
<td><strong>Example 2:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>k=0.5 In Equation (4.4.26)</td>
<td>One 2nd Order Filter</td>
<td>2,228,224</td>
</tr>
</tbody>
</table>

Image Size = 256x256 Pixels

Table 4.1: Computations For Recursive Filtering (One Real Operation Is Defined As One Real Multiplication and Addition. The Calculations Are Obtained According to 31).
General NOVA-840 mini-computer.

In the case of focus blur, the restoration is a failure. The cause of this can be clearly seen in Figure 3.19(b), which is an approximation obtained by the recursive filter implementation to the desired restoration filter response of Figure 3.18(a). This indicates the inability of quarter plane recursive digital filters to approximate the nearly circularly symmetric restoration filter response. In the preliminary study reported in [32], the restorations for focus blur were obtained using the same type of recursive filter implementations as shown in this thesis. The restorations obtained were reasonably good in comparison to restoration by convolution via FFT; however, the original images used in the study were simple computer generated images which contained only 16 gray levels in comparison to the 256 levels of the original images used in this thesis.

4.4 Extensions

This section discusses the possible extensions and further study of the filter design technique, and the further applications of two dimensional recursive digital filters that are considered in this thesis.

4.4.a On Two Dimensional Filter Design

The thesis has presented the linear programming technique for quarter plane two dimensional recursive filter design. Therefore, one might consider the extension of this technique to design half plane filters, that were discussed in Chapter I of this thesis. The basic design procedure remains unaltered; however, the linear stability constraint
may require some modifications. These modifications will have to take into account the change in the stability criteria from quarter plane filters to that of half plane filters.

4.4.b On Recursive Digital Filter Applications In Image Processing

Chapter IV of the thesis considers the application of recursive digital filters in image processing; in particular, the application of quarter plane filters. However, the case of focus blur indicates the failure of quarter plane filter implementations. Hence, some further work is required, specifically in the area of half plane filter applications to image processing. It may also be worthwhile, not only to consider half plane filter application in focus deblurring, but in all the problems of image processing that have been studied in this thesis.
CHAPTER V
CONCLUSIONS

A new technique of designing one and two dimensional recursive digital filter transfer functions, which can approximate simultaneously linear phase and arbitrary magnitude specifications is presented. The technique sets up a linear programming problem which is in terms of the filter coefficients and the desired constant group delay (linear phase), which is then solved iteratively for the best approximation to the given specifications. Since the approximation is performed using the linear programming technique which is a linear optimization procedure, the constraints on the filter coefficients that can be used to obtain stable filters, are required to be linear in form. In Chapter II, such stability constraints are proposed and proofs of the sufficiency of these constraints are also provided. A wide range of design examples are provided in both one and two dimensional cases and some limitations of the linear programming design technique are also discussed.

The thesis also has presented the application of recursive digital filters to image processing. A brief evaluation of various two dimensional recursive digital filter techniques are carried out and it is shown that the linear programming technique of Chapter II and the non-linear optimization design technique of [4] are useful in designing filters for image processing applications. The applications of recursive digital filters are considered in the areas of a) image enhancement and b) image restorations. Examples of
enhancements using recursive filters are shown for the cases of high frequency emphasis and edge enhancement and importance of linear phase characteristics is stressed by way of an edge enhancement example. The applications of recursive digital filters in image restoration is considered for the cases of motion, focus and atmospheric turbulence blurs. Using suitable restoration filter specification, it is shown, with examples, that recursive digital filter implementations cannot only provide good restorations, but are also computationally advantageous compared to restoration by convolution via FFT, for the cases of motion and atmospheric turbulence blurs. In the case of focus blur, the restoration is a failure and the cause of the failure is attributed to the inability of quarter plane filters to adequately meet the circularly symmetric specifications of the desired restoration filter.
REFERENCES


APPENDIX A

Digital Filtering Fundamentals
One and Two Dimensional Filtering

Digital filters fall into two classes. Filters whose spatial response contains a finite number of non-zero samples are called Finite Impulse Response (FIR) filters, and those whose spatial response contains, in general, an infinite number of non-zero samples†, are called Infinite Impulse Response (IIR) or recursive filters.

In the one dimensional case, the output sequence \( y(m) \), of a FIR filter, assuming an input sequence \( x(m) \), is given by [1]:

\[
\begin{align*}
  y(m) &= \sum_{k=0}^{K-1} h(k) x(m-k) \\
  &\quad \quad \text{(A.1)}
\end{align*}
\]

where \( h(k) \) is the impulse response defined over the interval \( 0 \leq k \leq K-1 \). Similarly, in the two dimensional case, the output array \( y(m,n) \) can be written as [1]:

\[
\begin{align*}
  y(m,n) &= \sum_{k=0}^{K-1} \sum_{\ell=0}^{L-1} h(k,\ell) \cdot x(m-k,n-\ell) \\
  &\quad \quad \text{(A.2)}
\end{align*}
\]

where \( x(m,n) \) is the input array and \( h(k,\ell) \) is the impulse response defined over the interval \( 0 \leq k \leq (K-1), 0 \leq \ell \leq (L-1) \).

A one dimensional recursive filter is characterized by the difference equation [1]:

\[
\begin{align*}
  y(m) &= \sum_{k=0}^{N} a(k) x(m-k) - \sum_{\ell=1}^{M} b(\ell) y(m-\ell) \\
  &\quad \quad \text{(A.3)}
\end{align*}
\]

† This definition is used loosely, since convergent properties are required of the filter.
Once again \( x(m) \) and \( y(m) \) are the input and output respectively, and \( y(m) \) is computable (i.e., the IIR filter is said to be recursive) for \( m > 0 \). The impulse response, \( h(m) \), corresponding to (1.2.3) is causal, i.e., \( h(m) = 0 \) for \( m < 0 \) and it extends up to infinity for positive values of \( m \). However, an extension of similar characteristics to the two dimensional case yields only a specific class of two dimensional recursive filters. This will be shown in the following discussion.

In the two dimensional case, a general difference equation that characterizes a two dimensional IIR filter can be written as:

\[
y(m,n) = \sum_{(k,l) \in \mathbb{R}_a} a(k,l)x(m-k,n-l) - \sum_{(i,j) \in \mathbb{R}_b \setminus (i+j \neq 0)} b(i,j)y(m-i,n-j) \quad (A.4)
\]

where \( x \) and \( y \) are the input and output arrays respectively, and \( \mathbb{R}_a \) and \( \mathbb{R}_b \subset \mathbb{I}_s^2 \), where \( \mathbb{I}_s \) is the set of integers. Before discussing further about the difference equation (A.4), it is important to understand about the recursibility of a two dimensional recursive digital filter.

**Definition**: A two dimensional IIR filter is said to be recursive if for every output point \( (m,n) \), the output mask covers only points which have been previously computed.

In the above definition, the term output mask, simply refers to a rectangle that encloses the appropriate samples.
of the output array $y$ for computation of an output point $y(m,n)$ (an input mask can also be defined in a similar manner). Now, consider the following case of $R_a$ and $R_b$, such that:

$$R_a = \{(k,l) \mid 0 \leq k \leq Kl, 0 \leq l \leq Ll\}$$  \hspace{1cm} (A.5)

$$R_b = \{(i,j) \mid 0 \leq i \leq Il, 0 \leq j \leq Jl\}$$

The difference equation corresponding to this case can be written as

$$y(m,n) = \sum_{k=0}^{Kl} \sum_{l=0}^{Ll} a(k,l)x(m-k,n-l)$$

$$- \sum_{i=0}^{Il} \sum_{j=0}^{Jl} b(i,j)y(m-i,n-j), \ i+j \neq 0$$  \hspace{1cm} (A.6)

In (A.6), if $Il$, $Jl$, $Kl$, $Ll$ are all chosen to be equal to 2, then the recursive filtering operation corresponding to this case can be described by Figure 1.1. In this figure, to produce the value of $y(m,n)$, which is the desired output hole, the input and output masks are placed over the appropriate samples of the input and output arrays respectively and each sample is multiplied by the coefficient associated with that position in the mask and the products are summed. From the type of the output mask, it can be seen that the recursive filter associated with Equation (A.6), is recursive either column or rowwise. Consider, for example, another case of $R_a$ and $R_b$, such that,
Figure A1: An Example of Input and Output Masks to Calculate $y(3,3)$ for Equation (1.2.6).
\[ R_a = \{(k, \ell) \mid 0 \leq k \leq K2 \text{ if } 0 \leq \ell \leq L2 \} \]

and

\[ -K4 \leq k \leq -K3 \text{ if } 0 \leq \ell \leq L2 \]

\[ R_b = \{(i, j) \mid 0 \leq i \leq I2 \text{ if } 0 \leq j \leq J2 \} \]

and

\[ -I4 \leq i \leq -I3 \text{ if } 0 \leq j \leq J2 \]

This results in an output sequence \( y(m, n) \) given by:

\[
y(m, n) = \sum_{k=0}^{K2} \sum_{\ell=0}^{L2} a(k, \ell)x(m-k, n-\ell) + \sum_{k=-K3}^{-K4} \sum_{\ell=1}^{L2} a(k, \ell)x(m-k, n-\ell) - \sum_{i=0}^{I2} \sum_{j=0}^{J2} b(i, j)y(m-i, n-j) - \sum_{i=-I3}^{-I4} \sum_{j=1}^{J2} b(i, j)y(m-i, n-j)
\]

\[ i+j \neq 0 \]  

(A.8)

Here again, if I2=J2=I3=I4=K2=L2=K4=K3=1, then the recursive filtering operation given in (A.8) can be described by Figure A2. An examination of the output mask for this case indicates that this filter is recursive only columnwise.

According to Huang 14, a two dimensional recursive digital filter is said to be causal if the impulse response \( h(m, n) \) is zero for \( m \) or \( n \) less than zero. Returning back to Figure A1, one can see that impulse response of the IIR filter (A.6) is spread over only the upper quadrant of the right half plane in the spatial domain and therefore it is a causal filter. This type of filter is also referred to as a quarter plane recursive filter, since its impulse response
Figure A2  An Example of Input and Output Masks to Calculate $y(2,2)$ for the Equation of (A.8).
is confined to one quarter of the spatial domain. Compared to this, however, from Figure A2, it can be seen that the recursive filter of (A.8) has its impulse response spread over the right half plane of the spatial domain and it is referred to as a semicausal recursive filter with respect to the definition of causal filters. Because the impulse response of this type of filter spreads over half the spatial domain, it is also referred to as a half plane filter.

The work reported in this thesis deals with only the quarter plane filters and therefore, from here on, any reference to two dimensional recursive digital filter should be assumed to be a reference to a quarter plane two dimensional recursive digital filter, unless specified. For more detailed information on half plane filters, the interested reader should refer to recent works reported in 40, 41, 42.

Z Transforms and Filter Transfer Functions

Given a sequence \( x(n) \), defined for all \( n \), the Z transform is defined as,

\[
x(Z) = \sum_{n=-\infty}^{\infty} x(n)Z^n; \quad Z = e^{-ST}
\]

(A.9)

where \( S \) and \( Z \) are complex variables. In (A.9) \( x(n) \) is a sequence that is obtained by sampling a continuous signal \( x(t) \) once every \( T \) units of time. \( T \) represents the sampling period and its choice is based on the sampling theorem (see [1]). When the complex variable \( S \) and the sampling period \( T \) are such that
\[ S = jw; \quad T = \frac{2\pi}{w_s} \]  \hfill (A.10)

where \( w \) is the continuous frequency variable and \( w_s \) is the sampling frequency, then the complex variable \( Z \) can be written as:

\[ Z = e^{-j\Omega} \]  \hfill (A.11)

In (1.3.3) \( \Omega = (\pi w/w_s/2) \) is referred to as the normalized frequency variable. For various values of \( \Omega \), \( Z \) takes on values on the unit circle in the \( Z \) plane. Using (A.11) in (A.9), one finds that the evaluation of the \( Z \) transform on the unit circle results in:

\[ X(Z) \bigg|_{Z=e^{-j\Omega}} = X(e^{-j\Omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\Omega n} \]

which is the Fourier transform of the sampled sequence \( x(n) \).

Using the above definition, it is now possible to derive the transfer functions for FIR and recursive digital filters. Thus for a causal FIR filter whose impulse response \( h(n) \) is zero for values of \( n \) outside of the range \( 0 < n < (N-1) \), the transfer function can be written as:

\[ H(Z) = \sum_{n=0}^{(N-1)} h(n)z^n \]  \hfill (A.12)

Similarly, the transfer function corresponding to the causal recursive (IIR) filter is obtained as:

\[ H(Z) = \sum_{n=0}^{\infty} h(n)z^n = \sum_{i=0}^{N} a(i)z^i \div \left( 1 + \sum_{j=1}^{M} b(j)z^j \right) \]  \hfill (A.13)

\[ \text{†Expressed in radians.} \]
where we assume that no roots of the denominator are cancelled by roots of the numerator. Most often \( N \) is less than or equal to \( M \) and the filter is referred to as an \( M^{th} \) order recursive filter. If \( N > M \), then the filter can be taken to be an \( M^{th} \) order recursive filter with an \( (N-M)^{th} \) order FIR filter. Another important quantity in filter theory is the frequency response \( H(\Omega) \) of the filter. This is obtained by evaluating the transfer function \( H(Z) \) on the unit circle in the \( Z \) plane, i.e., \( Z = e^{-j\Omega} \).

A two dimensional \( Z \) transform can be defined in exactly the same manner as the one dimensional \( Z \) transform. Hence, for a two dimensional sequence \( x(m,n) \), defined for all \( m \) and \( n \), the \( Z \) transform is defined as:

\[
X(Z_1, Z_2) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x(m,n) z_1^m z_2^n; \quad Z_1 = e^{-S_1T_1}, \quad Z_2 = e^{-S_2T_2}
\]  

(A.14)

where \( S_1, S_2, Z_1 \) and \( Z_2 \) are complex variables.

\( x(m,n) \) is a sequence obtained by sampling a continuous two dimensional signal \( x(t_1, t_2) \) at intervals of \( T_1 \) and \( T_2 \) units in spatial directions \( x \) and \( y \). With complex variables \( S_1, S_2 \) and the sampling periods \( T_1 \) and \( T_2 \), so that:

\[
S_1 = j\omega_1, \quad S_2 = j\omega_2, \quad T_1 = \left(\frac{2\pi}{\omega_{s_1}}\right), \quad T_2 = \left(\frac{2\pi}{\omega_{s_2}}\right)
\]  

(A.15)

where \( \omega_1 \) and \( \omega_2 \) are continuous spatial frequency variables and \( \omega_{s_1} \) and \( \omega_{s_2} \) are the frequencies at which the signal \( x(t_1, t_2) \) is sampled in \( x \) and \( y \) spatial directions, the complex variables \( Z_1 \) and \( Z_2 \) can be written as:
\[ z_1 = e^{-j\Omega_1}; \quad \Omega_1 = \pi w_1/(w_{s1/2}) \]
\[ z_2 = e^{-j\Omega_2}; \quad \Omega_2 = \pi w_2/(w_{s2/2}) \]  

(A.16)

In (A.17) \( \Omega_1 \) and \( \Omega_2 \) are referred to as normalized spatial frequency variables in radians and for various values of \( \Omega_1 \) and \( \Omega_2 \), \( z_1 \) and \( z_2 \) take on values on the unit circles in the \( z_1 \) and \( z_2 \) planes respectively. Therefore from (A.14) the evaluation of \( Z \) transform on the unit bidisc \(^+\) results in:

\[ X(z_1, z_2) |_{z_1 = e^{-j\Omega_1}} = x(e^{-j\Omega_1}, e^{-j\Omega_2}) \]

\[ z_2 = e^{-j\Omega_2} \]

\[ = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x(m, n) e^{-j(m\Omega_1 + n\Omega_2)} \]  

(A.17)

which is the two dimensional fourier transform of the sequence \( x(m, n) \).

It follows from the above discussion, that given a FIR filter whose impulse response \( h(m, n) \) is zero outside the region \( 0 \leq m \leq M_l \) and \( 0 \leq n \leq N_l \), the two dimensional transfer function can be obtained as:

\[ H(Z_1, Z_2) = \sum_{m=0}^{M_l} \sum_{n=0}^{N_l} h(m, n) Z_1^m Z_2^n \]  

(A.18)

For a two dimensional recursive digital filter, the transfer function can be obtained from the difference equation of (A.4) as:

\(^+\) Unit bidisc \( \Delta \{z_1, z_2: |z_1| < 1 \text{ and } |z_2| > 1 \} \)
\[
H(z_1, z_2) = \frac{\sum_{(k, \ell)} a(k, \ell) z_1^k z_2^\ell}{1 + \sum_{(i, j)} b(i, j) z_1^i z_2^j}
\]
(A.19)

Depending upon the choice of \( R_a \) and \( R_b \) the filter of (A.19) can either be a half plane or a quarter plane digital filter. The frequency response \( H(\Omega_1, \Omega_2) \) of (A.18) or (A.19) can be obtained by evaluating the respective transfer functions around the unit bidisc.

**Order of Two-Dimensional Recursive Digital Filter**

Unlike the one dimensional case, the general approach in the two dimensional case has been to specify the number of filter coefficients, instead of an order for the filter. However, as in the approach of some authors 2,3,4,5, it is convenient to specify an order for the two dimensional filter for a specific case of the sets \( R_a \) and \( R_b \) of (A.19). Hence if \( R_a \) and \( R_b \) are as indicated in (A.5), where \( K_1, L_1, L_1 \) and \( J_1 \) are all chosen to be equal to an integer \( K \), then the filter of (A.19) is referred to as a quarter plane filter of order \( K \). A straightforward extension of this can be carried out to the case of half plane filters. The above notation is adopted, throughout this thesis, in referring to the order of a two dimensional recursive digital filter. In situations when \( R_a \) and \( R_b \) do not belong to the specific case indicated above (which is not very common), the filter is referred to in terms of the number of numerator and
denominator coefficients.

Group Delay of a Filter

The group delay of a filter is a measure of average spatial or time delay as a function of frequency.

The transfer function $H(Z)$ of a one dimensional filter can be written in the form:

$$H(Z) = |H(Z)| e^{j\beta(Z)} \quad (A.20)$$

where $|H(Z)|$ is the magnitude and $\beta(Z)$ is the phase response of $H(Z)$. The phase response of the filter is defined as:

$$\beta(\Omega) = \beta(Z) \bigg|_{Z=e^{-j\Omega}} = \tan^{-1} \left( \frac{\text{Im} \ H(Z)}{\text{Re} \ H(Z)} \right) \bigg|_{Z=e^{-j\Omega}} \quad (A.21)$$

The group delay is now defined as:

$$\tau(\Omega) = -\frac{d\beta(\Omega)}{d\Omega} \quad (A.22)$$

$\tau(\Omega)$ can be expressed in terms of $H(Z)$ as:

$$\tau(\Omega) = \text{Re} \left\{ \frac{Z}{H(Z)} , \frac{dH(Z)}{dZ} \right\} \bigg|_{Z=e^{-j\Omega}} \quad (A.23)$$

The group delays for two dimensional filters can be defined in a manner similar to the one dimensional case. For a two dimensional filter, there exists group delays in each of the spatial directions and are functions of both the spatial frequencies.

Consider a two dimensional filter transfer function

\[ \text{Im}\{\} \text{ and } \text{Re}\{\} \text{ refers to imaginary part of and real part of, respectively.} \]
expressed in a form similar to (A.20) as:

\[ H(z_1, z_2) = |H(z_1, z_2)| e^{j\beta(z_1, z_2)} \]  

(A.24)

As before, the phase response is defined as:

\[ \beta(\Omega_1, \Omega_2) = \beta(z_1, z_2) \Bigg|_{\begin{array}{c} z_1 = e^{-j\Omega_1} \\
                z_2 = e^{-j\Omega_2} \end{array}} \]

\[ = \tan^{-1} \left( \frac{\text{Im} H(z_1, z_2)}{\text{Re} H(z_1, z_2)} \right) \Bigg|_{\begin{array}{c} z_1 = e^{-j\Omega_1} \\
                z_2 = e^{-j\Omega_2} \end{array}} \]  

(A.25)

The group delays can now be defined as (9):

\[ \tau_1(\Omega_1, \Omega_2) = -\frac{\partial \beta(\Omega_1, \Omega_2)}{\partial \Omega_1} \]  

(A.26)

\[ \tau_2(\Omega_1, \Omega_2) = -\frac{\partial \beta(\Omega_1, \Omega_2)}{\partial \Omega_2} \]

shows the spatial frequency domain and the corresponding delay directions in the spatial domain.

\( \tau_1(\Omega_1, \Omega_2) \) and \( \tau_2(\Omega_1, \Omega_2) \) can be expressed in terms of the two dimensional filter transfer function \( H(z_1, z_2) \) as:

\[ \tau_1(\Omega_1, \Omega_2) = \text{Re} \left( \frac{z_1}{H(z_1, z_2)} \cdot \left. \frac{\partial H(z_1, z_2)}{\partial z_1} \right|_{\begin{array}{c} z_1 = e^{-j\Omega_1} \\
                z_2 = e^{-j\Omega_2} \end{array}} \right) \]  

(A.27)
(a) Frequency Domain

(b) Spatial Domain

Figure A3  Directions of Delays in Spatial Domain.
\[ \tau_2(\Omega_1, \Omega_2) = \text{Re} \left\{ \frac{z_2}{H(z_1, z_2)} \frac{\partial H(z_1, z_2)}{\partial z_2} \right\} \biggr| \begin{align*} z_1 &= e^{-j\Omega_1} \\ z_2 &= e^{-j\Omega_2} \end{align*} \] (A.28)

**Stability of IIR Digital Filters**

**One Dimensional Case**

In the case of one dimensional filters, a necessary and sufficient condition for stability is:

\[ \sum_{n=0}^{\infty} |h(n)| < \infty \] (A.29)

where \( h(n) \) is the impulse response of the given filter.

FIR filters satisfy the above condition. Since their impulse response is defined only over a short period, i.e., \( h(n) \) is defined for \( N_1 < n < N_2 \). However, in the case of recursive filters, the impulse response extends up to infinity and in order to meet the stability criterion, it is required that poles of the transfer function be outside the unit circle.

Thus given an IIR filter,

\[ H(z) = \frac{P(z)}{Q(z)} = \frac{\sum_{n=0}^{N} a(n) z^n}{\sum_{m=0}^{M} b(m) z^m}, \quad z = e^{-j\Omega} \] (A.30)

in order for this to be stable, the singularities (often referred to as zeros) of \( Q(z) \) should lie outside the unit
circle in the \( Z \)-plane. Therefore in designing recursive filters of the form given in (A.30), proper stability constraints have to be imposed on the coefficients \( b(m) \), so that the zeros of \( Q(z) \) lie outside the unit circle in the \( Z \)-plane.

**Two Dimensional Case**

In the two-dimensional case, conditions for stability become more involved. The following definition 41, is useful in the statement of stability theorems for two-dimensional recursive filters.

**Definition 41:** In the two-dimensional complex space, we define the following spatial regions by:

\[
T^2 = \{(z_1, z_2) \mid |z_1| = 1, \quad |z_2| = 1\};
\]
\[
D^{++} = \{(z_1, z_2) \mid |z_1| < 1, \quad |z_2| < 1\};
\]
\[
D^{+-} = \{(z_1, z_2) \mid |z_1| > 1, \quad |z_2| < 1\};
\]
\[
D^{-+} = \{(z_1, z_2) \mid |z_1| < 1, \quad |z_2| > 1\};
\]
\[
D^{--} = \{(z_1, z_2) \mid |z_1| > 1, \quad |z_2| > 1\};
\]

\( T^2 \) is generally referred to as the unit bicircle.

Now, let \( H(z_1, z_2) \) be a rational two-dimensional transfer function such that:

\[
H(z_1, z_2) = \frac{A(z_1, z_2)}{B(z_1, z_2)} = \sum_{(k, \ell) \in \mathcal{R}_a} a(k, \ell) z_1^k z_2^\ell / \sum_{(i, j) \in \mathcal{R}_b} b(i, j) z_1^i z_2^j
\]

(A.32)
where \( z_1 = e^{-j\Omega_1} \) and \( z_2 = e^{-j\Omega_2} \) and \( R_a, R_b \) are as defined in (1.2.5) and (1.2.7). The stability theorems can then be stated as follows:

**Theorem 1.41:** A causal (quarter plane) filter of definition given in 14 is stable if \( B(z_1, z_2) \neq 0 \) for \( (z_1, z_2) \in D^+ \).

The above theorem is the same as Shanks' stability theorem for causal filters 14.

**Theorem 2.41:** A semicausal filter is said to be stable if \( B(z_1, z_2) \neq 0 \) for \( (z_1, z_2) \in (D^+ D^-) \) and \( B(z_1, 0) \neq 0 \) for \( |z_1| < 1 \).
APPENDIX B

Digital Image Processing Fundamentals
**Digital Image Processing**

Image processing is a broad area that deals with manipulation of pictorial data, which are inherently two dimensional in nature. It encompasses various areas such as image enhancement, restoration, pictorial pattern recognition and efficient picture coding for picture transmission and storage.

In order to understand various techniques of digital image processing, first it is important to understand some of the basic concepts in digital image processing such as image formation, recording and sampling. A brief description of the above follows.

The principle elements of an image formation system can be described as shown in Figure B1. The black box in Figure 1.4 acts upon a radiant energy component of the object to generate the image. Thus the image at point \((x,y)\) can be considered as a function of contributions in a (possibly infinite) neighbourhood of \((x_1,y_1)\). If \(g_i(x,y)\) represents image radiant energy distribution and \(i(x,y)\) represents object radiant energy distribution, then a general description of \(g_i\) is given by:

\[
g_i(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x,y,x_1,y_1) i(x_1,y_1) dx_1 dy_1 \quad (B.1)
\]

In (B.1), \(h\) is assumed to merely weigh the object distribution as a scalar multiplier and \(h\) is referred to as a point spread function. As such \(h\) in (B.1) represents a space variant point spread function. If \(h\) is made position invariant however, then \(g_i\) can be expressed as:
\[ g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \{h(x-x_1, y-y_1) \ i(x_1, y_1)\} \ dx_1 \ dy_1 \] (B.2)

This is true in situations such as optical image formation systems.

The two major means of recording images are photochemical and photoelectric. Example of photoelectric recording is a television camera and in the case of photochemical, photographic film. Part of this dissertation dealing with image processing is concerned with images that are recorded in a photographic film and therefore the process of image formation and recording associated with this case can be described by a model shown in Figure B2. This is a simplified block diagram that shows the image formation system with spatial response \( h \). The intensity of the recorded image is given by:

\[ g(x, y) = h(x, y) \ast i(x, y) + n(x, y) \] (B.3)

where \( n(x, y) \) is noise, which is modeled as additive. Although the modelling of noise as additive may not always reflect reality (an example is film grain noise [38], which is multiplicative) it does however appear to provide good results in most image processing problems [23].

The intensities \( g(x, y) \), recorded in the film can now be obtained by projecting a spot of light in a raster scanning sequence, e.g., left-to-right, top-to-bottom and sampling the intensity of the transmitted light at given coordinate spacings. In a system such as flying spot scanner,
the light that is transmitted through the film is received by a photomultiplier-tube, which generates an equivalent electrical signal at its output. This electrical output is converted to an integer number, via an analog to digital converter (A/D) which is then stored in a digital computer. Thus a digital image is formed as a two dimensional array of numbers representing individual brightness values at given co-ordinate spacings of the original continuous image.
APPENDIX C

Proof of Stability Criterion For One
Dimensional Filter Design
Proof for the sufficiency of the one dimensional stability criterion:

As indicated in Chapter III, the stability constraint used in the design of one dimensional filter design is:

\[ \text{Re}(Q(Z)) > 0 \text{ for } |Z| = 1 \quad (C.1) \]

where \( Q(Z) \) is the denominator polynomial of the recursive digital filter transfer function given by:

\[ H(Z) = \frac{P(Z)}{Q(Z)} = \frac{a_0 + a_1 Z + a_2 Z^2 + \ldots + a_n Z^n}{b_0 + b_1 Z + b_2 Z^2 + \ldots + b_M Z^M} \]

It is required to prove that if (C.1) is given to be true, then \( Q(Z) \) is non-zero (i.e., it has no zeroes) inside the unit circle in the \( Z \) plane.

Proof:

Consider the Figure Cl. From Cauchy's integral formula, at any point \( Z_0 \) inside the unit circle in \( Z \) plane, \( Q(Z_0) \) is given by:

\[ Q(Z_0) = \int_{|Z|=1} \frac{Q(Z)}{(Z-Z_0)} \, dZ \quad (C.2) \]

Reparameterizing the contour of integration, Figure Cl, i.e., with \( Z = e^{i\theta} \) and also with \( Z_0 = re^{i\phi} \), where \( r < 1 \), (C.2) can be rewritten as:

\[ Q(re^{i\phi}) = \frac{1}{2\pi i} \int_{0}^{2\pi} \frac{Q(e^{i\theta})}{e^{i\theta} - re^{i\phi}} \, ie^{i\theta} \, d\theta \]

which can be rewritten as:
\[
Q(re^{i\phi}) = \frac{1}{2\pi} \int_{0}^{2\pi} \frac{Q(e^{it})}{1-re^{i(\phi-t)}} \, dt \quad (C.3)
\]

(C.3) can also be expressed as:

\[
Q(re^{i\phi}) = \frac{1}{2\pi} \int_{0}^{2\pi} Q(e^{it}) \cdot \sum_{n=0}^{\infty} r^{n} e^{i(\phi-t)n} \, dt \quad (C.4)
\]

\[Q(e^{it}) \text{ is given by:}\]

\[
Q(e^{it}) = \sum_{m=0}^{M} b_{m} e^{imt} = b_{0} + b_{1} e^{it} + b_{2} e^{i2t} + \ldots + b_{M} e^{iMt}
\]

(C.5)

in which case it is true that:

\[
\frac{1}{2\pi} \int_{0}^{2\pi} Q(e^{it}) \cdot \sum_{n=-\infty}^{\infty} r^{-n} e^{i(\phi-t)n} \, dt = 0 \quad (C.6)
\]

Adding (C.6) to the right hand side of (C.3) does not alter the value of \(Q(re^{i\phi})\) and therefore \(Q(re^{i\phi})\) can be expressed as:

\[
Q(re^{i\phi}) = \frac{1}{2\pi} \int_{0}^{2\pi} Q(e^{it}) \sum_{n=0}^{\infty} r^{n} e^{i(\phi-t)n} + \sum_{n=-\infty}^{\infty} r^{-n} e^{i(\phi-t)n} \, dt
\]

(C.7)

In (C.7), the quantity:

\[
\sum_{n=0}^{\infty} r^{n} e^{i(\phi-t)n} + \sum_{n=-\infty}^{\infty} r^{-n} e^{i(\phi-t)n} = p_{r}(\phi-t) \quad (C.8)
\]

where \(p_{r}(\phi-t)\) is called the Poisson Kernel. One of the properties of the Poisson Kernel is that it is always positive for \(r < 1\).
i.e., \[ P_r(\phi-t) > 0 \quad \forall \ r < 1 \] (C.9)

Therefore, in (C.7), one can see that, if

\[ \text{Re}\{Q(z)\} = \text{Re}\{Q(e^{it})\} > 0 \quad 0 \leq t \leq 2\pi \] (C.10)

then,

\[ Q(re^{i\phi}) > 0 \quad \forall \ r < 1 \]

Therefore, it is proven that if,

\[ \text{Re}\{Q(z)\} > 0 \text{ for } |z| = 1 \]

then \( Q(z) \) has no zeros inside the unit circle in the Z plane.

**Figure C1**
APPENDIX D

On The Stability Criterion For Second Order One Dimensional Filter
Comparison of two types of stability constraints for the second order case:

Consider a second order filter transfer function:

\[ H(z) = \frac{P(z)}{Q(z)} = \frac{a_0 + a_1 z + a_2 z^2}{1 + b_1 z + b_2 z^2}; \quad z = e^{-j\Omega} \quad (D.1) \]

where \( \Omega \) is the normalized frequency variable.

Consider now, the type (a) constraint. Accordingly, the coefficients of \( Q(z) \) would be constrained as follows:

\[ 1 > b_1 > b_2 > 0 \quad (D.2) \]

Inequality (D.2) can be split into several inequalities as follows:

\[ b_1 < 1 \quad (D.3) \]
\[ b_1 - b_2 > 0 \quad (D.4) \]
\[ b_2 > 0 \quad (D.5) \]

The solution region for the inequalities (D.3) through (D.5) is shown in Figure D1(a). As one can see, the solution region is dependent upon \( b_1 \). The solution region gets smaller and smaller as \( b_1 \) decreases and therefore the type of stable filters one can design is very limited. Now consider the type (b) constraint. The coefficients of \( Q(z) \), according to type (b) constraint, would be constrained as follows:

\[ 1 + b_1 \cos \Omega + b_2 \cos 2\Omega > 0 \; ; \; 0 \leq \Omega \leq \pi \quad (D.6) \]

Substituting various values for \( \Omega \) we can get several inequa-
Figure D.1 Solution Region Corresponding to Various Types of Constraints.
qualities, which, to a considerable extent, is sufficient to determine the convex polyhedron solution region. Thus (D.6) for various values of $\Omega$ becomes:

$$\Omega = 0 \quad 1 + b_1 + b_2 > 0 \quad \text{i.e., } b_1 + b_2 > -1 \quad (D.7)$$

$$\Omega = \frac{\pi}{6} \quad 1 + \frac{3b_1}{2} + \frac{b_2}{2} > 0 \quad 3b_1 + b_2 > -2 \quad (D.8)$$

$$\Omega = \frac{\pi}{3} \quad 1 + \frac{b_1}{2} - \frac{b_2}{2} > 0 \quad b_1 - b_2 > -2 \quad (D.9)$$

$$\Omega = \frac{\pi}{4} \quad 1 + \frac{b_1}{2} > 0 \quad b_1 > -2 \quad (D.10)$$

$$\Omega = \frac{\pi}{2} \quad > 0 \quad b_2 < 1 \quad (D.11)$$

$$\Omega = \frac{2\pi}{3} \quad 1 - \frac{b_1}{2} - \frac{b_2}{2} > 0 \quad b_1 + b_2 < 2 \quad (D.12)$$

$$\Omega = \frac{3\pi}{4} \quad 1 - \frac{b_1}{2} > 0 \quad b_1 < 2 \quad (D.13)$$

$$\Omega = \pi \quad 1 - b_1 + b_2 > 0 \quad -b_1 + b_2 > -1 \quad (D.14)$$

Constraints on $b_1$ and $b_2$ for other values of $\Omega$ do not seem to alter the result significantly and Figure D1(b) shows the solution region for inequalities of (D.7) through (D.14).

Finally, according to Jury [34], the general constraints on coefficients of $Q(z)$ for a stable filter is as follows:

$$b_1 + b_2 > -1 \quad (D.15)$$

$$-b_1 + b_2 > -1 \quad (D.16)$$
\[ b_2 < 1 \]  \hspace{1cm} (D.17)

The solution region corresponding to these constraints is shown in Figure D1(c).

The assertion from Figure D1 is therefore the constraint type (b) offers the design of larger subclass of stable filter than type (a) constraint.
APPENDIX E

Proof of the Sufficiency of the Two Dimensional Stability Constraint
The stability constraint used in the two dimensional design method is:

\[ \text{Re} \{B(Z_1, Z_2)\} > 0 \text{ for } |Z_1| = 1, \ |Z_2| = 1 \quad (E.1) \]

where \( B(Z_1, Z_2) \) is the denominator of a two dimensional recursive digital filter transfer function, given by:

\[
H(Z_1, Z_2) = \frac{P(Z_1, Z_2)}{Q(Z_1, Z_2)} = \sum_{m=0}^{M1} \sum_{n=0}^{N1} a_{mn} Z_1^m Z_2^n
\]
\[
\sum_{m=0}^{M2} \sum_{n=0}^{N2} b_{mn} Z_1^m Z_2^n
\]

It is required to show that if (E.1) is true, then \( B(Z_1, Z_2) \) satisfies the Shank's stability theorem 14; i.e.,

\[ B(Z_1, Z_2) \neq 0 \text{ for } |Z_1| \leq 1 \text{ and } |Z_2| \leq 1 \]

Proof:

Consider a point, \( Z_2 = Z_{2a} \), on the unit circle in the \( Z_2 \) plane of Figure E1. Now, for any \( Z_2 = Z_{2a} \), \( |Z_2| = 1 \), \( B(Z_1, Z_2) \) is a polynomial in \( Z_1 \) only, i.e., \( f_1(Z_1) = B(Z_1, Z_{2a}) \), where the coefficients of \( f_1(Z_1) \) are, in general, complex.

From (A.1), it is then true, that,

\[ \text{Re} \{f_1(Z_1)\} > 0 \text{ for } |Z_1| = 1 \text{ and for any } Z_{2a}, \ |Z_{2a}| = 1 \]

(E.2)

Therefore, using the proof for the one dimensional case from Appendix C, it can be clearly seen that \( f_1(Z_1) \) is non-zero inside the unit circle in \( Z_1 \) plane, i.e.,

179
\[ \text{Re}\{f_1(Z_{10})\} > 0 \text{ for any } |Z_{10}| < 1 \quad (E.3) \]

Therefore, it is true that:

\[ B(Z_1, Z_2) \neq 0 \text{ for } |Z_1| < 1 \text{ and } |Z_2| = 1 \quad (E.4) \]

Consider now, the function \( f_2(Z_2) \) such that:

\[ f_2(Z_2) = B(Z_{10}, Z_2) \quad \forall \quad |Z_{10}| < 1 \text{ and } |Z_2| = 1 \quad (E.5) \]

From (E.3), it is clear that:

\[ \text{Re}\{f_2(Z_2)\} > 0 \quad \forall \quad |Z_2| = 1 \quad (E.6) \]

Therefore, once again, using the one dimensional proof, it can be said that for any \( Z_{20} \): \( |Z_{20}| < 1 \).

\[ f_2(Z_{20}) \neq 0 \quad (E.7) \]

i.e., \[ B(Z_{10}, Z_{20}) \neq 0 \text{ for any } Z_{10}; \quad |Z_{10}| < 1; \text{ and} \]

\[ \text{for any } Z_{20}; \quad |Z_{20}| < 1 \quad (E.8) \]

Thus, it is proved that:

\[ B(Z_1, Z_2) \neq 0 \quad \forall \quad |Z_1| < 1 \text{ and } |Z_2| < 1 \]

if \[ \text{Re}\{B(Z_1, Z_2)\} > 0 \quad \forall \quad |Z_1| = 1 \text{ and } |Z_2| = 1 \]

\[ \text{Figure E1} \]
APPENDIX F

Two Dimensional Recursive Digital Quarter Plane Filter
Design in Cascade Form
This appendix describes the frequency domain approximation of two dimensional recursive digital filters, using the technique of Ramamoorthy and Bruton [4]. The filters designed by this approach are realized in cascade form.

**Form of the Recursive Filter:**

The two dimensional recursive filter transfer function used in the approximation procedure is of the form:

\[ H(Z_1, Z_2) = A \cdot \prod_{k=1}^{K} H_{1_k}(Z_1, Z_2) \cdot \prod_{\ell=1}^{L} H_{2_\ell}(Z_1, Z_2) \]  

where \( Z_1 = e^{-j\Omega_1}, Z_2 = e^{-j\Omega_2} \) and \( \Omega_1, \Omega_2 \) are the normalized frequency variables. \( H_{1_k}(Z_1, Z_2) \) and \( H_{2_\ell}(Z_1, Z_2) \) are the first and second order filter sections, and \( A \) is the gain factor. The first and second order filter sections are of the form:

\[ H_{1_k}(Z_1, Z_2) = \frac{1}{\sum_{m=0}^{1} \sum_{n=0}^{1} a_k(m,n)Z_1^mZ_2^n} \]  

\( (F.2) \)

and

\[ H_{2_\ell}(Z_1, Z_2) = \frac{2}{\sum_{m=0}^{2} \sum_{n=0}^{2} c_\ell(m,n)Z_1^mZ_2^n} \]  

\( (F.3) \)

Central to the technique of [10] is the problem of obtaining the filter transfer functions of (F.2) and (F.3) in the stable form. The procedure required to obtain stable \( H_{1_k}(Z_1, Z_2) \) and \( H_{2_\ell}(Z_1, Z_2) \) is described in the following
A. Stable First and Second Order Transfer Functions

As shown in [4], a stable first order filter transfer function $H_k(z_1, z_2)$ or a stable second order transfer function $H_k(z_1, z_2)$ can be obtained by transforming a two dimensional analog transfer function $H(S_1, S_2)$ of corresponding order, via the double bilinear transformation, where:

$$H(S_1, S_2) = \frac{P(S_1, S_2)}{Q(S_1, S_2)} \text{ and } Q(S_1, S_2) \neq 0 ;$$  \hspace{1cm} (F.4)

for $\text{Re}\{S_1\} > 0, \text{Re}\{S_2\} > 0; S_1 = jw_1, S_2 = jw_2$

The double bilinear transformation is given by:

$$S_1 = \frac{1 - z_1}{1 + z_1} \quad ; \quad S_2 = \frac{1 - z_2}{1 + z_2} \hspace{1cm} (F.5)$$

First Order Case:

Consider a first order analog transfer function, given by:

$$H_k(S_1, S_2) = \frac{p_{1k}(S_1, S_2)}{q_{1k}(S_1, S_2)} = \frac{p_{1k} + p_{2k} S_2 + p_{3k} S_1 + S_1 S_2}{q_{1k} + q_{2k} S_2 + q_{3k} S_1 + S_1 S_2} \hspace{1cm} (F.6)$$

In order that (F.6) satisfy the condition given in (F.4), it is required that the coefficients of $Q_k(S_1, S_2)$ be expressed in terms of a set of non-zero variables $(x_{1k}, x_{2k}, x_{3k})$ such that:

$$q_{1k} = x_{1k}^2; q_{2k} = x_{2k}^2; \text{ and } q_{3k} = x_{3k}^2 \hspace{1cm} (F.7)$$
Using the double bilinear transform on (F.6) and also using (F.7), the coefficients of the stable $H_k(Z_1, Z_2)$, which is of the form given by (F.2), can be written as follows:

\[
\begin{align*}
  a_k(0,0) &= (p_{1k}^2 + p_{2k}^2 + p_{3k}^2 + 1) \\
  b_k(0,0) &= (x_{1k}^2 + x_{2k}^2 + x_{3k}^2 + 1) \\
  a_k(0,1) &= (p_{1k}^2 - p_{2k}^2 + p_{3k}^2 - 1) \\
  b_k(0,1) &= (x_{1k}^2 - x_{2k}^2 + x_{3k}^2 - 1) \\
  a_k(1,0) &= (p_{1k}^2 + p_{2k}^2 - p_{3k}^2 - 1) \\
  b_k(1,0) &= (x_{1k}^2 + x_{2k}^2 - x_{3k}^2 - 1) \\
  a_k(1,1) &= (p_{1k}^2 - p_{2k}^2 - p_{3k}^2 + 1) \\
  b_k(1,1) &= (x_{1k}^2 - x_{2k}^2 - x_{3k}^2 + 1)
\end{align*}
\]

Thus the first order filter $H_k(Z_1, Z_2)$ becomes a function of the parameter vector, given by:

\[
X_k = (p_{1k}, p_{2k}, p_{3k}, x_{1k}, x_{2k}, x_{3k})
\]

F.9

where $x_{1k}, x_{2k}$ and $x_{3k}$ are non-zero variables.

**Second Order Case:**

Consider an analog two dimensional filter, given by:

\[
H_{2k}(S_1, S_2) = \frac{p_{2k}^2(S_1, S_2)}{Q_{2k}(S_1, S_2)}
\]

\[
\begin{align*}
  Q_{2k}(S_1, S_2) &= U_{1k} + U_{2k} S_2 + U_{3k} S_2^2 + U_{4k} S_1 + U_{5k} S_1 S_2 + U_{6k} S_2^2 + U_{7k} S_1^2 + U_{8k} S_2^2 + S_1 S_2^2 \\
  V_{1k} + V_{2k} S_2 + V_{3k} S_2^2 + V_{4k} S_1 + V_{5k} S_1 S_2 + V_{6k} S_2^2 + V_{7k} S_1^2 + V_{8k} S_2^2 + S_1 S_2^2
\end{align*}
\]

F.10

In order that (F.10) satisfy (F.4), it is required that the coefficients of $Q_2(S_1, S_2)$ be expressed in terms of a set of
non-zero variables \((Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7, Y_8, Y_9, Y_{10})\) such that:

\[
\begin{align*}
v_{1_\ell} &= (y_{5_\ell} y_{10_\ell} - y_{6_\ell} y_{9_\ell} + y_{7_\ell} y_{8_\ell})^2 \\
v_{2_\ell} &= (y_{1_\ell} y_{8_\ell} - y_{2_\ell} y_{6_\ell} + y_{3_\ell} y_{5_\ell})^2 + (y_{1_\ell} y_{9_\ell} - y_{2_\ell} y_{7_\ell} + y_{4_\ell} y_{5_\ell})^2 \\
v_{3_\ell} &= y_{5_\ell}^2 \\
v_{4_\ell} &= (y_{1_\ell} y_{10_\ell} - y_{3_\ell} y_{7_\ell} + y_{4_\ell} y_{5_\ell})^2 + (y_{2_\ell} y_{10_\ell} - y_{3_\ell} y_{9_\ell} + y_{4_\ell} y_{8_\ell})^2 \\
v_{5_\ell} &= (y_{6_\ell}^2 + y_{7_\ell}^2 + y_{8_\ell}^2 + y_{9_\ell}^2) \\
v_{6_\ell} &= (y_{1_\ell}^2 + y_{2_\ell}^2) \\
v_{7_\ell} &= y_{10_\ell} \\
v_{8_\ell} &= y_{3_\ell}^2 + y_{4_\ell}^2
\end{align*}
\]

\[(F.11)\]

Using the double bilinear transform on \((F.10)\), the coefficients of the stable \(H_2(z_1, z_2)\) which is of the form given by \((F.3)\), can be written as:

\[
\begin{align*}
c_{\ell}(0, 0) &= (u_{1_\ell} + u_{2_\ell} + u_{3_\ell} + u_{4_\ell} + u_{5_\ell} + u_{6_\ell} + u_{7_\ell} + u_{8_\ell} + 1) \\
c_{\ell}(0, 1) &= 2(u_{1_\ell} - u_{3_\ell} + u_{4_\ell} - u_{6_\ell} + u_{7_\ell} - 1) \\
c_{\ell}(0, 2) &= (u_{1_\ell} - u_{2_\ell} + u_{3_\ell} + u_{4_\ell} - u_{5_\ell} + u_{6_\ell} + u_{7_\ell} - u_{8_\ell} + 1) \\
c_{\ell}(1, 0) &= 2(u_{1_\ell} + u_{2_\ell} - u_{3_\ell} - u_{7_\ell} - u_{8_\ell} - 1)
\end{align*}
\]

\[(F.12)\]

\[\text{(cont'd)}\]
\[ c_2(1,1) = 4(u_{1_2} - u_{3_2} - u_{7_2} + 1) \]

\[ c_2(1,2) = 2(u_{1_2} - u_{2_2} + u_{3_2} - u_{7_2} + u_{8_2} - 1) \]

\[ c_2(2,0) = (u_{1_2} + u_{2_2} + u_{3_2} - u_{4_2} - u_{5_2} - u_{6_2} + u_{7_2} + u_{8_2} + 1) \]

\[ c_2(2,1) = 2(u_{1_2} - u_{3_2} - u_{4_2} + u_{6_2} + u_{7_2} - 1) \]

\[ c_2(2,2) = (u_{1_2} - u_{2_2} + u_{3_2} + u_{4_2} + u_{5_2} - u_{6_2} + u_{7_2} - u_{8_2} + 1) \]

(F.12)

Similarly \( d_2(m,n) \) is expressed in terms of \( (v_{1_2}, v_{2_2}, \ldots, v_{8_2}) \) where \( (v_{1_2}, v_{2_2}, \ldots, v_{8_2}) \) are in turn expressed in terms of non-zero variables \( (y_{1_2}, y_{2_2}, \ldots, y_{10_2}) \), as given in (F.11). Thus the second order filter \( H_2(z_1, z_2) \) becomes a function of the parameter vector:

\[ X_{2_2} = (u_{1_2}, u_{2_2}, u_{3_2}, \ldots, u_{8_2}, y_{1_2}, y_{2_2}, y_{3_2}, \ldots, y_{10_2}) \]

(F.13)

where \( y_{1_2}, y_{2_2}, \ldots, y_{10_2} \) are non-zero variables.

B. Approximation Procedure

The procedure involves the minimization of an \( L^p \) norm of the form:

\[ L^p_{2_p}(X) = \frac{1}{p} \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \left( f(x, \Omega_{1i}, \Omega_{2j}) - f_d(\Omega_{1i}, \Omega_{2j}) \right)^2 \]

where \( X \) is a parameter vector with respect to which the minimization is performed. The minimization is carried out over a set of discrete frequency points \( \Omega_{1i} \) and \( \Omega_{2j} \) in the right half of the frequency domain such that:
$H_B(Z_1, Z_2)$ is chosen to be equal to that of $H_A(Z_1, Z_2)$.

**Case 3:** Atmospheric turbulence blur (2nd example):

$$f(X, \Omega_1, \Omega_2) = \left| H(X, Z_1, Z_2) \right|^2 \bigg|_{Z_1 = e^{-j\Omega_1}}^{Z_2 = e^{-j\Omega_2}} \quad (F.19)$$

where $X$ consists of parameter vectors $X_1$ and $X_2$ of each first and second filter sections respectively of $H(Z_1, Z_2)$.

The minimization of the $L_{\phi p}$ norm of (F.14) is performed by the widely used optimization procedure of Fletcher and Powell$^+$. 

\(-\pi < \Omega_{1i} < \pi \quad \text{and} \quad 0 < \Omega_{2j} < \pi \quad \text{(F.15)}\)

\(f_d(\Omega_{1i}, \Omega_{2j})\) is the desired specification. \(f(x, \Omega_{1i}, \Omega_{2j})\) is the approximating function whose form depends on the type of recursive filter implemented. The form of \(f(x, \Omega_{1i}, \Omega_{2j})\) for various types of implementations considered in this thesis, is as follows:

**Case 1:** Recursive filter implementation for Motion deblur:

\[
f(x, \Omega_1, \Omega_2) = 2 \cdot \text{Real} \{H(X, Z_1, Z_2)\} \quad \text{where} \quad Z_1 = e^{-j\Omega_1} \quad \text{and} \quad Z_2 = e^{-j\Omega_2} \quad \text{(F.16)}
\]

where \(H(Z_1, Z_2)\) is of the form given by (F.1). The parameter vector \(X\) consists of parameter vectors \(X_1\) and \(X_2\) of each first and second order filter sections respectively. \(X_1\) and \(X_2\) are as indicated in (F.9) and (F.13) respectively.

**Case 2:** Recursive filter implementation for Focus deblur:

\[
f(x, \Omega_1, \Omega_2) = |H_A(X_A, Z_1, Z_2)|^2 - |H_B(X_B, Z_1, Z_2)|^2 \quad \text{where} \quad Z_1 = e^{-j\Omega_1} \quad \text{and} \quad Z_2 = e^{-j\Omega_2} \quad \text{(F.17)}
\]

where parameter vector \(X\) is given by:

\[
X = \{X_A, X_B\} \quad \text{(F.18)}
\]

In (F.18), \(X_A\) of \(H_A(Z_1, Z_2)\) and \(X_B\) of \(H_B(Z_1, Z_2)\) each consist of parameter vectors \(X_{1k}\) and \(X_{2k}\) of each first and second order filter sections respectively. In (F.17), the order of
APPENDIX G

Image Processing Hardware
This appendix describes the image processing facility that was developed at the Signals and Systems Laboratory of the Department of Electrical Engineering, at the University of Windsor, which makes use of a Flying Spot Scanner System for image sampling and a Colorado Video Instrument display unit for image display. The block diagram of Figure G1 shows the complete hardware set-up. The facility is centred around a Data General Corporation NOVA-840 minicomputer with 128K, 16 bit words of core memory and moving head disk storage, with a total storage of 2.4 million 16-bit words. There is also an A/D and digital to analog (D/A) converters, with a general purpose high speed data channel interface.

**Flying Spot Scanner System**

The flying spot scanner system, used for sampling images, is shown in Figure G2. It consists of a cathode ray tube (CRT), objective lens, transmitting film, a condensing lens and photomultiplier tube. The x and y outputs from the digital to analog converter generates a raster scanned electron beam across the CRT. The resulting light pattern is focused by the objective lens, onto the transmitting film. The light that is transmitted by the film is received by a photomultiplier tube which is placed behind the film and the condenser lens system. This generates an equivalent electrical signal corresponding to the amount of light received.

In order to reduce the effects of internal and 60Hz beam supply noise, the output of the phototube is integrated over a time period before being sampled by the analog to digital converter.
Figure G1  Block Diagram of the Image Processing Facility.
The output of the analog digital converter is a digital number corresponding to the light that was transmitted through the film. It should be noted here that the sampled image does not exactly correspond to the original image that was recorded on the photographic film. The image being sampled is convolved with the CRT spot that scans across the film. The mathematical model that describes the light transmitted through the film is given by 38:

\[ g_1(x,y) = \int \int h_a(x-x_1, y-y_1) g(x_1, y_1) dx_1 dy_1 \] (G.1)

where \( g \) is the image recorded on the photographic film. \( g_j(j\Delta x, k\Delta y) \) are the matrix samples for \( j=0,1,\ldots,(N-1) \) and \( k=0,1,2,\ldots,(M-1) \). The function \( h_a \), describes the intensity profile of the CRT spot that is projected onto the film and this is the effective aperture through which the film is observed. The frequency domain equivalent of Equation (G.1) is given by:

\[ G_1(w_1, w_2) = H_a(w_1, w_2) \cdot G(w_1, w_2) \] (G.2)

where \( G_1 \) is the Fourier transform of \( g_1 \), etc. If the aperture projected on the film is infinitesimally small, i.e., a two-dimensional impulse, then \( G_1 = G \). In practice, such an aperture is impossible. From the viewpoint of aliasing+, such a response may be undesirable. Considering Equation (G.1), it can be seen that a suitable aperture can perform analog prefiltering and this may, in turn, reduce the problem.

+ Aliasing is the phenomenon in which the overlap of the Fourier spectrum takes place around the folding frequency due to incorrect choice of sampling interval.
of aliasing since the prefiltering is low pass in nature. This will also help to reduce the high frequency noise present in the image.

**Image Digitization Scheme**

As mentioned in the previous section, the output of the photomultiplier tube is integrated over a certain period of time. This period of time was chosen to be half the period of 60 Hz power supply (8 m.sec) since 60 Hz interference is present at the output of the photomultiplier tube.

With the flying spot scanner on, the timing and control circuit generates control pulses to control the integrator and A/D sampling. A block diagram of the timing and control circuit connections to the integrator is shown in Figure G3. The details of the timing and control circuit, along with the integrator circuit and the timing diagram can be found in the next section. The integrator is a precision integrator, with control inputs to allow reset, integrate and hold.

The entire image is sampled by scanning one line at a time, moving vertically downward after each line. This is carried out using a program 'PICSAMP' given in Appendix H. This program sets up two buffer storage areas in the memory. One buffer contains the x and y co-ordinate values for one line of the image to be scanned and the other is reserved for storing the sampled image values for the line that is being scanned.

The timing and control circuit sends out control signals soon after the power is turned on. The execution of the pro-
Figure G3  Timing and Control Circuit and Integrator SetUp.
gram then enables the D/A- and A/D to start scanning and sampling the image. The A/D waits for the operate pulse to go high (+5v), at which point the integration of the PMT output ends. After a small time delay, the A/D receives a sampling pulse via a specified trigger bus. The A/D converts the output of the integrator (which is held constant during the conversion) and stores the data in the buffer memory. Following the conversion, the A/D converter generates an end of conversion (EOC) pulse, which is sent to the D/A trigger bus. This generates a simultaneous D/A conversion of two data corresponding to the next x and y co-ordinates of the image. After a short time delay following the A/D sampling pulse, the timing and control circuit sends out a reset pulse to reset the output of the integrator to the appropriate initial condition.

Using the above procedure, images can be sampled with array sizes up to 512 x 512. The image scan can have its origin at any point on the image with different sampling grid sizes.

Details of Timing and Control Circuit and the Integrator Set Up

The main circuit is shown in Figure G4. The zero crossings of the 60 Hz supply are detected by two operational amplifiers (Motorola 741), operated in an open loop configuration. The outputs of OP1 and OP2 are square waves, with a period equal to that of the 60 Hz waveform, and are 180° out-of-phase with each other. These outputs are inputs to two monostables (74121) which are connected in such a way that they respond to only the positive transitions in the input. The outputs of
Figure G4 Timing and Control Circuit
Figure G5 Timing Diagram.
these monostables are the 'start' and 'stop' pulses. Through some extra circuitry, these pulses are then used to set or reset the 'operate flip-flop', which is made up of two nand gates; namely, 4 and 5 shown in Figure ... From the stop pulse, the A/D sampling trigger pulse is midway between the 'stop' and 'reset' pulse, thus ensuring correct sampling of the integrator output. The 'reset' pulse is generated by another network of delays as shown in Figure G4, and this pulse resets the integrator output to the desired initial condition (IC). The timing diagram corresponding to the timing and control circuit is shown in Figure G5.

The integrator shown in Figure G4 is a model 9018 precision integrator from Optical Electronics Inc. As indicated earlier (Chapter II), it has digital control inputs, enable or disable integration and reset the output to the given initial condition. As can be seen from the timing diagram, the integration takes place only when the OPERATE input is low (0 volts) and the RESET input is high (+5 volts). The output is reset when the OPERATE is high and RESET is low and is held constant when both are high. The output voltage swing of the integrator is 10V. This range of output is maintained by proper choice of capacitor Cx, the initial condition and proper input range.

The output of the photomultiplier tube is amplified prior to feeding it to the integrator. The amplifier circuit is shown in Figure G6. The output from the integrator is fed...
into the A/D converter through a buffer which is a model P201, FET operational amplifier.

**Image Display and Reproduction**

The display of sampled and processed images is carried out using the Colorado Video-270 A Instrument (CVI270A) Expander* which has Z modulation capability. The block diagram is shown in Figure G7.

In this display system, there exists a common interface with the NOVA computer, for both the D/A converter and the CVI270A Expander. In other words, the CVI270A and the D/A converter are considered as one peripheral device. The output latch for both the D/A converter and the CVI270A are the same. In order to transfer data to the CVI270A, the D/A converter has to be activated, and while the D/A conversion takes place, data is also transferred to the CVI270A.

The data transfer is accomplished via a handshaking operation between the D/A converter interface and the CVI270A. Considering Figure G7 - to display one pixel of the image requires the transfer of three data; namely, the x and y coordinates and the intensity value. The data is transferred one at a time. Thus for each pixel of the image, the D/A converter interface circuitry first presents the x data at the output latch, and sends a strobe pulse 'ACCEPT DATA', to the CVI270A, indicating that data is available, and the CVI270A in return sends back a pulse to D/A converter interface circuitry, indicating that it is ready for the next data.

---

* Polytron Devices Inc., P.O. Box 398, Paterson, N.J. 07524.
In a similar manner, y and Z (intensity) values are transferred.

Because of limited memory, the CVI270A can display images only up to size 256 x 256 pixels\. It can display intensities up to 256 gray levels, i.e., intensity data is 8 bits in length. The recording of the reproduced image on the T.V. monitor is carried out using a conventional camera, such as the Cannon FTb. For good results, one has to use a slow speed film, with exposure time set at 4th of a second.

\[+\] Pixel indicates a picture element.
APPENDIX H

Filter Design, Image Filtering and Other
Additional Image Processing Computer Programs
DIMENSION W(101), G(50), DM(101), DP(101), XCOF(50),
* COF(50), ROOTR(50), ROOTI(50), U(52, 52), DMA(101)

C
C MAIN PROGRAM - DESIGNING ONE-DIMENSIONAL RECURSIVE
C DIGITAL FILTER TO DESIRED MAGNITUDE AND LINEAR
C PHASE CHARACTERISTICS
C THE INPUT PARAMETERS ARE :-
C M1 : # OF FREQUENCY POINTS IN THE FREQUENCY AXI
C IFT : TYPE OF DESIRED FILTER
C LPF - LOW PASS ; BPF - BAND PASS
C HPF - HIGH PASS ; DIF - DIFFERENTIATOR
C WC1, WC2 : LOWER AND UPPER PASS BAND FREQUENCIES
C RESPECTIVELY, FOR THE CASE OF
C DIFFERENTIATOR, WC1 = 0.0 AND WC2 = 1.0
C ITrial : # OF TRIALS, WHERE FOR EACH TRIAL A FILTER
C OF DIFFERENT ORDER IS SPECIFIED
C ISETS : FOR EACH TRIAL VALUE OF ITrial, ISETS # OF
C GROUP DELAY VALUES ARE TRIED
C NORD : ORDER OF THE FILTER
C THE DIMENSIONS OF THE ARRAYS ARE AS FOLLOWS :-
C W(M1), G(NORD), DM(M1), DP(M1), XCOF(NORD+1), COF(NORD+1)
C ROOTR(NORD), ROOTI(NORD), U((2*NORD+4), (2*NORD+4)),
C DMA(M1)
C
C SUBROUTINES REQUIRED :- SPECNM, SIMPTB, KSIMP, PLOTMP,
C POLRT FOR OBTAINING POLES AND ZEROS OF THE FILTER
C
C
COMMON/B12, DM, DP, W, GDLV, WC1, WC2
COMMON/B30, NNC, NDC, NFRO
DATA FY, DIF/3. 141592, 'DIF '/
KTIME=0
READ(5, 60) M1
60 FORMAT(I1)
DW=1, / (M1-1)
W(1)=0,
DP(1)=0,
DO 123 I=2, M1
123 W(I)=W(I-1)+DW
READ(5, 655) IFT
655 FORMAT(A4)
READ(5, 201) WC1, WC2
201 FORMAT(2E12.5)
WRITE(6, 511)
511 FORMAT(' ', 2A, 'DESIGN SPECIFICATIONS' = (FREQUENCIES
* ARE IN FRACTIONS OF NYQUIST RATE), ')
CALL SPECNM(M1, IFT)
NFRO=M1
READ(5, 515) ISETS, ITrial
515 FORMAT(2I2)
40 FORMAT(I2)
   WRITE(6,50) M1,DM
50 FORMAT(' ',2X,'NO. OF POINTS ON FREQ AXIS= ',I2,5X,'FRE
*O INTERVAL= ',E12.5,1X)
   WRITE(6,512)
512 FORMAT(' ',4X,'FREQUENCY',3X,'MAGNITUDE',1X)
DO 444 I=1,M1
   DMA(I)=DM(I)
444 WRITE(6,20) W(I),DM(I)
20 FORMAT(' ',2(2X,F10.6))
   CALL PLOT3(W,DMA,M1)
   DIFPH=0.5*PY
   DO 2000 ITR=1,NTRIAL
      READ(5,40) NORD
      NNC=NCMD+1
      NDC=NORD
      GDLY=NORD
      WRITE(6,52) NORD
52 FORMAT(' ',2X,'ORDER OF THE FILTER:------ ',I2,1X)
   DO 1000 KTIMES=1,INETS
   C
   GENERATION OF LINEAR PHASE CHARACTERISTICS
   DO 2 I=2,M1
      DP(I)=-GDLY*W(I)*1
      IF(ITT.NE.DIF) GO TO 505
      DP(I)=DP(I)+DIFPH*0.5*W(I)*1
   505 CONTINUE
      IF(DP(I),LT,-1) DP(I)=DP(I)+2.0*1
      IF(DP(I),LT,-1) GO TO 505
   2 CONTINUE
   WRITE(6,513) KTIMES,GDLY
513 FORMAT(' ',2X,'SET NUMBER= ',I2,2X,'GROUP-DELAY=',
         *E12.5,1X)
   CALL SIMPTB(M1,NCR,NVR)
   CALL RSIMP(U,NCR,NVR,KSIM)
   IF(KSIM.EQ.1) GO TO 10002
   NRS=NCMD+2
   DO 70 I=1,NRC
70   G(I)=-U(NCMD+1,I)
   DO 71 I=1,NNC
71   G(NDC+I)+=-U(NCMD+1,NDC+I)
   IPK=1
   IFK=NCMD+1
   COE=1.
   WRITE(6,900) IPK,G(IPK),IPK,COE
   DO 901 IK=1,NDC
      IK2=IK+1
      IK2=IK+1
   900 FORMAT(' ',2X,'COEFN=',12,1X)= 014.7,5X,'COEFD=',12,1X)
901 WRITE(6,500)
DO 902 I=1,NNC
902 XCOF(I)=G(NDC+I)
      NNC=NORD
      NNCP=NNC
      IF(NNC.EQ.0) GO TO 910
      CALL POLRT(XCOF,COF,MNC,ROOTR,ROOTI,IER,NNCP)
      WRITE(6,800)
      WRITE(6,903) IER
      FORMAT(' ',2X,'IER=',I1,/',')
      GO TO 941
910 ROOTR(I)=XCQF(I)
      ROOTI(I)=0.
      MNC=1
911 WRITE(6,800)
      WRITE(6,904)
904 FORMAT( ' ',2X,'THE ZEROS OF THE FILTER ARE=','/',)
      WRITE(6,905) (I,ROOTR(I),ROOTI(I),I=I1,NNC)
905 FORMAT( ' ',2X,'ZERO','I2=','E14.7','+J 'E14.7,/',)
      XCQF(I)=1.
      DO 906 I=1,NDC
906 XCQF(I+1)=G(I)
      NDCP=NDC
      CALL POLRT(XCOF,COF,NDCP,ROOTR,ROOTI,IER,NNCP)
      WRITE(6,800)
      WRITE(6,903) IER
      WRITE(6,800)
      WRITE(6,907)
907 FORMAT( ' ',2X,'THE POLES OF THE FILTER ARE=','/',)
      WRITE(6,906) (I,ROOTR(I),ROOTI(I),I=I1,NDC)
906 FORMAT( ' ',2X,'POLE','I2=','E14.7','+J 'E14.7,/',)
      KST=0
      DO 900 I=1,NDC
      STB=ROOTR(I)**2+ROOTI(I)**2
      800 IF(STB.LE.1.) KST=1
      IF(KST.EQ.0) WRITE(6,601)
      IF(KST.EQ.1) WRITE(6,602)
      WRITE(6,800)
      DO 912 I=1,NNC
912 COF(I)=G(NDC+I)
      KTIME=KTIME+1
      CALL PLOTMP(M1,COF,XCOF,NNCP,NDCP,KTIME)
10002 CONTINUE
      GDLY=GDLY+1
1000 CONTINUE
2000 CONTINUE
908 FORMAT( ' ',//)
501 FORMAT( ' ',2X,'THE FILTER IS STABLE',//)
502 FORMAT( ' ',2X,'THE FILTER IS UNSTABLE',//)
STOP
END
SUBROUTINE SPEC(M, IFT)

SUBROUTINE TO GENERATE THE MAGNITUDE SPECTRUM CORRESPONDING TO THE DESIRED TYPE OF FILTER. THE CHARACTERISTICS BEYOND THE PASS BAND IS GENERATED ACCORDING TO THE FUNCTION

\[ F_N = \exp(-F_k^2(W_p-W_A)^2) \]

WHERE \( F_k \) CAN BE VARIED. \( F_k \) IS OBTAINED IN TERMS OF \( F_k^2 \).

SHARPER TRANSITION BAND IN THE SPECIFICATIONS CAN BE GENERATED BY SPECIFYING SMALLER VALUES FOR \( F_k^2 \), USUALLY LESS THAN 1.0

DIMENSION DM(101), DP(101), W(101)
INTEGER BP, HP, DIF
COMPLEX*16 TP1, TP2, TP3, TP4, TP5
COMMON/B12/DM, DP, W, GDL, WP1, WP2
DATA TF1, 'LOW PASS FILTER'
DATA TF2, 'BAND PASS FILTER'
DATA TF3, 'HIGH PASS FILTER'
DATA TF4, 'DIFFERENTIATOR'
FN(Fk, WP, Wp) = \exp(-F_k^2(W_p-W_A)^2)
IF(IF, EQ, DIF) GO TO 103
Fk2 = 0.7
Fk = 100.0 \times (Fk2 + Fk2)
IF(IF, EQ, BP) GO TO 100
IF(IF, EQ, HP) GO TO 101
TP5 = TP1
DO 1 I = 1, M1
IF(W(I), LT, WP2) GO TO 2
DM(I) = 1.0
GO TO 1
2 DM(I) = FN(FK, WP2, W(I))
1 CONTINUE
GO TO 102
100 TP5 = TP2
DO 3 I = 1, M1
IF(W(I), LT, WP1) GO TO 4
IF(W(I), LT, WP2) GO TO 5
DM(I) = 1.0
GO TO 3
4 DM(I) = FN(FK, WP1, W(I))
GO TO 3
5 DM(I) = FN(FK, WP2, W(I))
3 CONTINUE
GO TO 102
101 TP5 = TP3
DO 6 I = 1, M1
IF(W(I), LT, WP1) GO TO 7
DM(I) = 1.0
GO TO 6
7 DM(I) = FN(FK, WP1, W(I))
6 CONTINUE
GO TO 102
103 TP5=TP4
   DO 8 I=1, M1
   8 DM(I)=W(1:)
102 WRITE(6,202) TP5
   WRITE(6,203) WPI, WP2
202 FORMAT(2X,'FILTER TYPE :---', A16, /)
203 FORMAT(2X,'PASS BAND FREQUENCIES ++++', 2X, 'WP1=', E12.5, 2X, 'WP2=', E12.5, /)
RETURN
END

SUBROUTINE SIMPTB(M1, NTV, NVR)
C
C SUBROUTINE TO GENERATE THE SIMPLEX TABLEAU.
C THE DIMENSIONS OF THE ARRAYS ARE AS FOLLOWS
C A(2*NORD+1, 2+M1), B(2*NORD+4), R1(2, M1)
C R2(2, 2+M1), CT(NORD, M1)
C NORD AND M1 ARE AS INDICATED IN THE
C MAIN PROGRAM.
C
DIMENSION A(49, 202), B(52), R1(2, 101), R2(2, 202),
   CT(24, 101), W(101), DP(101), DM(101)
COMMON/B10/, NNC, NDC, MFREQ
COMMON/B11/, M2, DW2
COMMON/B12/, DM, DP, W, GDLY, WC1, WC2
COMMON/B24/, R1, R2, CT, A, B
PY=3.1415926
NTA=NDC+NNC
NTV=NTA+1
NTVI=NTA+3
K1=M1
KK1=K1+K1
KK2=KK1+K1
KK3=KK2+K1
KK4=KK3+M2
ICT=1
NVR=KK4
IF (ICT. EQ. 0) NVR=KK3
EPS=1. E-6
DO 1 I=1, M1
YR=DM(I)*COS(DP(I))
YI=DM(I)*SIN(DP(I))
SUM1=0.
SUM2=0.
SUM3=0.
SUM4=0.
XAI=PY*W(I)
DO 2 J=1, NDC
XX=COS(J*XAI)
YY=SIN(J*XAI)
A(J, I)=YR*XX+YI*YY
A(J, K1+I)=YI*XX-YR*YY
SUM1=SUM1+A(J, I)
SUM2=SUM2+A(J, K1+I)
2 CONTINUE
DO 3 K=1, NNC
XK=COS(K*XAI)
YK=SIN(K*XAI)
A(NDC+K, I)=-XX
A(NDC+K, K1+I)=YK
SUM1=SUM1+A(NDC+K, I)
SUM2=SUM2+A(NDC+K, K1+I)
3 CONTINUE
R2(1, I)=-YR
R2(1, K1+I)=-YI
R2(2, I)=-SUM1-1.
R2(2, K1+I)=-SUM2-1.
1 CONTINUE
DO 5 I=1, NTA
5 B1(I)=0.
B2(NTV)=1.
B2(NTV+1)=0.
B2(NTV1)=-1.
DO 16 I=1, M1
SUM1=0.
XAI=PY*W(I)
DO 17 J=1, NDC
XX=COS(J*XAI)
CT(J, I)=-XX
17 SUM1=SUM1+CT(J, I)
R1(1, I)=1. -EPS
R1(2, I)=-SUM1
RETURN
END
SUBROUTINE RSIMP(U, NCR, NVR, KSIM)

SUBROUTINE TO SOLVE THE LINEAR PROGRAMMING
PROBLEM USING THE REVISED SIMPLEX METHOD
THIS PROGRAM USES THE SIMPLEX TABLEAU
GENERATED IN THE ROUTINE SIMPTB.
X, E, AX ARE WORK VECTORS. THEIR SIZES
ARE AS FOLLOWS.
X(2*NORD+4), E(2*NORD+4), AX(2*K1)

DIMENSION A(49, 202), B(52), U(52, 52), X(52), E(52),
R1(2, 101), R2(2, 202), CT(24, 101), AX(202)
COMMON/B30, NNC, NDC, M1
COMMON/B35/R, R2, CT, A, B
REAL*8 X1, Y1, X2
REAL*8 XLM
KSIM=0
K1=ML+M1
K2=K1+K1
NCR1=NCR-1
NEQ=NCR+2
IKT=2
SF=2.0
DO 2 I=1, NEQ
DO 2 J=1, NEQ
2 U(I, J)=0.0
DO 4 I=1, NEQ
4 U(I, I)=1.0
ITR=1
EMU=-1.0, E=0.4
EMV=-EMU
IX=0
XLM=1.0, D=75
24 XM=1.0, E=75
IOM=0
DO 5 I=1, NVR
5 SUM=0.
IF(I.GT.K1) GO TO 100
DO 6 J=1, NCR1
IF(U(NEQ, J), EQ, 0, 0) GO TO 6
X1=U(NEQ, J)
Y1=AK(J, I)
X2=X1*Y1
IF(X2.GE.XLM) IOM=IOM+1
IF(IOM.GE.5) GO TO 37
X11=SNGL(X2)
SUM=SUM+X11
6 CONTINUE
AX(I)=SUM
DELJ=SUM+U(NEQ, NCR)+R2(1KT, I)
GO TO 102
100 IF(I.GT.KK2) GO TO 101
DELJ=-AX(I-KK1)+U(NEQ, NCR)-(R2(1KT, I-KK1)+SF)
GO TO 102
101 IKK2=I-KK2
DO 66 J=1, NDC
IF(U(NEQ, J).EQ.0.0) GO TO 66
SUM=SUM+U(NEQ, J)*CT(J, IKK2)
66 CONTINUE
DELI=SUM+R1(IKT, IKK2)
102 IF(XMIN. LE. DELJ) GO TO 5
XMIN=DELI
KK=I
5 CONTINUE
IF(X(1, EQ.0) GO TO 21
IF(XMIN. GE. -1E-04) GO TO 22
21 IF(KK. GT. KK2) GO TO 200
IF(KK. GT. KK1) GO TO 201
KKCOL=KK
GO TO 202
201 KKCOL=KK-KK1
202 DO 7 I=1, NCR
X(I)=0.0
DO 8 J=1, NCR1
IF(U(I, J).EQ.0.0) GO TO 8
X(I)=X(I)+U(I, J)*R(J, KKCOL)
8 CONTINUE
IF(KK. GT. KK1) X(I)=-X(I)
7 X(I)=X(I)+U(I, NCR)
DO 77 L2=1, IKT
L21=NCR+L2
X(LZ1)=0
DO 88 J2=1, NCR1
X(LZ1)=X(LZ1)+U(LZ1, J2)*R(J2, KKCOL)
88 CONTINUE
IF(KK. GT. KK1) X(LZ1)=-X(LZ1)+U(LZ1, NCR)+R2(LZ, KKCOL)
GO TO 77
301 IF(LZ. EQ. 1) X(LZ1)=-X(LZ1)+U(LZ1, NCR)-R2(LZ, KKCOL)
IF(LZ. EQ. 2) X(LZ1)=-X(LZ1)+U(LZ1, NCR)-(R2(LZ, KKCOL)+SF)
77 CONTINUE
GO TO 203
200 KKCOL=KK-KK2
KNT=0
DO 80 I=1, NEQ
X(I)=0.0
DO 81 J=1, NDC
IF(U(I, J).EQ.0.0) GO TO 81
X(I)=X(I)+U(I, J)*CT(J, KKCOL)
81 CONTINUE
IF(I. LE. NCR) GO TO 80
KNT=KNT+1
X(I)=X(I)+R1(KNT, KKCOL)
80 CONTINUE
202 XMIN=1. E79
KKJ=0
DO 10 I=1, NCR
IF \( X(I) \leq 0 \) GO TO 10

\( \text{THETA} = B(I)/X(I) \)

IF \( X \text{MIN} < \text{THETA} \) GO TO 10

\( X \text{MIN} = \text{THETA} \)

KKJ = I

10 CONTINUE

IF \( \text{KKJ} = 0 \) GO TO 35

DO 12 I = 1, N

12 \( E(I) = -(X(I)/X(KKJ)) \)

\( E(KKJ) = 1/X(KKJ) \)

\( Y_1 = B(KKJ) \)

DO 14 I = 1, N

14 \( B(I) = B(I) + E(I)*Y_1 \)

\( B(KKJ) = E(KKJ)*Y_1 \)

DO 15 I = 1, N

15 \( Y_1 = U(KKJ, I) \)

DO 16 J = 1, N

16 \( U(KKJ, I) = U(J, I) + E(J)*Y_1 \)

ITR = ITR + 1

IF \( X(I) = 0 \) GO TO 19

IF \( B(N) \leq \text{EMU} \) AND \( B(N) > \text{EMV} \) GO TO 18

IF \( B(N) \geq \text{EMV} \) GO TO 20

GO TO 19

IX = 1

NEO = NEO - 1

IKT = IKT - 1

SF = 0.

19 GO TO 24

25 PRINT 36

36 FORMAT(' \( \text{KSI} = 1 \) \)

GO TO 22

20 PRINT 41

41 FORMAT(' \( \text{KSI} = 1 \) \)

WRITE(6, 505) ITR, B(N)

505 FORMAT(' \( \text{KSI} = 1, \text{PHASE} = \), \( \text{NO. OF ITR} = \), \( \text{OBJ. FN. VAL} = \) ', E14.7)

GO TO 32

37 WRITE(6, 52)

52 FORMAT(' \( \text{KSI} = 1 \) \)

GO TO 32

22 WRITE(6, 42) ITR

42 FORMAT(' \( \text{OPTIMAL SOLUTION FOUND...} \) \)

WRITE(6, 72) B(NCR + 1)

72 FORMAT(' \( \text{THE OPTIMUM VALUE OF THE OBJECTIVE} \) \)

RETURN

END
SUBROUTINE PLOTMP(M1, COEF1, COEF2, NNC, NDC, KTIMES)

SUBROUTINE TO COMPUTE, PRINT AND PLOT OUT THE
MAGNITUDE AND GROUP-DELAY CHARACTERISTICS OF THE
DESIGNED FILTER. THIS ALSO COMPUTES THE
SQUARED ERROR BETWEEN THE DESIRED AND DESIGNED
CHARACTERISTICS. H IS THE ARRAY TO STORE
COMPUTED MAGNITUDE AND P IS THE ARRAY TO STORE
THE GROUP-DELAY VALUES.
THE SIZES OF ARRAYS LOCAL TO THIS ROUTINE ARE
Z(M1), H(M1), P(M1)
M1 IS AS INDICATED IN THE MAIN PROGRAM

DIMENSION H(101), P(101), COEF1(50), COEF2(50), W(101),
Z(101), DP(101), DM(101)
COMPLEX Z, H, HD, HZ, CMPLX, ZZ, CEXP
COMPLEX HN1, HD1
COMMON/B12/DM, DP, H, GDLY, WP1, WP2
DATA PY/3. 141592/
ERM=0.
ERG=0.
WRITE(6,88)
88 FORMAT(1X)
WRITE(6, 87)
87 FORMAT(1X, 5X, 'FREQUENCY', 4X, 'MAGNITUDE', 4X,
'GROUP-DELAY', 1X)
DO 4 I=1, M1
IF(KTIMES.GE.2) GO TO 500
Z(I)=CMPLX(0.0, -W(I)*PY)
Z(I)=CEXP(ZZ)
500 HN1=(0.0, 0.0)
HD1=(0.0, 0.0)
HN=(0.0, 0.0)
HD=(0.0, 0.0)
DO 5 J=1, NNC
HN=HN+COEF1(J)*Z(I)**(J-1)
5 HN1=HN1+(J-1)*COEF1(J)*Z(I)**(J-1)
DO 6 J=1, NDC
HD=HD+COEF2(J)*Z(I)**(J-1)
6 HD1=HD1+(J-1)*COEF2(J)*Z(I)**(J-1)
HZ=HN/HD
H(I)=CABS(HZ)
P(I)=REAL((HN1/HN)-(HD1/HD))
4 WRITE(6,82) W(I), H(I), P(I)
82 FORMAT(1X,3(2X,E12.5))
MT=M1
KB=0
DO 7 I=1, MT
KB=KB+1
ERM=ERM+(DM(I)-H(KB))**2
IF(W(KB).LT.WP1, OR, W(KB).GT.WP2) GO TO 7
ERG=ERG+(GDLY-P(KB))**2
7 CONTINUE
WRITE(6,90) ERM, ERG
90 FORMAT(1X, 'SUM OF SQUARED ERROR IN MAGNITUDE IS=', E12.5)
WRITE(6,50) ML, DW
CALL PLOTS(W, H, M1)
WRITE(6,60) ML, DW
CALL PLOTS(W, P, M1)
50 FORMAT(1H1, 'DESIGNED MAGNITUDE RESPONSE', 2X, 'NO. OF
* POINTS ON FREQ AXIS=', 12, 2X, 'FREQ INTERVAL=', E12.5)
60 FORMAT(1H1, 'DESIGNED GROUP DELAY RESPONSE', 2X, 'NO. OF
* POINTS ON FR AXIS=', 12, 2X, 'FREQ INTERVAL=', E12.5)
RETURN
END
DIMENSION DM(33,17),DP(33,17),W1(33),W2(17),
  *AC(6,6),DC(6,6),U(34,34)

MAIN PROGRAM - DESIGNING TWO-DIMENSIONAL RECURSIVE
DIGITAL FILTER TO DESIRED MAGNITUDE AND LINEAR
PHASE CHARACTERISTICS.
THE INPUT PARAMETERS ARE :
M1 : # OF FREQUENCY POINTS ALONG THE W1-AXIS
N1 : # OF FREQUENCY POINTS ALONG THE W2-AXIS
IFT : DESIRED FILTER TYPE
  LPF : LOW PASS ; BPF : BAND PASS
  HPF : HIGH PASS
WP1 , WP2 : LOWER AND UPPER RADIAL PASS BAND
  FREQUENCIES
ISETS : # OF GROUP DELAY VALUES TO BE TRIED
NP : HIGHEST POWER OF COMPLEX FREQUENCY VARIABLE Z1
    IN THE NUMERATOR OF THE TRANSFER FUNCTION
NQ : HIGHEST POWER OF COMPLEX FREQUENCY VARIABLE Z2
    IN THE NUMERATOR OF THE TRANSFER FUNCTION
SIMILARLY NP & NQ ARE FOR THE DENOMINATOR OF THE
DIGITAL FILTER TRANSFER FUNCTION
LET - IT=NP+1+1+NP+1+1+1-1
THEN THE DIMENSIONS OF THE ARRAYS ARE ASFOLLOWS :
W1(M1), W2(N1), DM(NM1, N1), DP(N1, N1), AC(NP, NQ),
BC(NP, NQ), U(IN1+1, IT+1)

COMMON/BS,DM, DP, W1, W2, WP1, WP2
COMMON/BT, MP1, MQ, MP, MQ, NN, ND, MNU
PY=3.1415926
PY2=PY+PY
READ(5,800) M1,N1
DW1=2./M1-1
DW2=1./N1-1
WRITE(6,908)
WRITE(6,909)
WRITE(6,910) M1,N1
WRITE(6,911) DW1,DW2
WRITE(6,901)
W1=1
DO 1 I=2,N1
1 W1(I)=W1(I-1)+DW1
W2(I)=0.
DO 2 I=2,N1
2 W2(I)=W2(I-1)+DW2
KTIME=1
READ(5,601) IFT
READ(5,604) WP1, WP2
501 FORMAT(A4)
604 FORMAT (2E12.5)
        CALL SPECM (ML, NL, IFT)
        READ (5, 503) ISETS
        READ (5, 301) NP, NQ, MP, MQ
        MP1=MP+1
        MQ1=MQ+1
        NP1=NP+1
        NQ1=NQ+1
        WRITE (6, 304) NP, NQ
        WRITE (6, 307) MP, MQ
        GDLV2=MQ
        GDLV1=MP
        WRITE (6, 302)
        CALL OUTPUT (DM, ML, NL)
        WRITE (6, 301)
        DO 100 KT I M S = 1, ISETS
        WRITE (6, 312) KT, PY
        WRITE (6, 313) GDLV1, GDLV2

C

C GENERATION OF LINEAR PHASE CHARACTERISTICS

C

DO 6 I=1, ML
        PF= GDLV1 + ML(I)*PY
        DO 6 J=1, NL
                DP(I, J)=-(PF+GDLV2+ML(J)*PY)
405 IF(DP(I, J).LT.-PY) DP(I, J)=DP(I, J)+PY2
        IF(DP(I, J).LT.-PY) GO TO 505
406 IF(DP(I, J).GT.PY) DP(I, J)=DP(I, J)-PY2
        IF(DP(I, J).GT.PY) GO TO 506

6 CONTINUE
        CALL SNIPTB (ML, NL, NCR, NVR)
        CALL RSIMP (U, NCR, NVR, KSIM)
        IF(KSIM.EQ.1) GO TO 1000
        NT=NN+ND
        NA=NT
        NT1=NA+1
        NT2=NA+2
        NT3=NA+3
        BC(1, 1)=1.
        KM=0
        DO 10 I=1, MP1
                KI=1
                IF(I.EQ.1) KI=2
                DO 10 J=KI, MQ1
                        KM=KM+1
        10 BC(I, J)=-U(NT1, KM)
        KM=NT+ND
        L=0
        DO 11 I=1, NP1
        11 RC(I, J)=-U(NT2, ND+L)
        WRITE (6, 301)
        WRITE (6, 300) (U(NT2, 1), I=1, NT3)
WRITE(6,901)
WRITE(6,905)
DO 12 I=1, MP1
DO 12 J=1, NG1
IF(I.GT.NP1.OR.J.GT.NG1) AC(I,J)=0.
12 WRITE(6,906) I,J,AC(I,J),I,J,BC(I,J),
CALL PLOTMP(MP1,ML,AC,BC,KTIME,DLYLY1,DLYLY2),
KTIME=KTIME+1
1000 CONTINUE
WRITE(6,901)
DLYLY1=DLYLY1-1
DLYLY2=DLYLY2-1
300 CONTINUE
800 FORMAT(2I2)
801 FORMAT(4I2)
802 FORMAT(I2)
900 FORMAT(2X,11(F10.4,1X))
901 FORMAT(2X,1X)
902 FORMAT(2X,2X,'DESIRED MAGNITUDE SPECIFICATION
*ONS',/)
904 FORMAT(2X,2X,'NUMERATOR INDICES P=',12/2X,'&
* Q=',12,/)
905 FORMAT(' ',2X,'THE COEFFICIENTS OF THE FILTER
*ARE:',-8X)'/
906 FORMAT(2X,2X,'G15.8',12,'/12.12','=','G15.8',/)
907 FORMAT(2X,2X,'DENOMINATOR INDICES P=',12/2X,'&
* Q=',12,/)
908 FORMAT(2X,2X,'DESIRED SPECS (FREQUENCIES ARE
*IN FRACTIONS OF NYQUIST RATE) ----',/)
910 FORMAT(' ',2X,'TOTAL NO. OF SPECS-POINTS ALONG W1-A
*XS=',12.2X,'TOTAL NO. OF SPECS-POINTS ALONG W2-AXI)
*S=',12,/)
911 FORMAT(2X,2X,'FREQ-INTERVAL ALONG W1-AXIS=',
*E12.5,'FREQ-INTERVAL ALONG W2-AXIS=',E12.5
*,',/)
912 FORMAT(2X,2X,'SET NO.=',12,/)
913 FORMAT(2X,2X,'GROUP-DELAY ALONG W1-AXIS=',E1
*2.5,5X,'GROUP-DELAY ALONG W2-AXIS=',E12.5',/)
STOP
END
SUBROUTINE SPECML, NL, IFT
C
SUBROUTINE TO GENERATE THE MAGNITUDE SPECs
CORRESPONDING TO THE DESIRED TYPE OF FILTER
THE CHARACTERISTICS BEYOND THE PASS BAND IS
GENERATED ACCORDING TO THE FUNCTION
FN=EXP(-FK**2(WP-WX)**2) WHERE FK CAN BE
VARIED. FK IS OBTAINED IN TERMS OF FK2.
SHARPER TRANSITION BAND IN THE SPECIFICATIONS
CAN BE GENERATED BY SPECIFYING SMALLER VALUES
FOR FK2. THE ARRAY PB(NL,N1) IS A LOGICAL
ARRAY WHICH IS TRUE IN THE PASS BAND AREA OF
DESIGNED SPECIFICATION
C
FUNCTION SUBPROGRAM REQUIRED : FN
C
DIMENSION DM(33,17), RP(33,17), W1(23), W2(17),
*PB(33,17)
COMPLEX*16 TP1, TP2, TP3, TP4
LOGICAL*1 A%, TRUE, /B%, FALSE, /PB
INTEGER BP, HP
COMMON/B8/DM, RP, W1, W2, WP1, WP2
COMMON/B30/PB
DATA TP1/'LOW PASS FILTER'/
DATA TP2/'BAND PASS FILTER'/
DATA TP3/'HIGH PASS FILTER'/
DATA BP,'BPF' /, HP,'HPF' , RY/3.141592/
DO 15 I=1, M1
DO 15 J=1, N1
15 PB(I, J)=B%
FK2=2.0
FK=100. /(FK2**2)
DO 1 I=1, M1
R1=W1(I)**W1(I)
DO 1 J=1, N1
1 RP(I, J)=SORT(R1+W2(J)**W2(J))
IF(IF, EQ, BP) GO TO 100
IF(IF, EQ, HP) GO TO 101
TP4=TP1
DO 2 I=1, M1
DO 2 J=1, N1
IF(RP(I, J), GT, WP2) GO TO 3
PB(I, J)=A%
DM(I, J)=1.0
GO TO 2
3 DM(I, J)=FN FK, WP2, RP(I, J))
2 CONTINUE
GO TO 102
100 TP4=TP2
DO 4 I=1, M1
DO 4 J=1, N1
IF(RP(I, J), LT, WP1) GO TO 5
1F(RP(I, J), GT, WP2) GO TO 5
PB(I, J) = RN
DM(I, J) = 1.0
GO TO 4
5 DM(I, J) = FN(FK, WP1, RP(I, J))
GO TO 4
6 DM(I, J) = FN(FK, WP2, RP(I, J))
4 CONTINUE
GO TO 102
101 TP4 = TP3
DO 7 I = 1, M1
DO 7 J = 1, N1
IF(RP(I, J) .GE. WP1) GO TO 8
DM(I, J) = FN(FK, WP1, RP(I, J))
GO TO 7
7 PB(I, J) = RN
DM(I, J) = 1.0
7 CONTINUE
102 CONTINUE
WRITE(*, 500) TP4
WRITE(*, 591) WP1, WP2
500 FORMAT(1X, 'FILTER TYPE : --- ', A16, 1X)
591 FORMAT(1X, 'PASS BAND FREQUENCIES : --- WP1 = 
* ', E12.5, 2X, 'WP2 = ', E12.5, 1X)
RETURN
END
FUNCTION FN(FK, WP, W)
FUNCTION SUBPROGRAM REQUIRED BY THE SUBROUTINE SPEC
ARG = FK + (WP-W) * HP
FN = 0.0
IF(ARG .GT. 157.0) GO TO 501
FN = EXP(-ARG)
501 RETURN
END
SUBROUTINE SIMTEB(N1, M1, MP, NO, NP, NG, NR, NV)
SUBROUTINE TO GENERATE THE SIMPLEX TABLEAU
IT, M1, N1 : IS AS DEFINED IN THE MAIN PROGRAM
LET ITX = MP1 + NO1 - 1 & ITY = M1 + N1
THE DIMENSIONS OF THE ARRAYS ARE AS FOLLOWS
A(IT, 2+ITY), B(IT+I), C(T1+ITY), R1(2, I1+ITY)
R2(2, 2+ITY)
DIMENSION A(21, 112), B(34), DM(22, 17), DK(13, 17),
W1(33), W2(17), C(15, 561), R1(2, 561), R2(2, 112)
DIMENSION YR(13, 17), Y1(33, 17)
COMMON/E8/DM, WP, MP, W1, W2, WP1, WP2
COMMON/E7/MP1, MO1, NP1, NO1, NN, ND, IC
COMMON/E5/R, B, CT, R1, R2
Y = 3.1415926
ICT = 1
\[ ND = MP1 + MG1 - 1 \]
\[ IC = M1 + N1 \]
\[ NN = NP1 + MG1 \]
\[ IC1 = IC + IC \]
\[ IB2 = ND + NN \]
\[ IC2 = IC1 + IC \]
\[ IC4 = IC2 + IC \]
\[ IC5 = IC4 + IC \]
\[ EPS = 1.6 \times 0.03 \]
\[ IL = 0 \]
\[ IL1 = IC \]
\[ IL2 = IC4 \]
\[ DO 1 \ L = 1, M1 \]
\[ IK = 0 \]
\[ DO 2 \ K = 1, MP1 \]
\[ IK1 = K - 1 \]
\[ IMG1 = MG1 \]
\[ IF(K, EQ, 1) IMG1 = MG1 - 1 \]
\[ DO 3 \ J = 1, N1 \]
\[ IM = 0 \]
\[ IF(K, EQ, 1) IM = 1 \]
\[ DO 3 \ I = 1, IMG1 \]
\[ XX = (IK1 + IL1(L) + IM * W2(J)) * FY \]
\[ AX2 = \cos\langle XX \rangle \]
\[ AY2 = \sin\langle XX \rangle \]
\[ AY1 = \sin\langle DP(L, J) \rangle \]
\[ AM1 = \cos\langle DP(L, J) \rangle \]
\[ A(K+I, IL1+J) = DM(L, J) * (AY1 * AM1 - AX2 * AY2) \]
\[ A(K+I, IL+J) = DM(L, J) * (AX1 * AY2 - AY1 * AX2) \]
\[ CT(IK+I, IL+J) = -\cos\langle XX \rangle \]
\[ IM = IM + 1 \]
\[ IK = IK + IMG1 \]
\[ IF(K, EQ, 1) IK = IK - 1 \]
\[ IL = IL + N1 \]
\[ IL1 = IL1 + N1 \]
\[ 1 \] IL2 = IL2 + N1
\[ IL = 0 \]
\[ IL1 = IC \]
\[ IL2 = IC4 \]
\[ DO 6 \ L = 1, M1 \]
\[ IK = ND \]
\[ DO 7 \ K = 1, MP1 \]
\[ IK1 = K - 1 \]
\[ DO 7 \ J = 1, N1 \]
\[ DO 7 \ I = 1, MG1 \]
\[ IM = IM + 1 \]
\[ XX = (IK1 + IL1(L) + IM * W2(J)) * FY \]
\[ A(K+I, IL+J) = -\cos\langle XX \rangle \]
\[ A(K+I, IL1+J) = \sin\langle XX \rangle \]
\[ IK = IK + MG1 \]
\[ IL = IL + N1 \]
\[ IL1 = IL1 + N1 \]
\[ 5 \] IL2 = IL2 + N1
\[ IN1 = IS2 + 1 \]
IN2=IB2+2
IN3=IB2+3
EPS1=1. -EPS
IL=0
DO 12 I=1, M1
   IL1=IL+IC
   DO 14 J=1, N1
      R2(I, IL1+J)=-DM(I, J)*COS(DP(I, J))
   14 R2(I, IL1+J)=-DM(I, J)*SINK(DP(I, J))
   12 IL=IL+N1
   DO 15 I=1, IB2
   15 B(I)=0.
   B(IN1)=1.
   B(IN2)=0.
   B(IN3)=-1.
   DO 20 I1=1, IC1
      SUM=0. 0
      DO 21 I2=1, IB2
      SUM=SUM+A(I2, 11)
      20 R2(2, I1)=-(SUM+1. 0)
      DO 22 I1=1, IC
      SUM=0. 0
      22 R1(2, I1)=SUM
      IF(ICT. EQ. 0) NVR=IC4
      NCR=IN1
      RETURN
   END

SUBROUTINE RSIMP(U, NCR, NVR, KS1M)

SUBROUTINE TO SOLVE THE LINEAR PROGRAMMING
PROBLEM USING THE REVISED SIMPLEX METHOD
THIS PROGRAM USES THE SIMPLEX TABLEAU
GENERATED IN THE ROUTINE SIMPTB. X, B, AX
ARE WORK VECTORS. THEIR SIZES ARE AS
FOLLOWS :-
X(IT+3) , E(IT+3) , AX(2+ITY)
IT :- IS AS DEFINED IN THE MAIN PROGRAM
ITY :- IS AS DEFINED IN THE ROUTINE SIMPTB

DIMENSION: A(31, 1122), B(24), U(34, 34), X(24), E(34),
+CT(15, 561), R1(2, 561), R2(2, 1122), AX(1122)
COMMON : B7, MP1, M01, NF1, NO1, NNC, NDC, M1
COMMON : B25, MP2, B, CT, RL, R2
REAL*8 X1, Y1, X2
REAL*8 XLIN
KS1M=0
KK1=M1+M1
KK2=KK1+KK1
NCR1=NCR-1
NEQ=NCR+2
IKT=2
SP=2.0
DO 2 I=1,NEQ
DO 2 J=1,NEQ
2 U(I,J)=0.0
DO 4 I=1,NEQ
4 U(I,I)=1.0
ITR=1
EMU=-1.E-03
EMV=-EMU
IX=0
XLIM=1.D+75
24 XMIN=1.E+75
IOVER=0
DO 5 I=1,NVR
SUM=0.
IF(I,GT,KK1) GO TO 100
5 DO 6 J=1,NCR1
6 IF(U(NEQ,J).EQ.0.0) GO TO 6
XL=U(NEQ,J)
YL=RJ(I)
X1=X1+XL
IF(X2.GE.XLIM) IOVER=IOVER+1
IF(IOVER.GE.5) GO TO 31
X11=SNGL(X2)
SUM=SUM+X11
6 CONTINUE
X=SUM
DELF=SUM+U(NEQ,NCR)+RZ(IKT,1)
GO TO 102
100 IF(I,GT,KK2) GO TO 101
DELF=-AX(I-KK1)+U(NEQ,NCR)-(RZ(IKT,1-KK1)+SF)
GO TO 102
101 IKK2=1-KK2
DO 66 J=1,NDC
IF(U(NEQ,J).EQ.0.0) GO TO 66
SUM=SUM+U(NEQ,J)*CT(J,IKK2)
66 CONTINUE
DELF=SUM+R1(IKT,IKK2)
102 IF(XMIN.LE.DELJ) GO TO 5
XMIN=DELJ
KK=I
5 CONTINUE
IF(I,GT,0) GO TO 21
IF(XMIN.GE.1E-04) GO TO 22
21 IF(KK.GT.KK2) GO TO 206
IF(KK.GT.KK1) GO TO 201
KKCOL=KK
GO TO 202
201 KKCOL=KK-KK1
202 DO 7 I=1,NCR
XI=0.0
DO 8 J=1,NCR1

IF(U(I, J).EQ.0, 0) GO TO 8
X(I)=X(I)+U(I, J)*A(J, KKCOL)
8 CONTINUE
IF(KK, GT, KK1) X(I)=-X(I)
7 X(I)=X(I)+U(I, NCR)
DO 77 JZ=1, IKT
LZ1=NCR+LZ
X(LZ1)=0.
DO 88 JZ=1, NCR1
88 X(LZ1)=X(LZ1)+U(LZ1, JZ)*A(JZ, KKCOL)
IF(KK, GT, KK1) GO TO 301
X(LZ1)=X(LZ1)+U(LZ1, NCR)+R2(LZ, KKCOL)
GO TO 77
301 IF(LZ, EQ, 1) X(LZ1)=-X(LZ1)+U(LZ1, NCR)-R2(LZ, KKCOL)
IF(LZ, EQ, 2) X(LZ1)=-X(LZ1)+U(LZ1, NCR)-(R2(LZ, KKCOL)+SF)
77 CONTINUE
GO TO 203
200 KKCOL=KK-KK2
KNT=0
DO 90 I=1, NEQ
X(I)=0.0
DO 91 J=1, NDC
IF(U(I, J).EQ.0, 0) GO TO 91
X(I)=X(I)+U(I, J)*CT(J, KKCOL)
91 CONTINUE
IF(I, LE, NCR) GO TO 30
KNT=KNT+1
X(I)=X(I)+R1(KNT, KKCOL)
30 CONTINUE
202 CONTINUE
XMIN=1. E70
KKJ=0
DO 10 I=1, NCR
IF(X(I), LE, 0.) GO TO 10
THETA=K(I)/X(I)
IF(XMIN, LT, THETA) GO TO 10
XMIN=THETA
KKJ=1
10 CONTINUE
IF(KKJ, EQ, 0) GO TO 35
DO 12 I=1, NEQ
12 EX(I)=-X(X(I), X(KKJ))
EK(KKJ)=1.*X(KKJ)
Y1=EK(KKJ)
DO 14 I=1, NEQ
14 BK(I)=B(I)+E(I)*Y1
Y1=U(KKJ, I)
DO 16 J=1, NEQ
16 U(J, I)=U(J, I)+E(J)*Y1
15 U(KKJ, I)=E(KKJ)*Y1
ITR=ITR+1
IF(IM, EQ, 1) GO TO 19
IF(B(NEG), GE. EMU, AND, B(NEG), LE. EMV) GO TO 18
IF(B(NEG), GT. EMV) GO TO 20
GO TO 19
18 IX=1
NEG=NEG-1
IKT=IKT-1
SF=0.
GO TO 24
19 PRINT 35
35 FORMAT('','2X,'NO LOWER BOUND FOR OPTIMUM',')
KSIM=1
GO TO 32
20 PRINT 41
41 FORMAT('','2X,'NO FEASIBLE SOLUTION',')
KSIM=1
WRITE(6,505) IX, ITR, B(NEG)
505 FORMAT('','2X,'PHASE - ', I1, 2X,'NO. OF ITR = ', I3)
*IX, 'OBJ. FN. VAL= ', E14.7, ')
GO TO 32
37 WRITE(6,52)
52 FORMAT('','2X,'OVER FLOW IN SIMPLEX ROUTIN
*B',')
KSIM=1
GO TO 32
32 WRITE(6,42) ITR
42 FORMAT('','2X,'OPTIMAL SOLUTION FOUND... ', 2X,'N
*O. OF ITERATIONS= ', I6, ')
WRITE(6,72) B(NCR+1)
72 FORMAT('','2X,'THE OPTIMUM VALUE OF THE OBJECTI
*VE FUNCTION= ', E14.7, ')
22 CONTINUE
RETURN
END

SUBROUTINE PLOTMP(ML, NL, A, B, KTIME, GDLY1, GDLY2)

SUBROUTINE TO COMPUTE AND PRINT THE MAGNITUDE AND
GROUP-DELAY CHARACTERISTICS OF THE TWO-DIMENSIONAL
RECURSIVE DIGITAL FILTER. THIS PROGRAM ALSO
COMPUTES THE SQUARED ERROR BETWEEN THE DESIRED &
DESIGNED CHARACTERISTICS. H IS THE ARRAY TO
STORE THE MAGNITUDE, T1 IS THE ARRAY TO STORE
THE GROUP-DELAY IN +VE VERTICAL DIRECTION AND T2
IS THE ARRAY TO STORE GROUP-DELAY IN THE +VE
HORIZONTAL DIRECTION. THE SIZE OF ARRAYS LOCAL
to this subroutine are as follows :-
H(ML/NL), T1(ML/NL), T2(ML/NL), Z1(NL)
Z2(NL)

DIMENSION A(6,6), B(6,6), Z1(22), Z2(17), W1(15), W2(17),
H(22,17), T1(22,17), T2(22,17)
DIMENSION F(22,17), DM(22,17), DP(22,17)
LOGICAL &I FE, EZ
COMPLEX Z1, Z2, Z3, CESP, CMPLX, AN, AD, AD, ANL, AN2, AD1, AD2,
HINT, ZA, ZB
COMMON/B7/MPI, MQ1, NP1, MQ1, NN, ND, MNU
COMMON/B8/DM, DP, W1, W2, WPI, NP2
COMMON/B38/FB
PY=3.1415926
IF(KTIME.GE.2) GO TO 6
Z1(I)=-1, 0, 0, 0
DO 1 I=1, N1
ZZ=CMPLX(0, W1(I)*PY)
1 Z1(I)=CEXP(-Z2)
ZZ(1)=(1, 0, )
DO 2 J=2, N1
ZZ=CMPLX(0, W2(J)*PY)
2 Z2(J)=CEXP(-Z2)
6 CONTINUE
ERM=0.
ERT1=0.
ERT2=0.
DO 3 I=1, N1
DO 3 J=1, N1
BZ=FB(I, J)
HN=(0, 0, 0,)
HD=(0, 0, 0,)
HN1=(0, 0, 0,)
HD1=(0, 0, 0,)
HN2=(0, 0, 0,)
HD2=(0, 0, 0,)
DO 4 K=1, NP1
ZA=Z1(I)+(K-1)
DO 4 L=1, NP1
HNT=A*(K, L)*Z2*(Z2(J)+(L-1))
HN=HN+HNT
HN1=HN1+(K-1)*HNT
4 HN2=HN2+HNT*(L-1)
DO 5 K=1, NP1
ZA=Z1(I)+(K-1)
DO 5 L=1, NP1
HNT=B*(K, L)*Z2*(Z2(J)+(L-1))
HD=HD+HNT
HD1=HD1+(K-1)*HNT
5 HD2=HD2+(L-1)*HNT
H(I, J)=CABS(HN)/HD
T1(I, J)=REAL((HN1/HN)-(HD1/HD))
T2(I, J)=REAL((HN2/HN)-(HD2/HD))
ERM=ERM+(DM(I, J)-H(I, J))*2
IF(.NOT.BZ) GO TO 2
ERT1=ERT1+(CDL*(T1(I, J)))*2
ERT2=ERT2+(CDL*(T2(I, J)))*2
2 CONTINUE
WRITE(6, 700)
WRITE(6, 701)
CALL OUTPUT(H, M1, N1)
WRITE(6, 704) ERM
WRITE(6, 700)
WRITE(6,702)
CALL OUTPUT(T1, M1, N1)
WRITE(6,703) ERT1
WRITE(6,700)
WRITE(6,702)
CALL OUTPUT(T2, M1, N1)
WRITE(6,706) ERT2

700 FORMAT(' ')/
701 FORMAT(' ',2X,'THE DESIGNED MAGNITUDE RESPONSE'/)
702 FORMAT(' ',2X,'GD1: THE GROUP-DELAY ALONG VERT-DI'
     'RECT'/)
703 FORMAT(' ',2X,'GD2: THE GROUP-DELAY ALONG HORIZ-DI'
     'RECT'/)
704 FORMAT(' ',2X,'SUM OF THE SquARED ERRORS IN MAG'
     '== E12.5'/)
705 FORMAT(' ',2X,'SUM OF THE SquARED ERRORS IN GD1'
     '== E12.5'/)
706 FORMAT(' ',2X,'SUM OF THE SquARED ERRORS IN GD2'
     '== E12.5')
RETURN
END

C SUBROUTINE OUTPUT(A, M, N)
C
C SUBROUTINE TO PRINT OUT THE OUTPUT
C REQUIRED BY SUBROUTINE PLOTMP
C
DIMENSION A(35,17)
NCOL=11
NZ=(N/11)+1
NFREM=N-(NZ-1)*11
IF(N, GT, 11) GO TO 1

NZ=1
NFREM=N
1 CONTINUE
NCOL=0
DO 6 J=1, NZ
   IF(J, EQ, NZ) NCOL=NFREM
   DO 7 I=1, M
7 WRITE(6,500) (A(K,NCOL+K),K=1,NCOL)
   WRITE(6,501)
   NCOL=NCOL+NCOL
6 CONTINUE

500 FORMAT(' ',2X,'11(F10.4,1X)
501 FORMAT(' ',//')
RETURN
END
EXTERNAL FUNCT  
DIMENSION X(100), DG(100), W1(41), W2(21), Z1(41), Z2(21)  
C  
MAIN PROGRAM :- DESIGNING TWO DIMENSIONAL RECURSIVE  
DIGITAL FILTERS IN CASCADES OF 1ST AND 2ND ORDER  
SECTIONS USING NON-LINEAR OPTIMIZATION PROCEDURE  
IN THIS PROGRAM GIVEN DATA CAN BE APPROXIMATED TO  
THE REAL PART OR THE MAGNITUDE SQUARED TRANSFER  
FUNCTION OF THE RECURSIVE DIGITAL FILTER  
THE INPUT DATA ARE :-  
M1 - # OF FREQUENCY POINTS IN W1 AXIS  
M2 - # OF FREQUENCY POINTS IN W2 AXIS  
KS,KF - # OF 2ND AND 1ST ORDER CASCADES RESPECTIVELY  
LIMIT - # OF ITERATIONS, ER - DESIRED FREQUENCY  
DOMAIN SPECS, X - INITIAL PARAMETER VALUES (ARE TO  
BE NON-ZERO VALUES), EST - ESTIMATED MINIMUM OF  
THE OBJECTIVE FUNCTION, EPS - STEP SIZE, LP - THE  
EXponential FACTOR OF THE ERROR (VALUES SHOULD BE  
2, 4, 6, ETC.)  
THE METHOD USES THE FLETCHER AND POWELLS OPTIMIZATION  
PROCEDURE.  
SUBROUTINES REQUIRED :- FUNCT, OUTPUT, DREAL, DIMAG  
AND DFMFP(FLETCHER AND POWELLS OPTIMIZATION SCHEME)  
C  
THE DIMENSIONS OF THE VARIOUS ARRAYS ARE :-  
X(N), DG(N), W1(M1), W2(M2), Z1(M1), Z2(M2), ZA(M1), ZB(M1)  
YS(KS,18), YF(KF,6), IR(M2), S(3*KS), F(KF*4), TA(M1),  
TB(M2), TD(M2), TC(M1), ER(M1, M2), RP(M1, M2), TE(M1, M2)  
TF(M1, M2), TG(M1, M2), HC(N*(N+7)/2) WHERE  
N EQUALS KS+15+KF*6+1 WHICH IS EQUAL TO THE  
# OF PARAMETERS IN THE DESIGN PROBLEM  
C  
*, ZA(41), ZB(21), YS(4,16), YF(4,6), IR(41)  
DIMENSION S(55), F(32), R(6), TA(41), TB(21), TC(41), TD(21),  
*ER(41,21), RP(41,21), TE(41,21), TF(41,21), TG(41,21), HC(400)  
REAL*8 DABS, DSQRT, DEXP  
REAL*8 X, DG, HC, Q, YS, YF, DREAL, S, F, R  
COMPLEX Z2, CMPLX, CEXP, Z1, Z2, ZA, ZB, HN, HD, POLY, P  
LOGICAL*1 ITYF, IREP, IMGS  
COMMON/B1,YS, YF, ER, RP, TA, TB, TC, TD, TE, TF, TG, AO  
COMMON/B2/KOUNT, M1, M2, KS, KF, LP  
COMMON/B3/ITYF  
DATA IREP/. TRUE./, IMGS/. FALSE. /  
P1=3.141593  
P2=P1/2.  
KOUNT=0  
C  
SET ITYF=IREP FOR REAL PART APPROXIMATION  
IF NOT SET ITYF=IMGS FOR MAGNITUDE SQUARED  
APPROXIMATION  
C  
ITYF=IREP
LP IS TO BE SET EQUAL TO 2, 4, 6 ETC I.E. ANY EVEN NUMBERED INTEGER

LP=2

READ IN # OF FREQUENCY POINTS ALONG W1 AND W2 AXIS IN THE FREQUENCY DOMAIN

READ(5, 500) M1, M2

READ IN THE # OF 2ND AND 1ST ORDER SECTIONS RESPECTIVELY

READ(5, 500) KS, KF

DW1=2./M1
DW2=1./(M2-1)
W1(1)=-1.
W2(1)=0.
DO 1 I=2, M1
1 W1(I)=W1(I-1)+DW1
DO 2 I=2, M2
2 W2(I)=W2(I-1)+DW2
W1(I)=W1(I)+0.0125
W2(I)=W2(I)-0.0125
DO 3 I=1, M1
3 ZZ=CMPLX(0., W1(I)*PY)
TA(I)=TAN(W1(I)*PY2)
TC(I)=TA(I)*TA(I)
Z1(I)=CEXP(-ZZ)

DO 4 I=1, M2

TB(I)=TAN(W2(I)*PY2)
TD(I)=TB(I)*TB(I)
ZZ=CMPLX(0., W2(I)*PY)
Z2(I)=CEXP(-ZZ)

DO 100 J=1, M1

TE(I,J)=TA(I)*TB(J)
TF(I,J)=TC(I)*TB(J)
100 TG(I,J)=TA(I)*TD(J)

READ IN THE FREQUENCY DOMAIN SPECIFICATIONS INTO THE ARRAY ER AND NORMALIZE THE DATA.

DO 554 I=1, M1
DO 554 J=1, M2
554 READ(5, 556) (IR(K), K=1, M2)
554 ER(I, J)=IR(J)

YMAX1=0.
556 FORMAT(16I5)
   DO 557 I=1,M1
   DO 557 J=1,M2
557 ER(I,J)=ER(I,J)/YMAX1
C
   WRITE(6,558) YMAX1
558 FORMAT(' ',1X,'MAXIMUM VALUE IN INPUT DATA=',E12.5)
   NCS=KS*13
   NCF=KF*6
   N=NCS+NCF+1
C
   READ IN THE INITIAL PARAMETER VALUES
C
   READ(5,555) (X(I),I=1,N)
555 FORMAT(3D23.16)
C
   IF(.NOT. ITYF) GO TO 27
   WRITE(6,650)
   GO TO 28
27 WRITE(6,29)
29 FORMAT(' ',1X,'THIS IS A MAGNITUDE SQUARED APPROXIMATION',/)
28 WRITE(6,600) M1,M2
   WRITE(6,601) KS,KF
   WRITE(6,602)
   WRITE(6,603) X(N)
   N1=N/2
   DO 19 I=1,N1
   NX1=N1+I
19 WRITE(6,604) I,X(I),NX1,X(NX1)
   WRITE(6,618)
   WRITE(6,605)
   CALL OUTPUT(ER,M1,M2)
   EST=0.1
   EPS=1.E-5
   LIMIT=40
C
   CALL DFMFP(FUNCT,N,X,Q,DQ,EST,EPS,LIMIT,IER,HC)
C
   WRITE(6,607) KOUNT
   WRITE(6,608) IER
   IF(IER)10,11,12
11 WRITE(6,610)
   GO TO 10
12 WRITE(6,609)
10 WRITE(6,612) Q
   WRITE(6,613)
   DO 20 I=1,N1
   NX1=N1+I
20 WRITE(6,604) I,X(I),NX1,X(NX1)
   WRITE(6,618)
   WRITE(6,614)
   WRITE(6,615)
   WRITE(6,616)
   WRITE(6,617)
WRITE(6, 624)
WRITE(6, 625)
WRITE(6, 626)
WRITE(6, 627)
WRITE(6, 628) X(N)
WRITE(6, 606)
IF(KS, EQ, 0) GO TO 700
KK=-8
KK1=-9
DO 141 I=1, KS
DO 141 J=1, 2
KK=KK+8
KK1=KK1+9
DO 142 L=1, 8
142 R(L)=VS(I, KK+L)
   S(KK1+1)=R(1)+R(2)+R(3)+R(4)+R(5)+R(6)+R(7)+R(8)+1. D0
   S(KK1+2)=2. D0*RL(1)+RL(3)+RL(4)+RL(6)+RL(7)+1. D0
   S(KK1+3)=RL(1)-RL(2)+RL(3)+RL(4)-RL(5)+RL(6)+RL(7)-RL(8)+1. D0
   S(KK1+4)=2. D0*RL(1)+RL(2)+RL(3)-RL(7)-RL(8)-1. D0
   S(KK1+5)=4. D0*RL(1)-RL(3)-RL(7)+1. D0
   S(KK1+6)=2. D0*RL(1)-RL(2)+RL(3)-RL(7)+RL(8)-1. D0
   S(KK1+7)=RL(1)+RL(2)+RL(3)-RL(4)-RL(5)-RL(6)+RL(7)+RL(8)+1. D0
   S(KK1+8)=2. D0*RL(1)-RL(3)-RL(4)+RL(6)+RL(7)-1. D0
141 S(KK1+9)=RL(1)-RL(2)+RL(3)-RL(4)+RL(5)-RL(6)+RL(7)-RL(8)+1. D0
KK=-18
DO 14 I=1, KS
WRITE(6, 625) I
KK=KK+18
DO 15 J=1, 9
JJ=J-1
15 WRITE(6, 619) JJ, S(KK+J), JJ, S(KK+9+J)
14 WRITE(6, 618)
700 IF(KF, EQ, 0) GO TO 701
KK=-3
KK1=-4
DO 161 I=1, KF
DO 161 J=1, 2
KK=KK+3
KK1=KK1+4
DO 162 L=1, 3
162 R(L)=VF(I, KK+L)
   F(KK1+1)=R(1)+R(2)+R(3)+1. D0
   F(KK1+2)=R(1)-R(2)+R(3)-1. D0
   F(KK1+3)=R(1)+R(2)-R(3)-1. D0
161 F(KK1+4)=R(1)-R(2)-R(3)+1. D0
KK=-8
DO 16 I=1, KF
II=KS+I
KK=KK+8
WRITE(6, 625) II
DO 17 J=1, 4
JJ=J-1
17 WRITE(6, 619) JJ, F(KK+J), JJ, F(KK+4+J)
WRITE(6, 618)
CONTINUE
DO 25 I = 1, M1
DO 25 J = 1, M2
IF(.NOT. ITYF) GO TO 26
RP(I, J) = 2*A0*RP(I, J)
GO TO 25
26 RP(I, J) = A0*RP(I, J)
CONTINUE
WRITE(6, 626)
CALL OUTPUT(RP, ML, M2)
FORMAT(2I2)
FORMAT(1X, 'THIS IS A REAL PART APPROXIMATION', /)
FORMAT(1X, 'NO. OF FREQ POINTS ALONG W1-', I2, 2X, 'ALONG W2-', I2, '/*', /)
FORMAT(1X, 'NO. OF SEC-ORDER CASCADES-', I2, 2X, 'NO. OF FIRST-ORD
ER CASCADES-', I2, '/*', /)
FORMAT(1X, 'INITIAL VALUES OF PARAMETERS:', /)
FORMAT(1X, 'AO-', D23.16, '/*', /)
FORMAT(1X, 'X<-', I2, '/>=', 'D23.16', 3X, 'X>', I2, '=/', 'D23.16')
FORMAT(1X, 'DESIR ED SPECIFICATIONS:', /)
FORMAT(1X, 'NO. OF FUNCTION EVALUATIONS-', I6, '/')
FORMAT(1X, 'IER=', I2, '/*', /)
FORMAT(5X, 'CONVERGENCE NOT OBTAINED IN LIMIT ITERATIONS', '/)
FORMAT(5X, 'CONVERGENCE OBTAINED IN LIMIT ITERATIONS', '/)
FORMAT(5X, 'SUM OF THE SQUARED ERRORS BETWEEN DESIRED AND DESIGNED
SPECIFICATIONS=', D15.8, '/')
FORMAT(1X, 'FINAL VALUES OF THE PARAMETERS:', /)
FORMAT(15X, 'NUMERATOR COEFS', 20X, 'DENOMINATOR COEFS', '/')
FORMAT(1X, 'THE SECOND ORDER SECTION IS OF THE FORM:', '/)
FORMAT(11X, 'A(0)+A(1)*Z2+A(2)*Z2**2+A(3)*Z1+f(4)*Z1*Z2+A(5)*Z1
**Z2**2+A(6)*Z1**2+A(7)**Z1**2+Z2+A(8)**Z1**2+Z2**2', '/*')
FORMAT(1X, 'H(Z1, Z2)=---------------------------')
FORMAT(11X, 'B(0)+B(1)*Z2+B(2)*Z2**2+B(3)*Z1+B(4)*Z1*Z2+B(5)*Z1
**Z2**2+B(6)*Z1**2+Z2+B(7)**Z1**2+Z2**2', '/*')
FORMAT(11X, 'THE FIRST ORDER SECTION IS OF THE FORM:', '/)
FORMAT(1X, 'A(0)+A(1)*Z2+A(2)*Z1+A(3)+Z1**2', '/*')
FORMAT(1X, 'H(Z1, Z2)=---------------------------')
FORMAT(11X, 'B(0)+B(1)*Z2+B(2)*Z1+B(3)**Z1+Z2', '/*')
FORMAT(1X, 'I1=', I2, '/*', /)
FORMAT(32X, 'A(1)=', 'D23.16', 15X, 'B(<', I1, '>)=', 'D23.16')
FORMAT(1X, 'CASCADE SECTION NO.=', I2, '/*', /)
STOP
END
SUBROUTINE FUNCT(N, X, Q, DQ)

SUBROUTINE THAT CALCULATES THE ERROR FUNCTION VALUE Q AND THE GRADIENTS DQ. ALSO IT PRINTS OUT THE FUNCTION EVALUATION NUMBER AND ERROR.

THE DIMENSIONS OF THE ARRAYS ARE AS FOLLOWS: U(KS, 5), P(10), V(10), HFD(KF), HSD(KS), HFN(KF)

HSN(KS), SUM1(3*KF), SUM2(3*KF), SUM3(8*KS), SUM4(10*KS)

THE DIMENSION OF THE REMAINING ARRAYS ARE SAME AS IN THE MAIN PROGRAM. THE ARRAY YD CORRESPONDS TO THE ARRAY ER IN THE MAIN PROGRAM.

DIMENSION X(N), DQ(N), YF(4, 6), YS(4, 16), U(4, 5), P(10),
* V(10), TA(41), TB(21), TC(41), TC(21), HFD(4), HSD(4),
* SUM(12), SUM2(12), SUM3(32), SUM4(48), HFN(4), HSN(4),
* TE(41, 21), TF(41, 21), TG(41, 21), RP(41, 21), YD(41, 21)

REAL*8 X, DQ, Q, YF, YS, U, P, V, SUM1, SUM2, SUM3, SUM4, SUMA, HFN, HSN,
* HFD, HFDI, HSNI, HSNR, HSDR, HSDI, DREAL, DIMAG, E1, EA, EH, ET1, ET2, SI1,
* SI2, SI3, SI4, AO1, AO2

LOGICAL*1 ITYF

COMPLEX*16 DCMPLX, PH, HFN, HFD, HSN, HSD, PTF

COMMON/B1/YS, YF, YD, RP, TA, TB, TC, TD, TE, TF, TG, AO

COMMON/B2/KOUNT, M1, M2, KS, KF, LP

COMMON/B3/ITYF

KFT=6+KF

KOUNT=KOUNT+1

IF(KF) 100, 100, 101

101 KFC=-6

DO 1 I=1, KF

KFC=KFC+6

DO 1 J=1, 3

YF(I, J)=X(KFC+J)

1 YF(I, J+3)=X(KFC+J+3)**2

100 IF(KS) 102, 102, 103

103 KSC=KFT+18

M=-8

DO 2 I=1, KS

M=M+16

KSC=KSC+18

DO 3 J=1, 8

YS(I, J)=X(KSC+J)

3 L=KSC+8

DO 4 K=1, 10

P(K)=X(L+K)

4 V(K)=X(L+K)**2

U(I, 1)=P(5)*P(10)-P(6)*P(3)+P(7)*P(8)

U(I, 2)=P(1)*P(8)-P(2)*P(6)+P(3)*P(5)

U(I, 3)=P(1)*P(9)-P(2)*P(7)+P(4)*P(5)

U(I, 5)=P(2)*P(10)-P(3)*P(9)+P(4)*P(8)

U(I, 4)=P(1)*P(10)-P(3)*P(7)+P(4)*P(6)

YS(I, M+1)=U(I, 1)**2

YS(I, M+2)=U(I, 2)**2+U(I, 3)**2

YS(I, M+3)=V(5)
YS(I,M+4)=U(I,4)**2+U(I,5)**2
YS(I,M+5)=V(6)+V(7)+V(8)+V(9)
YS(I,M+6)=V(1)+V(2)
YS(I,M+7)=V(10)
2.4YS(I,M+8)=V(3)+V(4)
102 AO=X(N)
Q=0.D0
SUMA=0.D0
IF(KF)104,104,105
105 KK1=-3
DO 5 K=1,KF
KK1=KK1+3
DO 5 L=L,3
SUM1(KK1+L)=0.D0
5 SUM2(KK1+L)=0.D0
104 IF(KS)106,106,107
107 KK1=-8
KK1=-10
DO 6 K=1,KS
KK1=KK1+8
KK2=KK2+10
DO 7 L=L,8
7 SUM3(KK1+L)=0.D0
DO 6 J=1,10
6 SUM4(KK2+J)=0.D0
106 DO 8 I=1,M1
DO 8 J=1,M2
PH=DCMPLX(1.D0,0.D0)
IF(KF)108,108,109
109 DO 9 K=1,KF
HFNR=YF(K,1)-TE(I,J)
HFNI=YF(K,2)*TB(J)+YF(K,3)*TA(I)
HFN(K)=DCMPLX(HFNR, HFNI)
HFDR=YF(K,4)-TE(I,J)
HFDI=YF(K,5)*TB(J)+YF(K,6)*TA(I)
HFD(K)=DCMPLX(HFDR, HFDI)
9 PH=PH*HFN(K)/HFD(K)
108 IF(KS)110,110,111
111 DO 10 K=1,KS
HSNR=YS(K,1)-YS(K,3)*TD(J)-YS(K,5)*TE(I,J)-YS(K,7)*TC(I)
**TE(I,J)**=2
HSNI=YS(K,2)*TB(J)+YS(K,4)*TA(I)-YS(K,6)*TG(I,J)-YS(K,8)*TF(I,J)
HSDR=YS(K,9)-YS(K,11)*TD(J)-YS(K,13)*TE(I,J)-YS(K,15)*TC(I)
**TE(I,J)**=2
HSDI=YS(K,10)*TB(J)+YS(K,12)*TA(I)-YS(K,14)*TG(I,J)-YS(K,16)*TF
**(I,J)**
HSD(K)=DCMPLX(HSDR, HSDI)
10 PH=PH*HSD(K)/HSD(K)
110 IF(NOT.ITYF)GO TO 125
RP(I,J)=SNGL(DREAL(PH))
EA=2.*AO*RP(I,J)-YD(I,J)
E1=EA**(LP-1)
EH=E1
EHA=2! *E1*RP(I, J)
PFT=PH
GO TO 126

125 
RP(I, J)=CNGL(CDABS(PH))**2 
EA=AO*RP(I, J)-YD(I, J) 
E1=EA**2*LP-1 
EH=E1*RP(I, J) 
PFT=(1, 0D, 0, 0D0) 
EHA=EH

GO TO 126

Q=Q+E1*EA 
ET1=EH*TB(J) 
ET2=EH*TA(I) 
IF(KF) 112, 112, 113
113 
KK=-3 
DO 11 K=1, KF 
KK=KK+3 
SI1=DIMAG(PFT/HFN(K)) 
SI2=DIMAG(PFT/HFD(K)) 
SUM1(KK+1)=SUM1(KK+1)-EH*DREAL(PFT/HFN(K)) 
SUM1(KK+2)=SUM1(KK+2)+ET1*SI1 
SUM1(KK+3)=SUM1(KK+3)+ET2*SI1 
SUM2(KK+1)=SUM2(KK+1)-EH*DREAL(PFT/HFD(K)) 
SUM2(KK+2)=SUM2(KK+2)+ET1*SI2 
SUM2(KK+3)=SUM2(KK+3)+ET2*SI2
11 
IF(KS) 8, 8, 115
115 
KL=8 
L=KFT-10 
DO 12 K=1, KS 
SI1=DREAL(PFT/HSN(K)) 
SI2=DIMAG(PFT/HSN(K)) 
SI3=DREAL(PFT/HSN(K)) 
SI4=DIMAG(PFT/HSN(K)) 
KJ=KJ+8 
KL=KL+10 
L=L+18 
SUM3(KJ+1)=SUM3(KJ+1)-EH*SI1 
SUM3(KJ+2)=SUM3(KJ+2)+ET1*SI2 
SUM3(KJ+3)=SUM3(KJ+3)+EH*TD(J)*SI1 
SUM3(KJ+4)=SUM3(KJ+4)+ET2*SI2 
SUM3(KJ+5)=SUM3(KJ+5)+EH*TEC(I, J)*SI1 
SUM3(KJ+6)=SUM3(KJ+6)-EH*TG(I, J)*SI2 
SUM3(KJ+7)=SUM3(KJ+7)+EH*TC(I)*SI2 
SUM3(KJ+8)=SUM3(KJ+8)-EH*TF(I, J)*SI2 
SUM4(KL+1)=SUM4(KL+1)+EH*(TB(J))*(U(K, 2)**(L+S)+U(K, 3)**(L+S)) 
**+(TA(I))*(U(K, 4)**(L+10)-TG(I, J)**(L+2))*(L+1)**SI4 
SUM4(KL+2)=SUM4(KL+2)+EH*(-TB(J))*(U(K, 2)**(L+6)+U(K, 3)**(L+7)) 
**+(TA(I))*(U(K, 5)**(L+10)-TG(I, J)**(L+2))*(L+2)**SI4 
SUM4(KL+3)=SUM4(KL+3)+EH*(TB(J))*(U(K, 2)**(L+5)-TA(I)**(U(K, 4)* 
**X(L+7)+U(K, 5)**(L+9)-TF(I, J)**(L+3)**SI4 
SUM4(KL+4)=SUM4(KL+4)+EH*(TB(J))*(U(K, 2)**(L+5)+TA(I)**(U(K, 4)* 
**X(L+6)+U(K, 5)**(L+9)-TF(I, J)**(L+4)**SI4 
SUM4(KL+5)=SUM4(KL+5)+EH*DREAL(DCMPLX(U(K, 1)**(L+10)-TD(J)* 
**X(L+5), (TB(J))*(U(K, 2)**(L+3)+U(K, 3)**(L+4)))/HSD(K))
SUM4(KL+6) = SUM4(KL+6) - EH*DREAL(DCMPLX((-U(K, 1)*X(L+9)-TE(I, J)*
*X(L+6)), (-TB(J)*U(K, 2)*X(L+2)+TA(I)*U(K, 4)*X(L+4))/HSD(K))
SUM4(KL+7) = SUM4(KL+7) - EH*DREAL(DCMPLX(U(K, 1)*X(L+8)-TE(I, J)*
*X(L+7)), (-TB(J)*U(K, 3)*X(L+2)+TA(I)*U(K, 4)*X(L+4))/HSD(K))
SUM4(KL+8) = SUM4(KL+8) - EH*DREAL(DCMPLX(U(K, 1)*X(L+7)-TE(I, J)*
*X(L+8)), (TB(J)*U(K, 2)*X(L+1)+TA(I)*U(K, 5)*X(L+4))/HSD(K))
SUM4(KL+9) = SUM4(KL+9) - EH*DREAL(DCMPLX((-U(K, 1)*X(L+6)-TE(I, J)*
*X(L+9)), (TB(J)*U(K, 3)*X(L+1)-TA(I)*U(K, 5)*X(L+4))/HSD(K))
12 SUM4(KL+10) = SUM4(KL+10) - EH*DREAL(DCMPLX(U(K, 1)*X(L+5)-TE(I)*
*X(L+10)), TA(I)*U(K, 4)*X(L+1)+U(K, 5)*X(L+2))/HSD(K))
8 SUMA = SUMA + EH

114 AO1 = -2*LP*AO
115 AO2 = -AO1*2. D0
116 IF(KF) 116, 116, 117
117 KK = -6
118 KKL = -3
119 DO 14 K = 1, KF
120 KKL = KKL + 3
121 KK = KK + 6
122 KJ = KK + 3
123 DO 14 LX = 1, 3
124 DQ(KK+LX) = AO1*SUM1(KKL+LX)
14 DQ(KJ+LX) = AO2*SUM2(KKL+LX)*X(KJ+LX)
116 IF(KS) 118, 118, 119
118 KJ = KJ - 18
119 KA = 8
120 KB = 10
121 DO 15 K = 1, KS
122 KJ = KJ + 18
123 KA = KA + 8
124 KB = KB + 10
125 DO 16 K1 = 1, 8
16 DQ(KJ+K1) = AO1*SUM3(KA+K1)
126 KC = KJ + 8
127 DO 15 K2 = 1, 18
15 DQ(KC+K2) = AO2*SUM4(KB+K2)
118 DQ(N) = LP*SUMA
128 WRITE(6, 800) KOUNT, Q
800 FORMAT(2X, 'FUNCTION EVALUATION NO. = ', I6, 2X, 'ERROR = ', D15.8)
END
Program PICSAMP

The program 'PICSAMP' is used to scan the image line by line. One can actually scan up to 4097 x 4097 rectangular array points in an image. Since the display facility can only display images of size 256 x 256 or less, the images are scanned in such a manner so as to fit into an array of 256 x 256.

Consider an image that is scanned as a rectangular array of points as shown in Figure H1. Figure H1(a) shows the rectangular sampling grid. Figure H1(b) shows the extreme integer co-ordinate values of the largest possible sampling grid. The input data to the D/A has to be less than or equal to 12 bits in length, where one bit is reserved for the sign. The co-ordinates of the sampling grid in Figure H1(a) should fall within that of Figure H1(b). The input parameters and subroutines required are as follows:

Parameters:

NSIZE: # of picture elements desired.
NLINES: # of lines to be sampled.
IXSTEP: step size in x direction.
IYSTEP: step size in y direction.
IXSHIFT: starting x co-ordinate.
IYSHIFT: starting y co-ordinate.
The values for all the above parameters should be chosen such that all the sampling grid points lie within the area of Figure H1(b).
Figure H1 Sampling of Image in a Rectangular Array.
SYINIT: subroutines that initialize the A/D and D/A systems.
PSINIT: subroutine to position the CRT beam to the initial
starting point of the sampling grid.
BEGIN: subroutine that waits for operator response to
start the image sampling.
DARUNP: subroutine that enables the sampling of image pixel
by pixel along each horizontal line and storing it
in an array IPIC.

After providing all the parameters, the program sets
up the A/D and D/A converter systems and types out the string
of characters "***TO START SAMPLING STRIKE ANY KEY***"; then
it enters the routine BEGIN and waits for the operator to
strike any key on the console. As soon as a key is struck,
the program returns from routine BEGIN and executes the rest
of the program.

Program PICTEST

The program 'PICTEST' is a test routine which can be
used to continuously scan a desired area of the image that
is to be sampled. While the image is being continuously
scanned, the brightness for the CRT spot is adjusted, such
that there is no saturation at the output of the photo-
multiplier tube or at the output of the integrator. Once
this is done, the program 'PICSAMP' can be used to sample
that particular area of the image.

In order to stop the program 'PICTEST', it is required
to set the NOVA-840 data switches to a value other than -27\(^10\)
(i.e. 100033\(_8\)). The description of the input parameters to this
255 minus the intensity values of the original image.

PICROT: Program to rotate the picture through an angle of 90° in a counter clockwise direction.

Image Display Program

CVIDISPI: A program used for displaying the sampled and processed image using the CVI Expander.

Input parameters for this program are:

M2: # of columns of the image to be displayed (1≤M2≤256).
M1: # of rows (1≤M1≤256).
IXSHIFT: x co-ordinate of the point from where the display begins.
IYSHIFT: y co-ordinate of the point from where the display begins.

IXSHIFT and IYSHIFT are integer values and they should be subject to the following limits:

0 ≤ IXSHIFT ≤ 256 and IXSHIFT + M2 ≤ 256
0 ≤ IYSHIFT ≤ 256 and IYSHIFT + M1 ≤ 256

Subroutines required are:

CVSET: sets up the A/D and D/A systems.
CVWRIT: program to transfer data from computer.
BEGIN: same as before.
DIMENSION IRAFT(2,512), IPIC(512), NAME(10), INIT(2)

********** PICSAMP **********

SUBROUTINE FILES REQUIRED: - SYINIT, PSINIT, BEGIN, DARUNP

DATA IZ/* /
DO 10 I=1,10
10 NAME(I)=IZ
ACCEPT"# OF PIXELS PER LINE = ", NSIZE
ACCEPT"# OF LINES TO BE SAMPLED = ", NLINE
WRITE(10,500)
500 FORMAT(1x, 'TYPE IN THE FILE NAME FOR IMAGE = ', Z)
READ(1L, 500)*(NAME(I), I=1, 10)

600 FORMAT(10A2)
NSIZE1=NSIZE
NSIZE2=2*NSIZE1
ACCEPT "STEP SIZE IN X-DIRECTION = ", IXSTEP
ACCEPT "STEP SIZE IN Y-DIRECTION = ", IYSTEP
ACCEPT "STARTING X-COORDINATE = ", IXSHIFT
ACCEPT "STARTING Y-COORDINATE = ", IYSHIFT
INIT(1)=IXSHIFT
INIT(2)=IYSHIFT
IERR=0
CALL SYINIT(IERR)
IF(IERR.GT.0) GO TO 50
NSIZEA=NSIZE1
NSIZEB=NSIZE2
DO 2 J=2, NSIZE
2 IRAFT(1,J-1)=IXSHIFT+(J-1)*IXSTEP
IRAST(1,NSIZE)=IXSHIFT
IYVAL=IYSHIFT
CALL PSINIT(INIT, IERR)
IF(IERR.GT.0) GO TO 75
TYPE" 
TYPE"**** TO START SAMPLING STRIKE ANY KEY ****"
TYPE" 
OPEN 1, NAME, LEN=2*NSIZE, REC=NLINE
CALL BEGIN
DO 1 I=1, NLINE
DO 4 K=1, NSIZE
4 IRAFT(2,K)=IYVAL
IYVAL=IYVAL-IYSTEP
IRAST(2,NSIZE)=IYVAL
CALL DARUNP(IRAST, IPIC, IERR, NSIZEA, NSIZEB, 110001K)
IF(IERR.GT.0) GO TO 200
1 WRITE(1) (IPIC(NA), NA=1, NSIZE)
CLOSE 1
STOP
50 TYPE"ERROR IN SYINIT: ERROR CODE # = ", IERR
STOP
75 TYPE"ERROR IN PSINIT: ERROR CODE # = ", IERR
STOP
200 TYPE"ERROR IN DARUNP: ERROR CODE # = "+IERR
CLOSE 1
STOP
END

; PROGRAM FILE NAME : - SYINIT
; TITLE SYINIT
; ENT SYINIT
ZREL
SYINIT : SYINIT
NREL
SYINIT : SAVE 0
LDA 0, ARGO, 3
STA 0, IERR
SUB 0, 0
STA 0, @IERR
LDA 0, ADDEY
SYSTM
DEBL
JMP .+1
LDA 0, DADEV
SYSTM
DEBL
JMP .+1
LDA 0, ADDEY
LDA 1, ADCTAD
MOVZL 1, 1
MOVOR 1, 1
; SET BIT 0
LDA 2, BLKA
SYSTM
IDEF
JMP ERR1
LDA 0, DADEV
LDA 1, ADCTDA
MOVZL 1, 1
MOVOR 1, 1
LDA 2, BLKB
SYSTM
IDEF
JMP ERR1
RTN
ERR1 : STA 2, @IERR
RTN
IERR : 0
BLKA : 6
BLKB : 12
ADDEY : 21
DADEV : 23
ADCTAD : +2
ADCTDA : +4
DCTAD : BLK 3
DCTDA : BLK 3
END
PROGRAM FILE NAME: PSINIT

TITLE PSINIT
ENT PSINIT
ZREL

PSINIT: PSINIT
MACRO SETMAP
LDA 0, ^1
LDA 1, ^2
SYSTM
STMAP
JMP ERR1
DOB 1, ^3
%
NREL

PSINIT: SAVE 0
LDA 0, ARG0, ^3
STA 0, IBUF1
LDA 0, ARG1, ^3
STA 0, IERR
SUB 0, 0
STA 0, @IERR
LDA 0, DACCHAN
DOAC 0, 23
SETMAP DADEV IBUF1 23
LDA 0, ADTRIG
DOAC 0, 21
LDA 0, DACNT
DOCP 0, 23
SKPDN 23
JMP -1
LDA 0, DADBL
DOAC 0, 23
LDA 0, ADDEV
DOAC 0, 21
LDA 0, ADCHAN
DOA 0, 21
RTN

ERR1: STA 2, @IERR
RTN

IERR: 0
ADCHAN: 140000
DACCHAN: 120443
ADTRIG: 127102
DACNT: -1
IBUF1: 0
ADDEV: 21
DADEV: 23
DADBL: 110443
ADDDBL: 103002
.END
```
.TITLE DARUNP
.ENT DARUNP
.ZREL

DARUNP: .DARUNP
.MACRO SEIMAP
LDA 0,-1
LDA 1,-2
.SYSTM
.STMAP
.JMP ERR1
.DOBC 1,-3

.DARUNP: .SAVE 0
.LDA 0,ARG0,3
.STA 0,IBUF1
.LDA 0,ARG1,3
.STA 0,IBUF2
.LDA 0,ARG2,3
.STA 0,IERR
.LDA 0,@ARG3,3
.STA 0,ADCNT
.LDA 0,@ARG4,3
.STA 0,ADCNT
.LDA 0,@ARG5,3
.STA 0,ADTRIG
.SUB 0,0
.STA 0,@IERR
.SYSTM
.ODIS
.JMP .+1
.SETMAP DAEV IBUF1 23
.SETMAP ADDEV IBUF2 21
.LDA 0,ADTRIG
.DOA 0,21
.LDA 0,ADCNT
.DOCP 0,23
.LDA 0,ADCNT
.DOCP 0,21
.SKPDN 21
.JMP .-1
.LDA 0,ADDBL
.DOA 0,21
.SYSTM
.OEBL
.JMP .+1
.RTN

ERR1: .STA 2,@IERR
.RTN

.IERR: 0
.ADTRIG: 0
.ADCNT: 0
```
DIMENSION IRAST(2,512), IPIC(512), INIT(2)

************ PICTEST ************

SUBROUTINE FILES REQUIRED: - SYINIT, PSINIT
BEGIN, DARUNP

IEST=100033K
ACCEPT"# OF PIXELS PER LINE = ", NSIZE
ACCEPT"# OF LINES TO BE SAMPLED = ", NLINE
NSIZE1=NSIZE
NSIZE2=2*NSIZE1
ACCEPT "STEP SIZE IN X-DIRECTION = ", IXSTEP
ACCEPT "STEP SIZE IN Y-DIRECTION = ", IYSTEP
ACCEPT "STARTING X-COORDINATE = ", IXSHIFT
ACCEPT "STARTING Y-COORDINATE = ", IYSHIFT
INIT(1)=IXSHIFT
INIT(2)=IYSHIFT
IER=0
CALL SYINIT(IERR)
IF(IERR.GT.0) GO TO 50
NSIZEA=-NSIZE1
NSIZEB=-NSIZE2
DO 2 J=2, NSIZE
2 IRAST(1,J-1)=IXSHIFT+(J-1)*IXSTEP
IRAST(1,NSIZE)=IXSHIFT
TYPE" 
TYPE"*** TO START TESTING STRIKE ANY KEY ***"
TYPE" 
CALL BEGIN
3 IYVAL=IYSHIFT
CALL PSINIT(INIT, IERR)
IF(IERR.GT.0) GO TO 75
DO 4 I=1, NLINE
DO 4 K=1, NSIZE
4 IRAST(2,K)=IYVAL
IYVAL=IYVAL-IYSTEP
IRAST(2,NSIZE)=IYVAL
CALL DARUNP(IRAST, IPIC, IERR, NSIZEA, NSIZEB, 110001K)
IF(IERR.GT.0) GO TO 200
CALL OUT(IYVAL)
IF(IYVAL.NE. ITEST) STOP
1 CONTINUE
GO TO 3
50 TYPE"ERROR IN SYINIT: ERROR CODE # = ", IERR
STOP
75 TYPE"ERROR IN PSINIT: ERROR CODE # = ", IERR
STOP
200 TYPE"ERROR IN DARUNP: ERROR CODE # = ", IERR
STOP
END
PROGRAM FILE : OUT
   TITLE OUT
   ENT OUT
   ZREL

OUT:
   OUT
   NREL

   OUT: SAVE 0
   LDA 0, ARG0, I
   STA 0, IVAL
   SUB 0, 0
   STA 0, @IVAL
   SYSTM
   RDSW
   JMP +1
   STA 0, @IVAL
   RTN

IVAL: 0
END

DIMENSION IWINDE(16384), IX(56, 266), NAME(5)

********** NORMPIC **********

IMAGE NORMALIZATION

COMMON/B1/IWINDE
EQUIVALENCE,(IWINDE, IX)
CALL VMEM(ICNT, IER)
IF(IER.EQ.5) GO TO 3000
   TYPE"* OF FREE 1024-WORD BLOCKS= ", ICNT
   CALL MAPDF(ICNT, IWINDE, 16, IER)
   IF(IER.GE.5) GO TO 3001
   WRITE(IO, 100)
100 FORMAT(1X, 'FILE NAME OF IMAGE= ', Z)
   READ(IO, 200) (NAME(I), I=1, 5)
200 FORMAT(5A2)
   ACCEPT "$ OF COLOMNS OF IMAGE = ", M2
   ACCEPT "$ OF ROWS OF IMAGE = ", M1
   ACCEPT "INTENSITY FACTOR = ", FACT
   TYPE"TO START ------ TYPE ANY KEY"
   CALL BEGIN
   OPEN 1, NAME, LEN=2*M2, REC=M1
   M1X=M1/54
   IF(M1X.EQ.0) GO TO 20
   MREM=M1-M1X*54
   IF(MREM.GT.0) GO TO 21
   MREM=54
   GO TO 22
21 M1X=M1X+1
   GO TO 22
20 M1X=1
   MREM=M1
22 KB=-16

M1A=64
DO 23 I=1, M1X
KB=KB+16
CALL REMAP(0, KB, 16, IER)
IF(IER.GE.29) GO TO 3002
IF(I.EQ. M1X) M1A=MREM
DO 23 J=1, M1A
23 READ(1) (IX(J,K), K=L,M2)
REWIND 1
KB=-16
M1A=64
DO 24 I=1, M1X
KB=KB+16
CALL REMAP(0, KB, 16, IER)
IF(IER.GE.29) GO TO 3002
IF(I.EQ. M1X) M1A=MREM
DO 25 J=1, M1A
DO 25 K=L,M2
25 IX(J,K)=((FLOAT(IX(J,K))+6192.0)/64.0)*FACT
DO 24 L=1, M1A
24 WRITE(1) (IX(L,N), N=L, M2)
GO TO 300
3000 TYPE"ERROR IN VMEM ; ERROR # = ", IER
GO TO 400
3001 TYPE"ERROR IN MAPDF ; ERROR # = ", IER
GO TO 400
3002 TYPE"ERROR IN REMAP ; ERROR # = ", IER
300 CLOSE 1
400 STOP
END

DIMENSION IWIND(15384), IX(64, 256), NAME(3)

************ INVERTPIC ************

INVERTING A NORMALIZED IMAGE

COMMON/B1/IWIND
EQUIVALENCE (IWIND, IX)
CALL VMEM(ICNT, IER)
IF(IER.EQ.5) GO TO 3000
TYPE"# OF FREE 1024-WORD BLOCKS= ", ICNT
CALL MAPDF(ICNT, IWIND, 16, IER)
IF(IER.GE.5) GO TO 3001
WRITE(10,100)
100 FORMAT(2,1X,'FILE NAME OF IMAGE= ',Z)
READ(11,200) (NAME(I), I=1,5)
200 FORMAT(SA2)
ACCEPT "# OF COLOUMNS OF IMAGE = ", M2
ACCEPT "# OF ROWS OF IMAGE = ", M1
TYPE "TO START -- TYPE ANY KEY"
CALL BEGIN
OPEN 1, NAME, LEN=2*M2, REC=M1
M1X = M1 / 64
IF (M1X .EQ. 0) GO TO 20
MREM = M1 - M1X * 64
IF (MREM .GT. 0) GO TO 21
MREM = 64
GO TO 22

20 M1X = M1X + 1
GO TO 22

21 M1X = 1
MREM = M1

22 KB = -16
M1A = 64
DO 23 I = 1, M1X
KB = KB + 16
CALL REMAP (0, KB, 16, IER)
IF (IER .GE. 29) GO TO 3002
IF (I .EQ. M1X) M1A = MREM.
DO 23 J = 1, M1A

23 READ (1) (IX (J, K), K = 1, M2)
REWIND 1
KB = -16
M1A = 64
DO 24 I = 1, M1X
KB = KB + 16
CALL REMAP (0, KB, 16, IER)
IF (IER .GE. 29) GO TO 3002
IF (I .EQ. M1X) M1A = MREM.
DO 25 J = 1, M1A
DO 25 K = 1, M2

25 IX (J, K) = 255.0 - FLOAT (IX (J, K))
DO 24 L = 1, M1A.

24 WRITE (1) (IX (L, N), N = 1, M2)
GO TO 300

3000 TYPE "ERROR IN VMEM ; ERROR # = ", IER
GO TO 400

3001 TYPE "ERROR IN MAPDF ; ERROR # = ", IER
GO TO 400

3002 TYPE "ERROR IN REMAP ; ERROR # = ", IER
300 CLOSE 1
400 STOP
END

DIMENSION IWIND(16384), IY(256, 64), IA(32, 256), NAME(5)

******************************************************************************
C
C PROGRAM TO ROTATE THE IMAGE
C
C COMMON /B1/IWIND
EQUIVALENCE (IWIND, IY)
CALL VMEM (ICNT, IER)
IF (IER .EQ. 5) GO TO 1000
TYPE "# OF FREE 1024-WORD BLOCKS= ", ICNT
ICNT=ICNT-1
CALL MAPDF(ICNT, IWIN, 16, IER)
IF(IER.GE.5) GO TO 1001
TYPE"SIZE OF IMAGE TO BE ROTATED (> 64x64)"
ACCEPT"# OF COLOUMNS (POWER OF 2) = ", M2
ACCEPT"# OF ROWS (POWER OF 2) = ", M1
WRITE(10,100)
100 FORMAT(1X, 'FILE NAME OF IMAGE= ',Z)
READ(11,200) (NAME(I), I=1,5)
200 FORMAT(SA2)
OPEN 1, NAME, LEN=2*M2, REC=M1
M21=M2+1
IR1=64
IR2=32
IT1=M1/IR1
IT2=M21/IR2
IB=16
KK=-IB
DO 1 K=L, IT1
KK=KK+IB
CALL REMAP(0, KK, IB, IER)
IF(IER.GE.29) GO TO 1002
DO 1 I=L, IR1
1 READ(1) (IV(M21-J, I), J=1, M2)
REWIND 1
KK1=-IR2
DO 2 K=L, IT2
KK1=KK1+IR2
KK2=-IR1
KK=-IB
DO 3 L=1, IT1
KK=KK+IB
KK2=KK2+IR1
CALL REMAP(0, KK, IB, IER)
IF(IER.GE.29) GO TO 1002
DO 3 I=1, IR2
DO 3 J=1, IR1
3 IA(I, KK2+J)=IV(KK1+I, J)
WRITE(1) (IA(IM, IK), IK=1, M1)
CLOSE 1
STOP
1000 TYPE"ERROR IN VMEM ; ERROR # = ", IER
CLOSE 1
STOP
1001 TYPE"ERROR IN MAPDF ; ERROR # = ", IER
CLOSE 1
STOP
1002 TYPE"ERROR IN REMAP ; ERROR # = ", IER
CLOSE 1
STOP
END
DIMENSION IXYZ(3, 4096), IFILE(5)

*************** CVDISPI ***************

PROGRAM TO WRITE PICTURE DATA ON CVI(LINE BY LINE)

SUBROUTINES REQUIRED:- CVSET, CVWRIT, BEGIN

ACCEPT"SAMPLING RATE= ", ISAMP1
ISAMP=ISAMP1
IERR=0
CALL CVSET(0, 0, IERR)
IF(IERR.GT.0) GO TO 150

ACCEPT"# OF COLOUMNS TO BE DISPLAYED= ", M2
ACCEPT"# OF ROWS TO BE DISPLAYED= ", M1
IWZ=0

ACCEPT"STARTING X - CO-ORDINATE= ", IXSHIFT
ACCEPT"STARTING Y - CO-ORDINATE= ", IYSHIFT
NSL=(M1/16)+1
NREM=(NSL*16)-M1
IF(NREM.EQ.16) NSL=NSL-1
IF(NREM.EQ.16) NREM=0
IF(IWZ.EQ.1) GO TO 400
ACCEPT"WISH TO CLEAR(0), OR DISPLAY(1) ON THE SCREEN?->", ID
IF(ID.EQ.0) GO TO 300
WRITE(10, 100)

100 FORMAT(" ", "INPUT FILE NAME: --", Z)
READ(11, 200)(IFILE(I), I=1, 5)

200 FORMAT(5A2)
ID=1
OPEN 1, IFILE, LEN=2*M2, REC=M1
CONTINUE
NST=16
TYPE"
TYPE"TO START DISPLAYING TYPE ANY CHARACTER"
TYPE"
CALL BEGIN
DO 30 K1=1, NSL
IF(K1.EQ. NSL) NST=16-NREM
DO 25 K2=1, NST
ICON=(IYSHIFT+(K1-1)*16+(K2-1))*2+(14000K)
J2=(K2-1)*M2
IF(ID.EQ.0) GO TO 320
READ(1) (IXYZ(3, KZ+J2), KZ=1, M2)
GO TO 330

250 DO 340 KQ=1, M2
340 IXYZ(3, KQ+J2)=0
CONTINUE
DO 25 J=1, M2
J1=J2+J
IXYZ(2, J1)=ICON
IXYZ(1, J1)=IXSHIFT+J-1+(142000K)
IXYZ(3, J1) = IXYZ(3, J1) + (103000K)

CONTINUE
MCOUNT = 1 * M2 * NST
IERR = 0
CALL CVWRITE(IXYZ, ISAMP, MCOUNT, IERR)
IF(IERR.GT.0) GO TO 160

CONTINUE
TYPE"WISH TO DISPLAY ANOTHER IMAGE HAVING"
TYPE"THE SAME NUMBER OF COLOUMNS AND ROWS"
ACCEPT"FROM THE SAME STARTING POINT? YES(1), NO(0)
* ----->", IWI
IF(IWI.EQ.0) GO TO 310
CLOSE 1

IF(IWI.NE.0) GO TO 400
ACCEPT"ERASE LAST DISPLAYED IMAGE? YES(1), NO(0)
* -------", IZR
IF(IZR.EQ.1) ID = 0
IF(IZR.EQ.1) GO TO 300
TYPE"WISH TO CHANGE X AND Y STARTING CO-ORDINATES"
TYPE"AND DISPLAY SAME IMAGE OR ANOTHER IMAGE OF"
ACCEPT"SAME # OF COLOUMNS AND ROWS? YES(1), NO(0)
* ----->", IWZ
IF(IWZ.EQ.1) GO TO 345
ACCEPT"WISH TO START FROM THE BEGINING? YES(1),
NO(0)->", IW2
IF(IW2.EQ.1) GO TO 500
STOP

160 TYPE"ERROR IN CVWRITE: CODE=", IERR
CLOSE 1
STOP

END

.TITL CVWRITE
.ENT CVWRITE
.ZREL

CVWRITE: .CVWRITE
.NREL

CVWRITE: SAVE 0
LDA 0, ARG0, 3
STA 0, IXYZ
LDA 0, @ARG1, 3
STA 0, ISAMP
LDA 0, @ARG2, 3
STA 0, MCOUNT
LDA 0, ARG3, 3
STA 0, IERR
LDA 0, DADEV
LDA 1, IXYZ
.SYSTM

.TITL CVSET
.ENT CVSET
.ZREL

CVSET: .CVSET
.NREL

CVSET: SAVE 0
LDA 0, ADCHAN
LDA 1, @ARG0, 3
ADDZ 1, 0
STA 0, ADCHAN
LDA 0, DACHAN
LDA 1, @ARG1, 3
ADDZ 1, 0
STA 0, DACHAN
LDA 0, ARG2, 3
STA 0, IERR
LDA 0, DADEV
LDA 1, IXYZ
.SYSTM
SUB 0, 0
STA 0, @IERR
LDA 0, ADDEV
.SYSTM
.DEBL
JMP .+1
LDA 0, DADEV
.SYSTM
.DEBL
JMP .+1
LDA 0, ADCHAN
DOAC 0, 21
LDA 0, ADTRIG
DOA 0, 21
LDA 0, DACHAN
DOAC 0, 23.
LDA 0, ADDEV
LDA 1, ADCTAD
MOVZL 1, 1
MOVOR 1, 1
LDA 2, BLK1
.SYSTM
.IDEF
JMP ERR1
LDA 0, DADEV
LDA 1, ADCTDA
MOVZL 1, 1
MOVOR 1, 1
LDA 2, BLKNUM
.SYSTM
.IDEF
STA 2, @IERR
RTN

ERR1: STA 2, @IERR
RTN

ADCHAN: 140000
ADTRIG: 123000
DACHAN: 136001
IERR: 0
BLKNUM: 15
BLK1: 5
ADDEV: 21
DADEV: 23
ADCTAD: .+2
ADCTDA: .+4
DCTAD: .BLK 3
DCTDA: .BLK 3

.STMAP
JMP ERR
DOA 1, 23
LDA 0, ADDEV
LDA 1, ATEMP
.SYSTM
.STMAP
JMP ERR
DOA 1, 21
LDA 0, MCOUNT
DOCC 0, 23
LDA 0, ADCOUNT
DOCC 0, 21
LDA 0, ISAMP
DOAP 0, 23
DOAP 0, 21
SKPDA 23
JMP .-1
RTN

ERR: STA 2, @IERR
RTN

IXYZ: 0
ISAMP: 0
MCOUNT: 0
IERR: 0
ADDEV: 21
DADEV: 23
ATEMP: .+2
ADCOUNT: .-2
TEMP: .BLK 2

.END
DIMENSION IWIN(4096), IX(64, 64), PS(32, 64)

********** PESTOLN **********
ESTIMATION OF THE AVERAGE POWER SPECTRUM OF
A GIVEN IMAGE OVER A GIVEN SEGMENT SIZE
SEGMENTS CAN BE OVERLAPPED AND WINDOWED
IF DESIRED. THE SIZE OF THE SEGMENT IS
N = 2**N. N IS TO BE LESS OR EQUAL TO 6. THE
INPUT IMAGE SIZE IS = M2 * M1. M2 AND M1
ARE TO BE POwers OF 2 AND SHOULD BE LESS OR
EQUAL TO 256. THE ESTIMATED POWER SPECTRUM IS
WRITTEN INTO A NEW FILE.

DIMENSION X(64, 64), IV(64/256), NAME(5), WD(32, 64)
COMPLEX X, CMPLX
COMMON/E1/IWIN;
COMMON/BK/X
EQUIVALENCE (IWIN, IX, PS, WD)
EQUIVALENCE (X, IV)
CALL VMEN(ICNT, IER)
IF(IER.EQ.5) GO TO 1000
TYPE"# OF FREE 1024-WORD BLOCKS = ", ICNT
CALL MAPDF(ICNT, IWIN, 4), 2, IER
IF(IER.GE.5) GO TO 1001
ACCEPT"FFT SIZE = 2**N J N = ", N
N1=2**N
N1A=N1/2
N1=1+M1
TYPE"SIZE OF THE INPUT DATA ( > OR = FFT SIZE ) ",
ACCEPT"# OF COLOURING (POWER OF 2) = ", M2
ACCEPT"# OF ROWS (POWER OF 2) = ", M1
KINC=4
M2A=M2/2
M1A=M1/2

IF THE D.C ( OR MEAN ) VALUE OF THE IMAGE IS TO BE
REMOVED THEN SET MDC EQUAL TO 1 AND READ IN THE
D.C VALUE

TYPE"
ACCEPT"OVERLAP THE SEGMENTS? YES(1), NO(0) -------") IOL
ACCEPT"WINDOW THE SEGMENTS? YES(1), NO(0) -------") , IWD
ACCEPT"ORIGIN OF P.S AT CENTRE? YES(1), NO(0) ------") , IMD
ACCEPT"REMOVE D.C VALUE OF IMAGE? YES(1), NO(0) ------") , MDC
IF(MDC.EQ.1) ACCEPT"D.C COMPONENT VALUE = ", DCV
IF(IOL.EQ.1) GO TO 201
KINC=2
M2A=(M2/2+1)*2-1
M1A=(M1/2+1)*2-1
M2A1=M2/2-1

201 TYPE "<15>"
TYPE"# OF SEGMENTS IN HORIZ DIRECTION= "M2A
TYPE"# OF SEGMENTS IN VERT DIRECTION= "N1A
TYPE "<15>"
KB1=12
DO 9 L1=1, 2
KB1=KB1+4
CALL REMAP(0, KB1, 4, IER)
IF(IER.GE.23) GO TO 1002
DO 9 K=1, NA1
DO 9 L=1, N1
9 PS(K,L)=0, 0
WRITE(10, 100)
100 FORMAT(1X, 'INPUT DATA FILE NAME= ', Z)
READ(11, 200) (NAME(I), I=1, 5)
200 FORMAT(5A2)
OPEN 1, NAME, LEN=2*N2, REC=N1
IF(IW=0. GO TO 202
WRITE(10, 104)
104 FORMAT(1X, ' FILE NAME OF WINDOW= ', Z)
READ(11, 200) (NAME(I), I=1, 5)
OPEN 3, NAME, LEN=4*N1, REC=N1
KB2=20
C
C READING IN 2D-WINDOW VALUES IF IWD=1
C
DO 11 L2=1, 2
KB2=KB2+4
CALL REMAP(0, KB2, 4, IER)
IF(IER.GE.23) GO TO 1002
DO 11 KA=1, NA1
11 READ(3) (WD(KA, KZ), KZ=1, N1)
CLOSE 3
202 WRITE(10, 101)
101 FORMAT(1X, 'FILE NAME FOR OUTPUT= ', Z)
C
C READ IN FILE NAME TO STORE THE ESTIMATED POWER SPECTRUM VALUES
C
READ(11, 200) (NAME(I), I=1, 5)
SUM=0. 0
IF(MDC.EQ.1) SUM=DCV
CALL MAPDF(1, IER)
IF(IER.GE.5) GO TO 1001
NXT=N1
NIT=0
DO 1 I=1, N1A
DO 2 J=1, NXT
C
C READING IMAGE DATA OF BLOCK SIZE N1 X N2
C
2 READ(1Y(KY(NIT+1, KX), KX=1, N2))
IF(I, EQ. 1. OR. 10L, EQ. 8) GO TO 112
KB=4
KS=-N1
DO 14 J1=1, M2A1
KB=KB+4
KS=KS+N1
CALL REMAP(0, KB, 4, IER)
IF(IER.GE.29) GO TO 1002
DO 14 K=1, NA1
DO 14 L=1, N1
14 IY(K, KS+L)=IY(NA1+K, L)
223 KB=-4
KS=-N1
C
C STOP THE PART OF INPUT DATA FOR OVERLAPPING THE
C DATA FOR NEXT ESTIMATION.
C
DO 3 J1=1, M2A1
KB=KB+4
KS=KS+N1
CALL REMAP(0, KB, 4, IER)
IF(IER.GE.29) GO TO 1002
DO 3 K=1, N1
DO 3 L=1, N1
3 IX(K, L)=IY(K, KS+L)
KB=0
DO 8 J2=1, M2A
WRITE(10, 102) I, J2
102 FORMAT(1X, 4, 1X, 'COMPUTING POWER SPECTRUM OF
* SEGMENT = ( / , 12 , , / , 12 , ) ' )
CALL REMAP(0, KB, 4, IER)
IF(IER.GE.29) GO TO 1002
IF(IND. EQ. 1) GO TO 222.
C
C SUBTRACTING THE D. C VALUE FROM THE DATA
C
DO 5 K=1, N1
DO 5 L=1, N1
XXA=IX(K, L)-SUM
5 IX(K, L)=CMPLX(XXA, 0.0)
GO TO 333
222 ICA=-1
C
C SHIFING THE ORIGIN OF THE POWER SPECTRUM
C TO THE CENTRE BY MULTIPLYING THE INPUT
C DATA BY -1.
C
DO 50 K=1, N1
ICA=ICA*(-1)
ICB=-1
DO 50 L=1, N1
ICB=ICB*(-1)
ICC=ICB*ICA
XXA=(IX(K, L)-SUM)*ICC
50 IX(K, L)=CMPLX(XXA, 0.0)
223 CALL NAPDF(2, IER)
MULTIPLYING THE DATA BY THE WINDOW VALUES,

\[ \text{DO 12 LI}=1,2 \]
\[ K\text{S2}=\text{K}\text{S2}+\text{NA1} \]
\[ K\text{B2}=\text{K}\text{B2}+4 \]
\[ \text{CALL REMAP}(0, \text{KB2}, 4, \text{IER}) \]
\[ \text{IF}(\text{IER}. \text{GE}. 29) \text{ GO TO 1002} \]
\[ \text{DO 12 KE}=1, \text{NA1} \]
\[ \text{DO 12 KU}=1, \text{NI} \]
\[ 12 \quad X(K\text{S2}+\text{KE}, \text{KU})=X(K\text{S2}+\text{KE}, \text{KU})*\text{WD}(\text{KE}, \text{KU}) \]

COMPUTING THE TWO-DIMENSIONAL FFT

\[ \text{204 \ CALL FFTPZ}(N, N, N, N, -1, \text{NL}, \text{NL}) \]
\[ K\text{B1}=12 \]
\[ K\text{S1}=-\text{NA1} \]

SUMMING THE POWER SPECTRUM OF SEGMENTS

\[ \text{DO 4 LI}=1,2 \]
\[ K\text{S1}=K\text{S1}+\text{NA1} \]
\[ K\text{B1}=K\text{B1}+4 \]
\[ \text{CALL REMAP}(0, K\text{B1}, 4, \text{IER}) \]
\[ \text{IF}(\text{IER}. \text{GE}. 29) \text{ GO TO 1002} \]
\[ \text{DO 4 K}=1, \text{NA1} \]
\[ \text{DO 4 L}=1, \text{NI} \]
\[ X(\text{XX}+K\text{S1}+K, \text{L}) \]
\[ 4 \quad F(\text{K}, \text{L})=F(\text{K}, \text{L})+(\text{XX}+\text{XX})/\text{NIN1} \]
\[ \text{CALL MAPDF}(1, \text{IER}) \]
\[ \text{IF}(\text{IER}. \text{GE}. 5) \text{ GO TO 1001} \]
\[ K\text{B}=K\text{B}+\text{KINC} \]
\[ \text{IF}(10L. \text{EQ}. 0) \text{ \text{GO TO 1} \] 
\[ \text{NXT}=\text{NA1} \]
\[ \text{NIT}=\text{NA1} \]

1 CONTINUE
\[ \text{MC}=\text{N2A}+\text{N1A} \]
\[ \text{CALL MAPDF}(2, \text{IER}) \]
\[ \text{IF}(\text{IER}. \text{GE}. 5) \text{ GO TO 1001} \]
\[ \text{OPEN 2, NAME=LEN=4*NL, REC=NL} \]
\[ K\text{B1}=12 \]
\[ \text{WRITE}(10, 103) \]

103 FORMAT(/,",", 4X, "COMPUTING AVERAGE POWER \/
\* SPECTRUM", ")

\[ \text{DO 6 LI}=1,2 \]
\[ K\text{B1}=K\text{B1}+4 \]
\[ \text{CALL REMAP}(0, K\text{B1}, 4, \text{IER}) \]
\[ \text{IF}(\text{IER}. \text{GE}. 29) \text{ GO TO 1002} \]

COMPUTING THE AVERAGE POWER SPECTRUM
DO 7 K=1, NA1
DO 7 L=1, N1
7 PS(K, L)=PS(K, L)/MC
DO 6 K1=1, NA1
6 WRITE(2) (PS(K1, K2), K2=1, N1)
CLOSE 2
CLOSE 1
STOP
1000 TYPE"ERROR IN VMEM ; ERROR # = ", IER
GO TO 500
1001 TYPE"ERROR IN MAPDF ; ERROR # = ", IER
GO TO 500
1002 TYPE"ERROR IN REMAP ; ERROR # = ", IER
CLOSE 2
-- 500 CLOSE 1
STOP
END
FILTERING DATA USING THE OVERLAP-ADD METHOD OF FFT

SUBROUTINE TFT

THE INPUT IMAGE SIZE = 2*M X 2*N AND M IS TO BE
LESS OR EQUAL TO 8, THE FFT ARRAY SIZE IS EQUAL TO
2*M X 2*N, AND N IS TO BE LESS OR EQUAL TO 76.
LET M = 2*M, THEN THE BLOCK SIZE OF IMAGE THAT IS
OVERLAPPED THE FILTERED IMAGE IS WRITTEN BACK
TO THE ORIGINAL IMAGE FILE.

DIMENSION XN(64), XN(64)
READ(12, 200) <NAME(I), I=1,5>
TYPE"SIZE OF IMAGE"
ACCEPT"# OF COLOUMNS=ROWS= M = ", M
NC=2**M
OPEN 2, NAME, LEN=2*NC, REC=NC
NC1=(2**NC)/M1
M2=M1/2
MTN=M2*NC
NT=NC1/2
DO 1 I=1, NC1
IREC=(I-1)*M2+1
CALL fseek(2, IREC)

READING OUT AN IMAGE BLOCK OF SIZE NC X M2

DO 7 I1=1, M2
7 READ(2) <ID(I1, I2), I2=1, NC>
CALL MAPDF(I, IER)
IER1=5
IF(IER1.GE.5) GO TO 1001
MIN=-M1
KBL=14
DO 400 IU=1, NT
KBL=KBL+2
MIN=MIN+M1
CALL REMAP(0, KBL, 2, IER)
IER2=2
IF(IER2.GE.29) GO TO 1003
DO 400 IV=1, M2
DO 400 IW=1, M1
400 IT(IW, IW)=ID(IW, MIN+IW)
IJ=16
IK=24

C FILTERING OVERLAPPED IMAGE BLOCKS OF SIZE EQUAL TO M1 X 2+M1

DO 100 J=1, NT
IBL1=(J-1)*2+1
IBL2=IBL1+1
WRITE(10, 555) I, IBL1, I, IBL2
555 FORMAT(4A10) "PROCESSING BLOCKS ((12, ), (12, ))",
& "((12, ), (12, ))"
CALL REMAP(0, IJ, 2, IER)
IER2=1
IF(IER2.GE.29) GO TO 1003
IF(I.EQ.1) GO TO 100
IF(J.EQ.1) GO TO 210
GO TO 320
300 DO 2 K1=1, M2
2 DO 2 K2=1, M1
2 \( \times (K1, K2) = (0, 0, 0) \)
   IF (I.EQ. 1) GO TO 320
320 DO 4 K1=1, M2
   DO 4 K2=1, M2
      \( \times (M2+K1, K2) = \text{COMPLEX}(\text{FLOAT}(I2), I1, K2) \), \text{FLOAT}(I1, K1, K2) \)
4 \( \times (M2+K1, M2+K2) = \text{COMPLEX}(\text{FLOAT}(I1, K1, K2), \text{FLOAT}(I1, K2, K2)) \)
   IF (I.EQ. 1) GO TO 330
   GO TO 10
330 DO 3 K1=1, M2
   DO 3 K2=1, M2
      \( \times (M2+K1, K2) = \text{COMPLEX}(0, \text{FLOAT}(I1, K1, K2)) \)
3 \( \times (M2+K1, M2+K2) = \text{COMPLEX}(\text{FLOAT}(I1, K1, K2), \text{FLOAT}(I1, K2, K2)) \)
   IF (I.EQ. 1) GO TO 10
   CALL REMAP(0, I1, K2, IER)
      IER2=2
   IF (IER.GE.29) GO TO 1001
   DO 5 K1=1, M2
   DO 5 K2=1, M2
      \( \times (K1, K2) = \text{COMPLEX}(0, \text{FLOAT}(IY(K1, K2))) \)
5 \( \times (K1, M2+K2) = \text{COMPLEX}(\text{FLOAT}(IY(K1, K2)), \text{FLOAT}(IY(K1, K2))) \)
   GO TO 10
10 CALL REMAP(0, I1, K2, IER)
      IER2=3
   IF (IER.GE.29) GO TO 1001
   DO 6 K1=1, M2
   DO 6 K2=1, M2
      \( \times (K1, K2) = \text{COMPLEX}(\text{FLOAT}(IY(K1, K2)), \text{FLOAT}(IY(K1, K2))) \)
6 \( \times (K1, M2+K2) = \text{COMPLEX}(\text{FLOAT}(IY(K1, K2)), \text{FLOAT}(IY(K1, K2))) \)
   GO TO 10

COMPUTE FFT OF OVERLAPPED IMAGE BLOCKS
10 CALL FFTP2(6, 6, 6, 6, -1, M1, M1)
CALL MAPDF(4, IER)
      IER1=2
   IF (IER.GE.5) GO TO 1001
      KS=-8

FREQUENCY DOMAIN MULTIPLICATION OF THE FFT OF IMAGE WITH THE TRANSFER FUNCTION OF FILTER

DO 11 KA=1, 8
   KBK=(KA-1)*8
   CALL REMAP(0, KBK, 2, IER)
      IER2=4
   IF (IER.GE.29) GO TO 1001
      KS=KS+2
   DO 11 KB=1, 8
      KSA=KS+KB
   DO 11 KC=1, M1
11 X(KSA, KC) = C(KSA, KC) * M1(KC, KB) / 4096.0

C COMPUTE THE INVERSE FFT OF THE PRODUCT OF
C FFT OF IMAGE BLOCK AND TRANSFER FUNCTION
C
CALL FFTP2(6, 6, 6, 1, M1, M1)
CALL MAPDF(1, IER)
IER1 = 4
IF (IER. GE. 5) GO TO 1001
CALL REMAP(0, IJ, 2, IER)
IER2 = 5
IF (IER. GE. 29) GO TO 1003
DO 12 KA = 1, M2
12 DO 12 KB = 1, M1
IF (KB .LE. M2) GO TO 12
IT1(KA, KB-M2) = IT(KA, KB)
13 IE(KA, KB) = IT(KA, KB)
CALL REMAP(0, IK, 2, IER)
IER2 = 6
IF (IER. GE. 29) GO TO 1003
DO 14 KA = 1, M2
14 DO 14 KB = 1, M2
15 IY(KA, KB) = IY(KA, M2+KB)
DO 402 KA = 1, M2
DO 402 KB = 1, M1
402 IY(KA, KB) = IY(KA, KB)
CALL REMAP(0, IJ, 2, IER)
IER2 = 7
IF (IER. GE. 29) GO TO 1003
DO 15 KA = 1, M2
15 DO 15 KB = 1, M2
IT(KA, KB) = REAL(C(KM2+KA, M2+KB))
15 IT(KA, M2+KB) = IMAG(C(KM2+KA, M2+KB))
IJ = IJ + 2
100 IY = IK + 2
MIN = -M1
KBL = 14
DO 401 IU = 1, NT
KBL = KBL + 2
MIN = MIN + M1
CALL REMAP(0, KBL, 2, IER)
IER2 = 9
IF (IER. GE. 29) GO TO 1003
DO 401 IV = 1, M2
DO 401 IW = 1, M1
401 ID(IY, MIN+IW) = IT(IY, IW)
CALL FSEEK(2, IREC)

C WRITING BACK THE FILTERED IMAGE BLOCK INTO
C THE SAME IMAGE FILE
C
DO 15 II = 1, M2
15 WRITE(2) (ID(II, IJ), IJ = 1, NC)
1. CONTINUE
STOP
1000 TYPE"ERROR IN VMEM ->ERROR #: " IER
STOP
1001 TYPE"ERROR IN MAPDF ->ERROR #: " IER1," ERROR #: "
STOP
1002 TYPE"ERROR IN REMAP ->ERROR #: " IER2," ERROR #: "
STOP
END

SUBROUTINE FFTP2(N1, M2, L1, L2, SI, NX, NY)

*************** TDFFT ***************

SUBROUTINE FFTP2 COMPUTES THE TWO-DIMENSIONAL FFT
OF A GIVEN TWO-DIMENSIONAL ARRAY OF SIZE NX X NY.

DIMENSION X(64, 64)
DOUBLE PRECISION RA, BB, CC, SCL, ARG, UCOS, DSIN.
COMPLEX CMPLX, W, XT
COMMON/BLK/x,
DATA CC, '6, 26319920717266,' N1=2**M1
N2=2**M2
LP1=2**L1
LP2=2**L2
DO 9 LL=1, LP1
DO 1 LO=1, M2
LMX=2**M2-LO
LIX=2*LMLX
SCL=CC/LIX
IF (LO*M2+LMX) 2, 2, 3
2 DO 4 LM=1, LP2
ARG=CMPLX(AR1, BB)
RA=UCOS(ARG)
BB=SI+DSIN(ARG)
AR1=SNGL(RA)
BB1=SNGL(BB)
W=CMPLX(AR1, BB1)
DO 4 LI=LIX, N2, LIX
J1=L1-LIX+LM
J2=J1+LM
4 X(LL, J2)=W*X(LL, J1)
GO TO 1
3 DO 5 LM=1, LMX
ARG=CMPLX(AR1, BB1)
RA=UCOS(ARG)
BB=SI+DSIN(ARG)
AR1=SNGL(RA)
BB1=SNGL(BB)
W=CMPLX(AR1, BB1)
DO 5 LI=LIX, N2, LIX
J1=L1-LIX+LM
J2=J1+LM
5
$T = X(J1) - X(LL, J2)$

$X(LL, J1) = X(LL, J1) + X(J1, J2)$

5 $X(LL, J2) = 0 + T$

1 CONTINUE

NV2 = N2/2

NM1 = N2 - 1

J = 1

DO 7 I = 1, NM1

IF(I .GE. J) GO TO 6

T = X(LL, J)

X(LL, J) = X(LL, I)

X(LL, I) = T

6 K = NV2

IF(K .GE. J) GO TO 7

J = J - K

K = K / 2

GO TO 8

7 J = J + K

9 CONTINUE

DO 10 LL = 1, N2

DO 11 LM = 1, NM1

LMX = 2 ** (M1 - LO)

LIX = 2 ** LMX

SCL = CC / LIX

IF(LO - M1 + L1) 12, 12, 13

12 DO 14 LM = 1, LMX

ARG = (LM - 1) * SCL

AA = DCOS(ARG)

BB = SIN(ARG)

AA1 = SNGL(AA)

BB1 = SNGL(BB)

W = CMPLX(AA1, BB1)

DO 14 L1 = LIX, NM1, LIX

J1 = L1 - LIX + LM

J2 = J1 + LMX

14 X(J2, LL) = W * X(J1, LL)

GO TO 11

13 DO 15 LM = 1, LMX

ARG = (LM - 1) * SCL

AA = DCOS(ARG)

BB = SIN(ARG)

AA1 = SNGL(AA)

BB1 = SNGL(BB)

W = CMPLX(AA1, BB1)

DO 15 L1 = LIX, NM1, LIX

J1 = L1 - LIX + LM

J2 = J1 + LMX

T = X(J1, LL) - X(J2, LL)

X(J1, LL) = X(J1, LL) + T

X(J2, LL) = W + T

11 CONTINUE

NV2 = N1/2

NM1 = N1 - 1

J = 1
DO 17 I=1, NML
IF (I .GE. J) GO TO 18
  T=X(J, LL)
  X(J, LL)=X(I, LL)
  X(I, LL)=T
16  K=NV2
16  IF (K .GE. J) GO TO 17
  J=J-K
  K=K/2
  GO TO 16
17  J=J+K
10  CONTINUE
     RETURN
    END
DIMENSION A(10,10), B(10,10), NAME(5), IWINID(16384)

***************  TORPE  **************

TWO DIMENSIONAL RECURSIVE FORWARD FILTERING
BY A RECURSIVE FILTER WHICH IS REALIZED IN
THE DIRECT FORM, THE SIZE OF THE INPUT DATA
IS M1 X M2. M1 AND M2 ARE TO BE POWERS OF
2, AND SHOULD BE G.E. 64 AND L.E. 256.

DIMENSION WN(2,264), X0(64,256)
INTEGER X0
COMMON/W1/IWINID
EQUIVALENCE (IWIND,X0)
CALL VMENK(1CNT, IER)
IF (IER.EQ. 5) GO TO 1000
TYPE"# OF FREE 1024-WORD BLOCKS="1CNT
CALL MAPOF(1CNT, IWIND, 16, IER)
IF (IER.EQ. 5) GO TO 1001
TYPE "SIZE OF INPUT IMAGE"
ACCEPT"# OF COLUMNS = " M1
ACCEPT"# OF ROWS = " M1
WRITE(10, 100)

100 FORMAT(' ',1X,'FILE NAME OF IMAGE=' ',2)
READ(11, 100) (NAME(I), I=1, 5)

C READ IN A MULTIPLICATION FACTOR AFC
C IF THE FILTERED OUTPUT IS TO BE AMPLIFIED
C BY A CERTAIN FACTOR. IF NOT READ IN
C AFC = 1.
C
ACCEPT"MULTIPLICATION FACTOR= " AFC

OPEN 2,NAME, LEN=2+M2, REC=M1

200 FORMAT('5A2')

C READ COEFFICIENTS OF THE FILTER. NF1 &
C NGL ARE THE HIGHEST POWERS OF 21 & 22
C RESPECTIVELY IN THE NUMERATOR OF THE
C 2D TRANSFER FUNCTION. SIMILARLY NF1 & NGL
C ARE FOR THE DENOMINATOR. AO IS THE GAIN
C FACTOR OF THE TRANSFER FUNCTION.

WRITE(10, 102)
READ(11, 200) (NAME(I), I=1, 4)

102 FORMAT(' ',1X,'FILE NAME OF COEF= ',2)
OPEN 4, NAME
READ(4) NF1, NG1, NF1, NG1
READ(4) (( A(I,J), J=1, NG1), I=1, NF1)
READ(4) (( B(I,J), J=1, NG1), I=1, NF1)
READ(4) AO

CLOSE 4
NORMALIZE THE COEFFICIENTS OF THE FILTER SUCH THAT A(1,1) AND B(1,1) ARE EQUAL TO 1. THE GAIN FACTOR AO IS MODIFIED ACCORDINGLY.

IF(A(1,1) .EQ. 1.0) GO TO 500
AO=AO*A(1,1)
XCOF=A(1,1)
DO 1 I=1, MP1
DO 1 J=1, MQ1
YCOF=A(I, J)
1 A(I, J)=YCOF/XCOF
500 IF(B(1,1) .EQ. 1.0) GO TO 501
XCOF=B(1,1)
AO=AO*XCOF
DO 2 I=1, MP1
DO 2 J=1, MQ1
YCOF=B(I, J)
2 B(I, J)=YCOF/XCOF
501 MP2=MP1+1
MP=MP1-1
MQ=MQ1-1

SET THE INITIAL VALUES OF THE INTERMEDIATE ARRAY WN EQUAL TO ZERO

DO 888 L=1, 6
DO 888 M=1, 1264
888 WN(L, M)=0.0
MT=M1/54
KB=-16

READ THE INPUT DATA AND PERFORM THE FILTERING OPERATION

DO 800 NT=1, NT
KB=KB+16
CALL REMAP(0, KB, 16, IER)
IF(IER .GE. 29) GO TO 1002
DO 800 I=1, 64
800 READ(2) (X0(I, J), J=1, M2)
REWIND 2
KB=-16
DO 1 NT=1, NT
KB=KB+16
CALL REMAP(0, KB, 15, IER)
IF(IER .GE. 29) GO TO 1002
DO 1 I=1, 64
DO 4 J0=1, M2
X0=0
SUM=0
JI=J0+MQ1
JA=JI-1
LS=2
DO 33 K=1, MP1
MPX=MP2-K
DO 44 L=LS, M01
44 SUM=SUM+B(K, L)+WN(MPX, JI-L)
33 LS=1
WN(MP1, JA)=XX=SUM
SUM=0.
LS=2
DO 34 K=1, MP1
MPX=MP2-K
DO 45 L=LS, M01
45 SUM=SUM+B(K, L)+WN(MPX, JI-L)
34 LS=1
XX=SUM+WN(MP1, JA)
4 X0(I, J0)=XX=AFC
DO 177 K=1, N2
JI=K+M0
DO 177 L=1, MP
177 WN(L, JI)=WN(L+1, JI)
3 CONTINUE
C WRITE THE FILTERED DATA BACK TO THE
C SAME FILE FROM WHICH THE INPUT DATA
C WAS OBTAINED.
C
KB=-16
DO 16 NT=1, MT
KB=KB+16
CALL REMAP(KB, KB, I5, IER)
IF (IER.GE.20) GO TO 1002
16 WRITE (2) X0(I, J0), J=1, M2
CLOSE 2
STOP
1000 TYPE "ERROR IN VNAME : ERROR # = "'IER
STOP
1001 TYPE "ERROR IN MAPDF : ERROR # = "'IER
STOP
1002 TYPE "ERROR IN REMAP : ERROR # = "'IER
STOP
END
DIMENSION YS(3,18), YF(2,8), NAME(5)

************ 2DRECMB ***************

REMOVAL OF MOTION BLUR BY FILTERING THE IMAGE BY THE REAL PART OF TWO-D RECURRENT DIGITAL FILTER.
IMAGE SIZE SHOULD BE EQUAL TO ( M1 X M2 ) WHERE M1 & M2 ARE POWERS OF 2 AND GREATER THAN 63
THE DIMENSIONS OF YS, YF, WN, WM, WN1, WN1 SHOULD BE AS FOLLOWS :- YS(KS,18), YF(KF,8), WN(KS,3,258),
WM(KS,3,258), WN1(KF,2,258), WN1(KF,2,258)
THE DIMENSIONS OF OTHER ARRAYS ARE FIXED AND THEY CAN HANDLE IMAGES OF SIZE UPTO 256 X 256
THE PROGRAM WRITES BACK THE DEBLURRED IMAGE INTO THE ORIGINAL BLURRED IMAGE FILE.
KS AND KF ARE THE # OF 2ND AND 1ST ORDER FILTER SECTIONS RESPECTIVELY OF THE 2D FILTER TRANSFER FUNCTION.

DIMENSION WN(3,3,258), WN(3,3,258), WN1(2,2,258),
*WN1(2,2,258)
DIMENSION IWIN(4096), IX(16,256), IY(16,256)
DOUBLE PRECISION YS, YF, A0, B0
COMMON/B1/IWIN
EQUIVALENCE(IWIN, IY)
CALL VMEM(ICNT, IER)
IF (IER.EQ.5) GO TO 3000
TYPE"# OF FREE 1024-WORD BLOCKS= ", ICNT
IWS=4
CALL MAPDF(ICNT, IWIN, IWS, IER)
IF (IER.GE.5) GO TO 3001
TYPE"SIZE OF INPUT IMAGE"

ACCEPT THE IMAGE SIZE AND READ OUT THE IMAGE DATA INTO THE EXTENDED MEMORY

ACCEPT"# OF COLUMNS= ", M2
ACCEPT"# OF ROWS= ", M1
WRITE(10,100)
100 FORMAT( ',1X, 'FILE NAME OF NOISY BLURRED IMAGE= ',Z)
READ(11,200) (NAME(I), I=1,5)
OPEN 2, NAME, LEN=2*M2, REC=M1
NR=16
NR=NR+1
M1X=M1/NR
M2X1=M2+1
M2X2=M2+2
KBX=IWS*M1X
KB=-IWS
DO 60, I=1, M1X
KB=KB+IWS
CALL REMAP(0, KB, IWS, IER)
IF (IER.GE.23) GO TO 3002
DO 60 K=1, NR
60 READ(2) (IY(K, J), J=1, M2)
REWIND 2
C READ IN THE COEFFICIENT FILE NAME AND STORE
C THE COEFFICIENTS IN THE COEFFICIENT ARRAYS
C YS AND YF
C WRITE(10, 102)
READ(11, 200) (NAME(I), I=1, 5)
102 FORMAT(9, 1X, 'FILE NAME OF COEFS = ', 2)
OPEN 4, NAME
READ(4) KS, KF
NT1=KS*N1
NT2=KF*N2
IF(KS .GE. 1) READ(4) (YS(K, L), L=1, 18), K=1, KS
IF(KF .GE. 1) READ(4) (YF(K, L), L=1, 8), K=1, KF
READ(4) A0
CLOSE 4
C SET THE INITIAL VALUES OF THE INTERMEDIATE
C ARRAYS WN, WM, WM1, WMI TO ZERO
C DO 234 NZ1=1, 3
DO 234 NZ2=1, 3
DO 234 NZ3=1, M2X2
WN(NZ1, NZ2, NZ3)=0. D0
234 Wjian(NZ1, NZ2, NZ3)=0. D0
DO 456 NZ1=1, 2
DO 456 NZ2=1, 2
DO 456 NZ3=1, M2X1
WN1(NZ1, NZ2, NZ3)=0. D0
456 WM1(NZ1, NZ2, NZ3)=0. D0
KX1=-NR+1
KB1=KBX
C FILTER THE IMAGE BY THE REAL PART OF THE 2D FILTER
C DO 3 I=1, M1X
KX1=KX1+NR
CALL FS11EK(2, KX1)
DO 43 IM=1, NR
43 READ(2) (IX(IM, IN), IN=1, M2)
KB1=KB1-IWS
CALL REMAP(0, KB1, IWS, IER)
IF(IER .GE. 29) GO TO 3002
DO 555 IO=1, NR
DO 4 J=1, M2
XX=A0*FLOAT(IX(IO, J))
YY=A0*FLOAT(IY(NRA-IO, M2X1-J))
IF(KS .EQ. 0) GO TO 222
JZ=J+2
DO 50 K=1, KS
C
\[ WN(K, 3, JZ) = XX - YS(K, 11) * WN(K, 3, JZ - 1) - YS(K, 12) * WN(K, 3, JZ - 2) \\
* - YS(K, 13) * WN(K, 2, JZ) - YS(K, 14) * WN(K, 2, JZ - 1) - YS(K, 15) * WN(K, \\
* 2, JZ - 2) - YS(K, 16) * WN(K, 1, JZ) - YS(K, 17) * WN(K, 1, JZ - 1) - YS(K, 18 \\
* ) * WN(K, 1, JZ - 2) \\
WN(K, 3, JZ) = YY - YS(K, 11) * WN(K, 3, JZ - 1) - YS(K, 12) * WN(K, 3, JZ - 2) \\
* - YS(K, 13) * WN(K, 2, JZ) - YS(K, 14) * WN(K, 2, JZ - 1) - YS(K, 15) * WN(K \\
* 2, JZ - 2) - YS(K, 16) * WN(K, 1, JZ) - YS(K, 17) * WN(K, 1, JZ - 1) - YS(K, \\
* 18) * WN(K, 1, JZ - 2) \\
XX = WN(K, 3, JZ) + YS(K, 2) * WN(K, 3, JZ - 1) + YS(K, 3) * WN(K, 3, JZ - 2) + Y \\
S(K, 4) * WN(K, 2, JZ) + YS(K, 5) * WN(K, 2, JZ - 1) + YS(K, 6) * WN(K, 2, JZ - \\
2) + YS(K, 7) * WN(K, 1, JZ) + YS(K, 8) * WN(K, 1, JZ - 1) + YS(K, 9) * WN(K, \\
* 1, JZ - 2) \\
58 YY = WM(K, 3, JZ) + YS(K, 2) * WM(K, 3, JZ - 1) + YS(K, 3) * WM(K, 3, JZ - 2) + Y \\
S(K, 4) * WM(K, 2, JZ) + YS(K, 5) * WM(K, 2, JZ - 1) + YS(K, 6) * WM(K, 2, JZ \\
- 2) + YS(K, 7) * WM(K, 1, JZ) + YS(K, 8) * WM(K, 1, JZ - 1) + YS(K, 9) * WM(K \\
* 1, JZ - 2) \\
C \]

222 IF(KF, EQ. 0) GO TO 44
JD = J + 1
DO 52 K = 1, KF

C

WN1(K, 2, JD) = XX - YF(K, 6) * WN1(K, 2, JD - 1) - YF(K, 7) * WN1(K, 1, JD) - \\
* YF(K, 8) * WN1(K, 1, JD - 1)
WN1(K, 2, JD) = YY - YF(K, 6) * WN1(K, 2, JD - 1) - YF(K, 7) * WN1(K, 1, JD) - \\
* YF(K, 8) * WN1(K, 1, JD - 1)
XX = WN1(K, 2, JD) + YF(K, 2) * WN1(K, 2, JD - 1) + YF(K, 3) * WN1(K, 1, JD) + \\
* YF(K, 4) * WN1(K, 1, JD - 1)
52 YY = WM1(K, 2, JD) + YF(K, 2) * WM1(K, 2, JD - 1) + YF(K, 3) * WM1(K, 1, JD) + \\
* YF(K, 4) * WM1(K, 1, JD - 1)

C

44 IX(IO, J) = XX
IY(NRA - IO, M2X1 - J) = YY
4 CONTINUE
IF(KS, EQ. 0) GO TO 444
DO 177 K = 1, KS
DO 177 L = 1, M2
LZ = L + 2
WN(K, 1, LZ) = WN(K, 2, LZ)
WN(K, 2, LZ) = WN(K, 3, LZ)
WM(K, 1, LZ) = WM(K, 2, LZ)
177 WN(K, 2, LZ) = WN(K, 3, LZ)
444 IF(KF, EQ. 0) GO TO 555
DO 28 K = 1, KF
DO 28 L = 1, M2
LZ = L + 1
WN1(K, 1, LZ) = WN1(K, 2, LZ)
28 WM1(K, 1, LZ) = WM1(K, 2, LZ)
555 CONTINUE
CALL FSEEK(2, KX1)
DO 3 IM = 1, NR
3 WRITE(2) (IX(IN, IN), IN = 1, M2)
REWIND 2
KB = IHS
DO 55 I = 1, M1X
KB = KB + IWS
CALL REMAP(0, KB, IWS, IER)
IF (IER .GE. 29) GO TO 3002
DO 56 K = 1, NR
  56 READ(2) (IX(K, L), L = 1, M2)
  DO 55 K = 1, NR
  DO 55 L = 1, M2
  55 IY(K, L) = IY(K, L) + IX(K, L)
REWIND 2
KB = -IWS

C
C WRITE BACK THE FILTERED IMAGE BACK
C TO THE ORIGINAL BLURRED IMAGE FILE
C
DO 57 I = 1, MX
  KB = KB + IWS
  CALL REMAP(0, KB, IWS, IER)
  IF (IER .GE. 29) GO TO 3002
  DO 57 K = 1, NR
  57 WRITE(2) (IY(K, L), L = 1, M2)
CLOSE 2
200 FORMAT(5A2)
STOP
3000 TYPE "ERROR IN VMEM ; ERROR # = ", IER
CLOSE 2
STOP
3001 TYPE "ERROR IN MAPDF ; ERROR # = ", IER
CLOSE 2
STOP
3002 TYPE "ERROR IN REMAP ; ERROR # = ", IER
CLOSE 2
STOP
END
DIMENSION YS(3, 18), YF(2, 8), NAME(5)

************** 2DRECAB **************

REMOVAL OF ATMOSPHERIC BLUR BY ZERO PHASE
IWO-DIMENSIONAL RECURSIVE DIGITAL FILTERING.
IMAGE SIZE SHOULD BE EQUAL TO ( M1 X M2 )
WHERE M1 & M2 ARE POWERS OF 2 AND GREATER
THAN 63 BUT LESS OR EQUAL TO 256. THE
DIMENSIONS OF YS, YF, WN, WN1 ARE AS FOLLOWS
YS(KS, 18), YF(KF, 8), WN(KS, 3, 258), WN1(KF, 2, 258)
KS, KF ARE THE # OF 2ND AND 1ST ORDER FILTER
SECTIONS RESPECTIVELY OF THE 2D FILTER. THE
DIMENSIONS OF THE OTHER ARRAYS ARE FIXED AND
THEY CAN HANDLE IMAGES OF SIZE UPTO 256 X 256
THE PROGRAM WRITES BACK THE RESTORED IMAGE TO
THE ORIGINAL BLURRED IMAGE FILE

DIMENSION WN(3, 3, 258), WN1(2, 2, 258)
DIMENSION IWIN(16384), IY(64, 256)
DOUBLE PRECISION YS, YS1, YF, YF1, A0, .B0-.B1
COMMON/BL/IWIN
EQUIVALENCE(IWIN, IY)
CALL YMEM(ICNT, IER)
IF(IER.EQ.5) GO TO 3000
TYPE"# OF FREE 1024-WORD BLOCKS= ", ICNT
IWS=16
CALL MAPDF(ICNT, IWIN, IWS, IER)
IF(IER.GE.5) GO TO 3001
TYPE"SIZE OF INPUT IMAGE"

READ IN THE SIZE OF IMAGE AND READ OUT THE
IMAGE DATA INTO THE EXTENDED MEMORY

ACCEPT"# OF COLOURS= ", M2
ACCEPT"# OF ROWS= ", M1
WRITE(10, 100)

100 FORMAT(' ', 1X, 'FILE NAME OF NOISY BLURRED IMAGE= ', Z)
READ(11, 200) (NAME(I), I=1, 5)
OPEN 2, NAME, LEN=2*M2, REC=M1
NR=64
NRA=NR+1
M1X=M1/NR
M2X1=M2+1
M2X2=M2+2
KBX=IWS+M1X
KB=-IWS
DO 80 I=1, M1X
KB=KB+IWS
CALL REMAP(0, KB, IWS, IER)
IF(IER.GE.29) GO TO 3002
DO 80 K=1, NR
80 READ(2) (IY(K, J), J=1, M2)
READ IN THE COEFFICIENT FILE NAME AND STORE
THE COEFFICIENTS IN THE COEFFICIENT ARRAYS
YS AND YF

WRITE(10,102)
READ(11,200) (NAME(I),I=1,5)
102 FORMAT(' ',1X,'FILE NAME OF COEFS = ',Z)
OPEN 4,NAME
READ(4) KS,KF
NT1=KS+19
NT2=KF+8
IF(KS.GE.1) READ(4) ((YS(K,L),L=1,18),K=1,KS)
IF(KF.GE.1) READ(4) ((YF(K,L),L=1,8),K=1,KF)
READ(4) A0
CLOSE 4

SET THE INITIAL CONDITIONS OF THE INTERMEDIATE
ARRAYS WN AND WNL TO ZERO AND FILTER THE IMAGE
BY THE MAGNITUDE-SQUARED 2D FILTER TRANSFER
FUNCTION.

DO 3 IJ=1,2
DO 234 NZ1=1,3
DO 234 NZ2=1,3
DO 234 NZ3=1,N2X2
234 WN(NZ1,NZ2,NZ3)=0. DO
DO 456 NZ1=1,2
DO 456 NZ2=1,2
DO 456 NZ3=1,N2X1
456 WNL(NZ1,NZ2,NZ3)=0. DO
KB=KBX
IF(IJ.EQ.1) KB=-IWS
DO 3 I=1,M1X
IF(IJ.EQ.2) GO TO 778
KB=KB+IWS
CALL REMAP(0,KB,IWS,IER)
IF(IER.GE.29) GO TO 3002
GO TO 779
778 KB=KB-IWS
CALL REMAP(0,KB,IWS,IER)
IF(IER.GE.29) GO TO 3002
779 DO 3 IO=1,NR
DO 4 J=1,M2
IF(IJ.EQ.2) GO TO 780
XX=IY(IO,J)*A0
GO TO 781
780 XX=IY(NRA-IO,M2X1-J)*A0
781 IF(KS.EQ.0) GO TO 222
JZ=J+2
DO 50 K=1,KS

WN(K,3,JZ)=XX*YS(K,11)*WN(K,3, JZ-1)-YS(K,12)*WN(K,3, JZ-2)
*YS(K,13)*WN(K,2,JZ)-YS(K,14)*WN(K,2, JZ-1)-YS(K,15)*WN(K,
*2, JZ-2) - YS(K, 16) * WN(K, 1, JZ)-YS(K, 17) * WN(K, 1, JZ-1) - YS(K, 18) * WN(K, 1, JZ-2)
50 XX = WN(K, 3, JZ) - YS(K, 2) * WN(K, 3, JZ-1) + YS(K, 3) * WN(K, 3, JZ-2) + YS(K, 4) * WN(K, 2, JZ) + YS(K, 5) * WN(K, 2, JZ-1) + YS(K, 6) * WN(K, 2, JZ-2) + YS(K, 7) * WN(K, 1, JZ) + YS(K, 8) * WN(K, 1, JZ-1) + YS(K, 9) * WN(K, 1, JZ-2)

C
222 IF(KF, EQ. 0) GO TO 44
   JD = J + 1
   DO 52 K = 1, KF
C
52 WN1(K, 2, JD) = XX - YF(K, 6) * WN1(K, 2, JD-1) - YF(K, 7) * WN1(K, 1, JD) - YF(K, 8) * WN1(K, 1, JD-1)

C
50 XX = WN1(K, 2, JD) + YF(K, 2) * WN1(K, 2, JD-1) + YF(K, 3) * WN1(K, 1, JD) + YF(K, 4) * WN1(K, 1, JD-1)

C
44 IF(IJ, EQ. 2) GO TO 556
   IY(ID, J) = XX
   GO TO 4
556 IY(NRA-10, M2X1-J) = XX

4 CONTINUE
   IF(KS, EQ. 0) GO TO 444
   DO 177 K = 1, KS
   DO 177 L = 1, M2
   L2 = L + 2
   WN(K, 1, L2) = WN(K, 2, L2)
177 WN(K, 2, L2) = WN(K, 3, L2)

444 IF(KF, EQ. 0) GO TO 3
   DO 23 K = 1, KF
   DO 23 L = 1, M2
   L2 = L + 1
23 WN1(K, 1, L2) = WN1(K, 2, L2)

3 CONTINUE
   REWIND 2
   KB = - IWS

C
WRITE THE DEBLURRED IMAGE BACK INTO THE ORIGINAL BLURRED IMAGE FILE

C
DO 57 I = 1, M1X
   KB = KB + IWS
   CALL REMAP(0, KB, IWS, IER)
   IF(IER, GE. 29) GO TO 3002
   DO 57 K = 1, NR
57 WRITE(2) (IY(K, L), L = 1, M2)
   CLOSE 2
200 FORMAT (5A2)
   STOP
3000 TYPE "ERROR IN VMEM ; ERROR # =", IER
   CLOSE 2
   STOP
3001 TYPE "ERROR IN MAPDF ; ERROR # =", IER
   CLOSE 2
   STOP
3002 TYPE "ERROR IN REMAP ; ERROR # =", IER
   CLOSE 2
   STOP
## VITA AUCTORIS

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1951</td>
<td>Born on 7th of December, 1951, Coorg District, Mysore State, INDIA.</td>
</tr>
<tr>
<td>1966</td>
<td>Completed high school at St. Anne's Higher Secondary School, Virajpet, Coorg, Mysore State, INDIA.</td>
</tr>
<tr>
<td>1967</td>
<td>Completed Pre-University Course, University of Madras, Madras, INDIA.</td>
</tr>
<tr>
<td>1972</td>
<td>Graduated from University of Madras, with a B.E. (Bachelor of Engineering) degree in Electronics and Communications Engineering.</td>
</tr>
<tr>
<td>1974</td>
<td>Graduated from University of Windsor, with the degree of M.A.Sc. in Electrical Engineering.</td>
</tr>
<tr>
<td>1979</td>
<td>Candidate for the degree of Doctor of Philosophy in Electrical Engineering. University of Windsor, Windsor, Ontario, Canada.</td>
</tr>
</tbody>
</table>