Fall 12-2010

BEMDEC: An Adaptive and Robust Methodology for Digital Image Feature Extraction

Isaac Kueth Gang
University of Southern Mississippi

The University of Southern Mississippi

BEMDEC: AN ADAPTIVE AND ROBUST METHODOLOGY FOR DIGITAL IMAGE FEATURE EXTRACTION

by

Isaac Kueth Gang

Abstract of a Dissertation
Submitted to the Graduate School
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

December 2010

ABSTRACT

BEMDEC: AN ADAPTIVE AND ROBUST METHODOLOGY FOR DIGITAL IMAGE FEATURE EXTRACTION

by Isaac Kueth Gang

December 2010

The intriguing study of feature extraction, and edge detection in particular, has, as a result of the increased use of imagery, drawn even more attention not just from the field of computer science but also from a variety of scientific fields. However, various challenges surrounding the formulation of feature extraction operator, particularly of edges, which is capable of satisfying the necessary properties of low probability of error (i.e., failure of marking true edges), accuracy, and consistent response to a single edge, continue to persist. Moreover, it should be pointed out that most of the work in the area of feature extraction has been focused on improving many of the existing approaches rather than devising or adopting new ones. In the image processing subfield, where the needs constantly change, we must equally change the way we think.

In this digital world where the use of images, for variety of purposes, continues to increase, researchers, if they are serious about addressing the aforementioned limitations, must be able to think outside the box and step away from the usual in order to overcome these challenges. In this dissertation, we propose an adaptive and robust, yet simple, digital image features detection methodology using bidimensional empirical mode decomposition (BEMD), a sifting process that decomposes a signal into its two-dimensional (2D) bidimensional intrinsic mode functions (BIMFs). The method is further extended to detect corners and curves, and as such, dubbed as BEMDEC, indicating its ability to detect edges, corners and curves. In addition to the application of BEMD, a unique combination of a flexible envelope estimation algorithm, stopping criteria and boundary adjustment made the realization of this multi-feature detector possible. Further application of two morphological operators of binarization and thinning adds to the quality of the operator.

The University of Southern Mississippi

BEMDEC: AN ADAPTIVE AND ROBUST METHODOLOGY FOR DIGITAL IMAGE FEATURE EXTRACTION

by

Isaac Kueth Gang

A Dissertation
Submitted to the Graduate School
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

December 2010

DEDICATION

TO MY FAMILY

ACKNOWLEDGMENTS

When I embarked on this journey of PhD pursuit, many questions, not the least of which was who I would turn to when the going gets rough, were lingering in my mind. By the end of the first year, it became clear that most of my worries were nonfactor. This was, in large part, due to the generosity, support and friendship of many people whose names may not appear here due to their number coupled with institutional space restrictions. But to them, I say thank you and you know who you are. Furthermore, my dissertation committee, is, in my opinion, the best that has ever been assembled. Their doors were always opened and I can honestly say that if I did not walk through those doors, it was always my personal choice not that they were not ready for me. Their sincerity in dealing with me and addressing research issues certainly made me better. They truly are the epitome of mentorship and leadership. For that I cannot thank them enough.

My committee chair and dissertation advisor, Dr. Dia Ali, has not only been the best advisor and mentor that anyone could have asked for, but he has been a major influence in my life and throughout my graduate studies here at The University of Southern Mississippi. He stuck with and continued to believe in me even when things did not go well, and for that, words are insufficient to express my gratitude for his guidance, support and encouragement. My committee members were superb with each bringing unique quality to the table. Dr. Jean Gourd of Louisiana Tech University has helped me greatly in shaping the direction of my research. His positive critiques helped make the final product what it is. Dr. Joe Zhang has been instrumental in asking the right research questions and offering constructive suggestions in an effort to improve my work. I came to appreciate his humility and leadership over the years both as my committee member and as the director of the School of Computing at The University of Southern Mississippi. I cannot thank Dr. Ray Seyfarth enough, first for accepting my late request to complete my committee and for being a positive force. His insights, views and sense of humor were very valuable to me both as a researcher and a person. Dr. Ras Pandey exemplified a rare image of optimism while managing to be very inquisitive at the same time. His positive reinforcement certainly made me see the light. I want to thank several of my colleagues whose involvement was indispensable in the successful completion of this work. Dr. Paulus Wahjudi of Marshal University, Dr. Jeremy Kackley of Noetic Strategies, Dr. Jim Jacobs of Jackson State, and Daniel Bond provided very helpful insights. Finally, I want to thank Dr. Joseph Kolibal, for without the utilization of his Latex Packages, this manuscript would be impossible.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure

LIST OF TABLES

LIST OF ABBREVIATIONS

2D	-	2-Dimensional
ACIS	-	Adaptive Color Image Segmentation
BEMD	-	Bidimensional Empirical Mode Decomposition
BEMDEC	-	BEMD for Edges, Corners and curves
BEMF	-	Bidimensional Empirical Mode Function
BIMF	-	Bidimensional IMF
BR	-	Bidimensional Residue
CCD	-	Charge-Coupled Device
CMG	-	Color Morphological Gradient
CS	-	Cubic Spline
CSS	-	Compactly Supported Spline
CT	-	Computer Tomography
DEM	-	Digital Elevation Model
DEMD	-	Directional EMD
DIM	-	Distance-based Interpolation method
DR	-	Deviation Ratio
DRT	-	Deviation Ratio Threshold
EA	-	Extended Averaging
EC	-	Edges and Corners
ED	-	Edge Direction
EM	-	Edge Magnitude
EMD	-	Empirical Mode Decomposition
EMD	-	Earth Mover’s Distance

EOF	-	Orthogonal Function Expansion
EPGF	-	Edge-Preserving Gaussian Filter
ES	-	Exponential Spline
ET	-	Extrema Threshold
FFT	-	Fast Fourier Transform
FNN	-	Fuzzy Neural Network
GIS	-	Geographic Information System
GP	-	Gradient Projection
GS	-	Gaussian Spline
IBR	-	Intermediate Bidimensional Residue
IEMD	-	Image EMD
IM	-	Inverse Multiquadrics
IMF	-	Intrinsic Mode Function
ISBIMF	-	Intermediate State BIMF
ISIMF	-	Intermediate State IMF
L	-	Linear
LCM	-	Linear Combination Model
LE	-	Lower Envelope
LIM	-	Linear Interpolation Method
LSC	-	Least Squares Collocation
LSM	-	Low-order Surface model
LVG	-	Lattice Vector Quantization
M	-	Multiquadrics
ME	-	Mean Envelope
MNAI	-	Maximum Number of Allowable Iterations
MSE	-	Mean Squared Error
$\mathbb{N}$	-	Notation ’for all natural numbers’
NEP	-	Number of Extrema Points

NMR	-	Nuclear Magnetic Resonance
NWM	-	Neighboring Window Method
OI	-	Orthogonality Index
OTF	-	Optical Transfer Function
PSF	-	Point Spread Function
R	-	Residue
RBF	-	Radial Basis Function
RCMG	-	Robust Color Morphological Gradient
RMS	-	Root-Mean-Squared
ROC	-	Receiver Operating Characteristic
ROF	-	Rudin, Osher and Fatemi
SNR	-	Signal-to-Noise Ratio
SP	-	Sifting Process
SPIHT	-	Set Partitioning In Hierarchical Trees
TPS	-	Thin-Plate Spline
TPS	-	Thin Plate Splines
TV	-	Total Variation
UE	-	Upper Envelope
USC	-	University of Southern California
VOS	-	Vector Order Statistics
$\mathbb{Z}$	-	Notation ’for all integers’

NOTATION AND GLOSSARY

General Usage and Terminology

The notation used in this text represents fairly standard mathematical and computational usage. In many cases these fields tend to use different preferred notation to indicate the same concept, and these have been reconciled to the extent possible, given the interdisciplinary nature of the material. In particular, the notation for partial derivatives varies extensively, and the notation used is chosen for stylistic convenience based on the application. While it would be convenient to utilize a standard nomenclature for this important symbol, the many alternatives currently in the published literature will continue to be utilized.

The blackboard fonts are used to denote standard sets of numbers: $\mathbb{R}$ for the field of real numbers, $\mathbb{C}$ for the complex field, $\mathbb{Z}$ for the integers, and $\mathbb{Q}$ for the rationals. The capital letters, $A, B, \cdots$ are used to denote matrices, including capital greek letters, e.g., Λ for a diagnonal matrix. Functions which are denoted in boldface type typically represent vector valued functions, and real valued functions usually are set in lower case roman or greek letters. Caligraphic letters, e.g., $\mathcal{V}$, are used to denote spaces such as $\mathcal{V}$ denoting a vector space, $\mathcal{H}$ denoting a Hilbert space, or $\mathcal{F}$ denoting a general function space. Lower case letters such as i, j, k, l, m, n and sometimes p and d are used to denote indices.

Vectors are typset in square brackets, e.g., $[\cdot]$, and matrices are typeset in parenthesese, e.g., $(\cdot)$. In general the norms are typeset using double pairs of lines, e.g., $||\cdot||$, and the abolute value of numbers is denoted using a single pairs of lines, e.g., $|\cdot|$. Single pairs of lines around matrices indicates the determinant of the matrix.

Furthermore, the use of the letter R in the text is distinctively for Residue and shall not be confused with matrices representation or $\mathbb{R}$ representing the field of real numbers.

Chapter 1

INTRODUCTION

1.1 Brief Overview of Edges and Features Extraction

Setting out to define a methodology for detecting an edge or feature of an object is not computer scientists' attempt to be different or aloof. In fact, this process is significant in almost every aspect of our daily lives and is a part and parcel of a bigger picture of our existence and interaction with science and nature. In that sense, it is increasingly becoming popular in many fields of study [1, 2, 3]. Through history, humans have always looked to science for the purpose of trying to understand certain natural phenomena. The reverse is also true, for they also look to nature when they are trying to understand scientific phenomena. Whether it is Charles Darwin's "natural selection," which he described as "preservation of favored races in the struggle for life" in his book entitled *On the Origin of Species* and later described as "survival of the fittest" by Herbert Spencer, or John Holland's Genetic Algorithm which emulates biological evolution in the computer, nature and science have proven to be complementary pair.

Backed by scientific evidence, it is believed that the human species is the smartest of all species on earth. Psychologists, sociologists, physiologists, biologists and computer scientists alike have all announced their verdicts. The human brain is of particular significance in this distinction. Equipped with thousands of neurons, making a decision is a trivial matter in comparison to the computer. Though computers make decisions faster than humans, they first go through a rigorous, explicit and detailed process to make even the simplest decision, a process that would take little to no effort for human. Our recognition system is particularly powerful because we can recognize an enormous range of objects, involving unrehearsed detection of edges and other features, with little difficulty [1]. As such, physiological theorists believe that human vision system (HVS) go through some sort of edge and feature detection prior to recognizing a color or an image intensity [4, 5].

In essence, computer scientists believe that some edge and feature detection is required prior to image interpretation for an automatic computer system. As a result, finding edges and other features in an image is considered an important process in many artificial vision systems and the source of motivation for researchers in this area. The area of edge detection and its subsets have been around for quite sometime. Nonetheless, as images have no real edges as we know them in the real world, the term edge, in reference to image, is somewhat misplaced. What we describe as edges are actually abrupt changes of intensity in image. Because the overall goal is to locate edges in the real world through the image however, the term edge detection has gained general acceptance in the research community [6, 7]. Indeed, edges characterize boundaries and, as such, have fundamental importance in image processing [8].

The main reason for detecting sharp changes in image brightness is to capture important events and changes in properties of the world. In fact, it can be shown that under rather general assumptions for an image formation model, discontinuities in image brightness are likely to correspond to discontinuity in depth and surface orientation, changes in material properties, and variations in scene illumination [9, 10]. In the ideal case, the result of applying an edge detector to an image may lead to a set of connected curves that indicate the boundaries of objects, the boundaries of surface markings as well as curves that correspond to discontinuities in surface orientation. For this reason, applying an edge or corner detector to an image may significantly reduce the amount of data to be processed and may therefore filter out information that may be regarded as less relevant, while preserving the important structural properties of an image [7, 11]. If the edge detection step is successful, the subsequent task of interpreting the information contents in the original image may therefore be substantially simplified [9, 10, 7, 11, 12].

1.2 Edge Properties

Generally speaking, edges extracted from a two-dimensional image of a three-dimensional scene can be classified as either viewpoint dependent or viewpoint independent. A viewpoint independent edge typically reflects inherent properties of the three-dimensional objects, such as surface markings and surface shape [8]. A viewpoint dependent edge may change as the viewpoint changes, and normally reflects the geometry of the scene, such as objects moving toward one another. A typical edge might, for example, be the border between a block of red color and a block of yellow. In contrast, a line, typically extracted by a ridge detector, can be a small number of pixels of a different color on an otherwise unchanging background. For a line, there may, therefore, usually be one edge on each side of the line [1].

Edges play quite a significant role in many applications of image processing, in particular for machine vision systems that analyze scenes of man-made objects under controlled illumination conditions. In recent years, however, substantial and perhaps successful research has also been made on computer vision methods that use various image operations in addition to edge detection as a pre-processing step [13, 14].

1.3 Motivation

With the ever increasing utilization of imagery in our daily lives and in many real world applications, it is often necessary to take a given image and perform some analysis in order to improve its quality, gain useful information from it, identify its texture, or extract desired feature. These operations add even greater importance in areas such as medical imaging, aerial survey, remote sensing and military planning. Whether it is performing a Magnetic Resonance Imaging (MRI) on a patient, surveying a town or a city, analyzing images of the enemy position, every detail is of utmost importance.

Despite having made strides with algorithms development in recent years, certain limitations continue to hamper the development of application software that can effectively handles many of the aforementioned tasks. This is particularly true with feature detection algorithms where inflexible real world assumptions have to be made in order to realize many of the existing feature detection methodologies. Although research effort has been made to circumvent some of the underlying issues, it is usually at the expense of quality and computational efficiency. However, it is often the case that a reasonable feature detection methodology is developed but is only able to extract one given feature. While this is a splendid endeavor, it is often desirable to have a feature detector that has the ability to extract more than one feature at once. The need for such a feature motivates the formulation of the methodology proposed herein.

This new methodology, derived from Bidimensional Empirical Mode Decomposition (BEMD), incorporates the ability to detect edges, corners and curve and is called Bidimensional Empirical Mode Decomposition for edges, corners, and curves (hereby referred to as BEMDEC) and is introduced in chapter 7.

1.4 Dissertation Overview

The aim of this research is to introduce and formulate a new and distinctive methodology to analyze and extract features of a digital image. This means analyzing the given image is expected to not only be adaptive but also robust as it is computationally efficient. Because of the computational complexities introduced by many of the existing schemes, development of a detection operator that is computationally efficient and of high quality has been a challenge, if not entirely impossible.

The combination of the bidimensional empirical mode decomposition (BEMD) with a robust surface interpolation algorithm, known as the cubic spline (CS), in a unique fashion, in this dissertation, resulted in the formulation of BEMDEC, a new and dynamic image analysis technique for various digital images.

1.5 Dissertation Contribution

The formulation of the proposed BEMDEC methodology introduces new and unique outlook into the area of edge and feature extraction/detection in general. Although some attempts have been made to circumvent the difficulties surrounding edge detection and other features extraction, successful scheme that combines low cost with high quality is yet to be attained. The BEMDEC method is forumulated to introduce a robust and adaptive decomposition technique, and efficient envelope estimation method, both of which will make it possible to analyze digital images and extract their features efficiently. BEMDEC attempts to achieve this as followed:

1. A complete analysis of the image including the detection and smoothing of edges and corners is performed and is extended by addition of the bidimensional empirical mode decomposition. The introduction of BEMD allows the decomposition of the image into its bidimensional empirical mode functions (BEMFs). This is, in and of itself, a unique procedure in terms of edge map analysis and production of various images. Furthermore, supplemental corners and curves detection techniques are introduced.

2. An efficient algorithm that extends the classical bidimensional empirical mode decomposition, which allows the manipulation of the image data is introduced. This is achieved by combining the BEMD method with radial basis function interpolation method known as the cubic spline (CS).

3. Finally, a comprehensive and efficient image analysis is performed using the proposed method. Furthermore, edge detection and feature extraction is carried out using a grayscale image. With the incorporation of the interpolation algorithm into the BEMD method, the BEMDEC (to detect edges, corners and curves - EC) methodology is able and is extended to detect corners and curves. This extension makes BEMDEC very versatile in comparison to other features detectors.

1.6 Dissertation Organization

The rest of the dissertation is organized as followed. In Chapter 2, the edge detection along with other image operations are explained with edge detection being discussed in more details. In Chapter 3, an in-depth literature review on the edge and boundary detection and extraction is presented, while some existing works as well as challenges, in color image detection and analysis are also presented in that chapter. Chapter 4 presents the available surface interpolation algorithms, including the cubic spline algorithm used in this dissertation, and their role in image edge detection/extraction. This is meant for completeness. Chapter 5 presents the classical empirical mode decomposition (EMD) in reasonable details. A detailed description of its extension to two dimensional data, the bidimensional empirical mode decomposition (BEMD), used in this dissertation, is given in Chapter 6. Chapter 7 presents the BEMDEC method proposed in this dissertation and Chapter 8 concludes the dissertation by presenting the experimental results and future direction. Apendices A-D respectively present brief summaries of Fourier, Wavelet, Gabor and Gabor-Wigner transforms.

1.7 Summary

Successful development of robust edge and corner detection operators and/or edge maps could energize edge and corner detection research because of the uniqueness of the various methods involved. Many existing methods try to achieve either edge, corner or curve detection, but no method, to the author's knowledge has demonstrated the ability to do both simultaneously. Thus, an approach such as BEMDEC can greatly benefit the research community, and is therefore worth pursuing and improving, especially in the context of multi feature detection.

Chapter 2

EDGE DETECTION AND OTHER IMAGE OPERATIONS

2.1 Introduction

Edge detection, as mentioned previously, is a fundamental problem and is therefore important in the area of image processing in general and computer vision in particular. Nonetheless, it is not the only image processing operation in the area. For this reason, a brief summary and discussion of other operations is necessary at this point in order to undertand edge detection in relation to them. Below, edge detection is reviewed and image denoising, restoration, registration and deblurring are briefly presented and discussed.

2.2 Edge Detection

Edge detection plays a major role in the overall image processing scheme and for this reason, it continues to be a major area of research for quite sometimes.

2.2.1 How Edges Are Detected

As mentioned above, detecting edges is a fundamental problem, for it allows us to gain important information from the image or object whose edges we are detecting. In addition, it reduces the amount of data and filters out needless information while preserving the important structural properties of the image. As a result, numerous methods to achieve this goal have been proposed in the literature [3, 1, 15]. Many of the existing edge detection methods can be categorized as either gradient or Laplacian. Gradient methods detect the edge by looking at the minimum and maximum values in the first derivative of the image whereas the Laplacian methods search for zero-crossing in the second derivative of the image in order to find edges. This is necesitated by the fact that an edge has the one-dimensional shape of a ramp and calculating the derivative of the image can highlight its location [16, 14, 17].

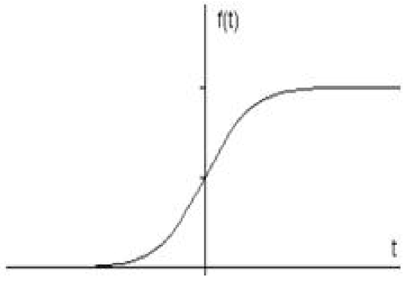


Figure 2.1: First derivative with respect to t.

Fig. 2.1 shows a gradient of a signal, which is simply the first derivative with respect to t. If we take this signal, we can observe the properties of the gradient filter family of edge detectors [8, 1, 18, 19]. In fact, the resulting behavior (Fig. 2.2) demonstrates the idea of the maxima. Looking at this figure, it is clear that the maximum is located at the center of the edge in the original signal.

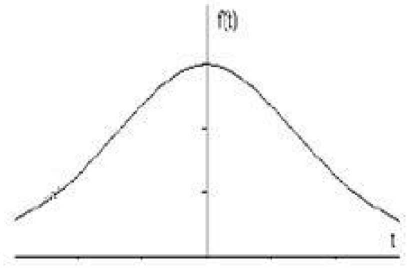


Figure 2.2: Resulting signal showing the maximum.

Gradient filters accomplish their edge detecting task by looking at every pixel in the image and using a threshold. More specifically, a pixel location is declared an edge location if the value of the gradient exceeds some threshold. Because edges, by definition, are abrupt changes in the image intensity, they will have higher pixel intensity values than non edges. Hence, by setting the threshold, this threshold can be compared to the gradient value where, as a result, an edge can be detected anytime the gradient value is greater than the threshold [13, 20, 21].

Furthermore, when the first derivative is at a maximum - the situation we have in Fig. 2.2, the second derivative is zero. As a result, another alternative to finding the location of an edge is to locate the zeros in the second derivative. This process is called the Laplacian method and characterizes all the Laplacian family of edge detectors [22, 13, 14]. Fig. 2.3 illustrates the zero-crossing in the second derivative.

Typically, any edge detection method is characterized by the two fundamental questions classical edge detection attempts to answer [13, 14].

- What scene properties give rise to abrupt changes in intensity?
- What do these abrupt changes in intensity look like, mathematically?

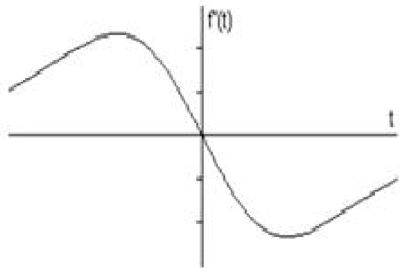


Figure 2.3: Zero-crossing in the second derivative.

2.2.2 Mathematical Notations for the Methods Dicussed

Along with smoothing, taking one or more spatial derivatives of an image is not only fundamental to edge detection but to the general image processing. We have seen that the gradient methods look for the minimum and maximum values in the first derivative in order to detect edges [16]. It is to be noted that taking the derivative of an image is a challenge due to the strict mathematical definition of the derivative [3]. This is due, in part, to the fact that a digitized image is not a continuous function *a(x, y)* of the spatial variable but rather a discrete function *a[m, n]* of the integer spatial coordinates [1]. It is, therefore, necessary to look at algorithms involving derivatives for detecting edges as approximations of the true spatial derivatives of the original spatially-continuous image [13].

2.2.3 The First Derivatives

Since an image is a function of two (or more) variables, it is necessary to define the direction in which the derivative is taken. For the two-dimensional case, we have the horizontal direction, the vertical direction, or an arbitrary direction which can be considered as a combination of the two [14]. Using h_x to denote a horizontal derivative filter (matrix), h_y to denote a vertical derivative filter (matrix), and h_θ to denote the arbitrary angle derivative filter (matrix), then:

$$[h_\theta] = cos\theta \bullet [h_x] + sin\theta \bullet [h_y] \tag{2.1}$$

We can, therefore, generate a vector derivative description as the gradient, $\nabla a[m,n]$ of an image:

$$\nabla a = \frac{\delta a}{\delta x}\overrightarrow{i_x} + \frac{\delta a}{\delta y}\overrightarrow{i_y} = (h_x \otimes a)\,\overrightarrow{i_x} + (h_y \otimes a)\,\overrightarrow{i_y}, \tag{2.2}$$

where $\overrightarrow{i_x}$ and $\overrightarrow{i_y}$ are are unit vectors in the horizontal and vertical direction, respectively [14]. As a result, we have two descriptions namely, gradient magnitude and gradient direction. The gradient magnitude is given by the following:

$$|\nabla a| = \sqrt{(h_x \otimes a)^2 + (h_y \otimes a)^2} \tag{2.3}$$

whereas the gradient direction is given by the following

$$\psi(\nabla a) = arctan\left\{(h_y \otimes a)\,/\,(h_x \otimes a)\right\} \tag{2.4}$$

It is sometimes necessary to approximate the gradient magnitude as followed:

$$|\nabla a| \cong |(h_x \otimes a| + |(h_y \otimes a| \tag{2.5}$$

It is worth mentioning here that the final results of the above calculations greatly depend on the choices of h_x and h_y [14, 6, 7].

2.3 Denoising

Digital images are prone to a various types of noise. Noise is, by and large, the result of errors in the image acquisition process that result in pixel values that do not reflect the true intensities of the real scene. There are several ways that noise can be introduced into an image, depending on how the image is created [23]. For example, if the image is scanned from a photograph made on film, the film grain is a source of noise. Also, noise can be the result of damage to the film, or be introduced by the scanner itself. Furthermore, if the image is acquired directly in a digital format, the mechanism for gathering the data, such as a charge-coupled device (CCD) detector, can introduce noise. Electronic transmission of image data can also introduce noise [24].

In the area of image processing, a good portion of the processing is devoted to image restoration. Algorithm development and goal oriented image processing are in the heart of the ongoing research in this direction. Image restoration is defined as the removal or reduction of degradations that are incurred during the image attainment process. Degradation comes from blurring as well as noise due to electronic and photometric sources [25]. In addition to the blurring effect, an image can also be corrupted by noise. As previously mentioned, a noise is introduced in the transmission medium due to a noisy channel, errors during the measurement process and quantization of the data for storage. Upon realization that an image is degraded after it is obtained, it is necessary that we improve it by applying a denoising technique of some sort before it is printed [26].

However, a successful application of any denoising technique requires us to know something about the degradation process in order to simulate or develop a model for it. Once we have a model for the degradation process, the inverse process can be applied to the image to restore it back to its original form [27]. Image denoising finds applications in fields such as astronomy where the resolution limitations are severe, in medical imaging where the physical requirements for high quality imaging are needed for analyzing images of unique events, and in forensic science where potentially useful photographic evidence is sometimes of extremely low quality [23, 26].

To help in the subsequent discussions, it should be mentioned that the most commonly used images are binary, grayscale and color [23]. Binary images are the simplest type of images, for they can take only two discrete values of black and white, where black is denoted by a 0 and white by a 1 [26]. Also note that a binary image is usually created from a grayscale image and it finds applications in computer vision areas where the general shape or outline information of the image is needed. These types of images are sometimes referred to as 1 bit/pixel images [26].

Furthermore, grayscale images are known as monochrome or one-color images, and are, generally speaking, the most commonly used type in many image analysis algorithms and processing tasks. This is because they contain no color information and manipulating them is not computationally complex. This category of images contains 8 bits/pixel data, meaning it can have up to 256 $(0-255)$ different brightness levels where a 0 denotes black and 255 denotes white. Values between them (i.e., 1 to 254) represent the different gray levels, and because they contain the intensity information, they are also referred to as intensity images [23].

Finally, color images are considered as three band monochrome images, with each band representing a different color. Each band provides the brightness information of the corresponding spectral band. Typical color images are red, green and blue images and are most commonly known as RGB images. This is a 24 bits/pixel image [26, 23].

To illustrate the denoising concept, let us take a digital image as an example. A 2-dimensional digital image can be represented as a 2-dimensional array of data $s(x,y)$, where (x,y) represent the pixel location. The pixel value corresponds to the brightness of the image at location (x,y) [23, 26]. In case of image denoising methods, the characteristics of the degrading system and the noises are assumed to be known beforehand. The image $s(x,y)$ is blurred by a linear operation and noise $n(x,y)$ is added to form the degraded image $w(x,y)$. This is convolved with the restoration procedure $g(x,y)$ to produce the restored image $z(x,y)$.

The Linear operation shown in Fig. 2.4 is the addition or multiplication of the noise $n(x,y)$ to the signal $s(x,y)$. Once the corrupted image $w(x,y)$ is obtained, it is subjected to the denoising technique to get the denoised image $z(x,y)$ [23].

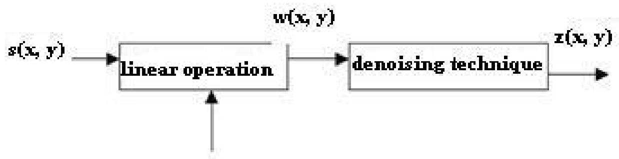


Figure 2.4: Illustration of the denoising concept.

The concept of denoising, as is the case with the subsequent image operations, is introduced here for the purpose of completeness, and even though there are various methods to help restore an image from noisy distortions, they will not be discussed here in any reasonable details. It suffices to note, however, that selecting the appropriate method plays a major role in getting the desired image. This is because the denoising methods tend to be problem specific. For example, a method that is used to denoise satellite images may not be suitable for denoising medical images and vice versa [23, 26]. Three broad techniques are available to remove or reduce noise. These are filtering, wavelet analysis, and multifractal analysis. Each technique has its advantages and disadvantages. Denoising by wavelets and multifractal analysis are some of the recent approaches. Wavelet techniques consider thresholding while multifractal analysis is based on improving the Hölder regularity of the corrupted image.

Furthermore, any denoising algorithm has to identify the type of noise it is attempting to remove or reduce. The following are the most commom types of noise [26]: Gaussian noise (truly distributed over the signal); Salt and Pepper noise (impulse or intensity spike); Speckle noise (multiplicative), and Brownian noise (fractal).

2.4 Restoration

Restoration is a very broad image operation that actually includes various operations, such as denoising and deblurring. As defined previously, it is the removal or reduction of degradations that are incurred during the image attainment process. Through restoration, low quality images can be restored to their original forms [26, 28].

As a result, various techniques and methodologies that stretch the boundary of the overall image improvement exist. For example, a shape-based landmark matching approach to optimize conformal parameterization of cortical surfaces was introduced by Lui et al. [29]. In this work, a meaningful parameterization of the cortical surface utilizing prior anatomical information in the form of anatomical landmarks, such as a curve, is proposed. The basic premise is that the computed maps are guaranteed to give a shape-based diffeomorphism, or invertible function, between the landmark curves. Furthermore, distorted documents, resulting from geometric distortion and non-uniform illumination, require restoration but no particular technique provides a silver bullet solution to this type of degradation. In the literature, various techniques are suggested. Brown et al. [30], for instance, proposed conformal mapping to rectify geometric distortion. In this work, a 3D surface is mapped back to the plane in order to correct the specified geometric distortion while minimizing angular distortion. The framework basically restores the 2D contents printed on the document in the presence of the geometric distortion and and non-uniform illumination [30].

In addition, various restoration models, most popular of which are total variation (TV) based, introduced by Rudin, Osher and Fatemi (ROF) in 1992, are proving to be useful [31, 32, 33]. These models aim at preserving sharp discontinuities, or edges in an image while removing noise and other unwanted fine-scale details. One of these models is the gradient projection (GP) algorithms applied to dual formulation in [31] where it is tested with different step length selection and line search strategies. Also, fuzzy techniques for digital image restoration are gaining momentum. In particular, fuzzy rank selection, fuzzy weighted, fuzzy neural network (FNN) and soft-switching are finding greater application [34]. Exposure-splitting, nonlinear technique for video scratching, posterior, and distribution have also produced some useful image restoration results [35, 36, 37, 38].

2.5 Registration

Registration is an image operation whose aim is to align two or more images of the same scene. In this process, the base or reference image is compared to the input image with the objective of bringing the input image into the alignment with the base image. The process involves application of spatial transformation to the input image. Generally speaking, the differences between the input image and the output image may occur as a result of terrain relief and other changes in perspective when shooting the same scene from different locations or viewpoints. In addition, lens, other internal sensor distortions, or differences between sensors and sensor types, can also cause distortion that necessitates the registration routine [26, 39].

Registration is one of the most important image analysis tasks available is in the area of image processing and computer vision, for it allows us to gain useful information from various data sources [39]. In general, registration is required in remote sensing where multispectral classification, environment monitoring, change detection, image mosaicing, weather forecasting, creation of super-resolution images, and integration of information into geographic information system (GIS), are desired [40, 39]. It is also useful in medicine where it makes it possible to combine computer tomography (CT) and nuclear magnetic resonance (NMR) data in order to obtain more complete information about the patient. Additionally, it helps in monitoring tumor growth, treatment verification, and comparison of the patient's data with anatomical atlases. Cartography (map updating) and computer vision also make heavy use of the registration operation [41].

Application of the registration task can be categorized into four main categories depending on the manner in which the image is acquired [41, 39].

- *Different viewpoints (multiview analysis)*: In this case, images of the same scene are acquired at different times, often on regular basis, and possibly under different conditions. The goal is to find and evaluate changes in the scene which appeared between the consecutive image acquisitions. Mosaicing images of a surveyed area in remote sensing and shape recovery in computer vision are some examples of this type of analysis.

- *Different sensors (multimodal analysis)*: In this case, images of the same scene are acquired by different sensors. The goal is to integrate the information obtained from different source streams to gain more complex and detailed scene representation. Recording the anatomical body structure like magnetic resonance image (MRI), ultrasound or CT scan in medical imaging are just a few examples of this analysis.

- *Different time (multitemporal analysis)*: In this case, images of the same scene are acquired at different times, often on regular basis, and possibly under different conditions. The goal is to find and evaluate changes in the scene which appeared between the consecutive images acquisitions. Monitoring of global land usage and landscape planning in remote sensing, to mention just a few, are some application areas of this analysis.

- *Sense to model registration*: In this case, images of a scene and a model of the scene are registered. The model can be a computer representation of the scene, for instance maps or digital elevation models (DEM) in GIS, another scene with similar content (another patient), "average" specimen, etc. The aim is to localize the acquired image in the scene/model and/or to compare them. Target template matching with real-time images and automatic quality inspection in computer vision are some example applications.

Because of the diversity of images to be registered and due to various types of degradations, it is unfeasible to design a universal method applicable to all registration tasks. For this reason, every method should take into account not only the assumed type of geometric deformation between the images but also radiometric deformations and noise corruption. Similarly, every method should require registration accuracy and application-dependent data characteristics. Nevertheless, the majority of the registration methods consist of feature detection, feature matching, transform model estimation, and image resampling and transformation [40, 39].

2.6 Deblurring

According to Dangeti [26], blurring is a form of bandwidth reduction of the image caused by the imperfect image formation process such as relative motion between the camera and the original scene or by an optical system that is out of focus. For instance, when aerial photographs are produced for remote sensing purposes, blurs are introduced by atmospheric turbulence, aberrations in the optical system and relative motion between camera and ground.

In general, a blurred or degraded image can be roughly described by the equation

$$g = Hf + n, \tag{2.6}$$

where g is the blurred image while H is the distortion operator, also known as the point spread function (PSF). The PSF describes the degree to which an optical system blurs, or spreads a point of light in the spatial domain [26, 40, 39]. Fundamentally, The PSF is the inverse Fourier transform of the optical transfer function (OTF). The OTF describes the response of a linear, position-invariant system to an impulse in the frequency domain. In essence, the OTF is the Fourier transform of the point spread function (PSF). The PSF or the distortion operator, when convolved with the image, creates the distortion. It is to be noted, however, that distortion caused by a point spread function (PSF) is just one type of distortion. Finally, f in the above equation represents the original true image, which in reality does not exist as it represents the image under perfect acquisition conditions. The last variable n represents the additive noise that actually corrupted the image, introduced during the acquisition of the image [26, 39].

2.7 Conclusion

The various image operations described above are certainly essential in the image processing area. In fact, they complement each other in many respects. For example, a denoising routine may first call the blurring routine to accomplish its task, and the same thing is true for the restoration routine. Indeed, most of the above operations are also preprocessing steps in the edge detection [26]. Furthermore, the registration operation is particularly important in many areas, such as computer vision, remote sensing, cartography, and medical imaging [40, 39].

Chapter 3

WORK ON EDGE AND BOUNDARY DETECTIONS

3.1 Overview

Edge detection forms the first stage in a very large number of vision components; and, as such, any edge detector should be formulated in the appropriate context. Nevertheless, the requirements of many vision modules are similar and it should be possible to design one edge detector that performs well in several contexts [7].

As stated in chapter one and reemphasized above, the quest to find an edge detection operator that performs well in several contexts has long been the goal of many researchers, but achieving it has not been easy [42, 43]. This chapter, therefore, provides illustrative examples of edge definition and background while summarizing the work that has been previously done.

3.2 Definition of Edges

In chapter one and two, a rather brief introduction of edges, including their properties and some of the means by which edges are detected was given. It was also noted that edges are important because they convey important information about the image, and detecting them allows us to gain useful information which, as a result, make subsequent image operations easier [44, 45]. Furthermore, it was also claimed that edges are abrupt changes in an image in chapter two. To understand what was meant, we must first understand the origin of edges and the factors that cause them [1, 46, 47].

Fig. 3.1 illustrates the conversion of a two-dimensional image into a set of curves where extraction of salient features of the scene can be carried out [1, 48, 49]. Note, however, that the curves in this figure are more compact than typical edges, but the idea of extracting the important feature of a scene is clearly illustrated therein. Still, the question remains

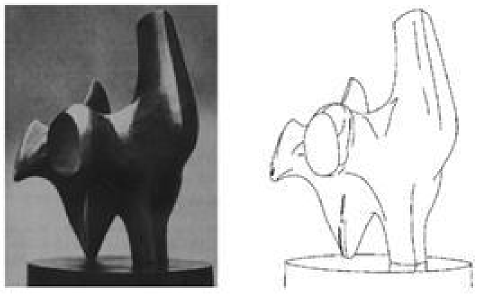


Figure 3.1: Converting a two-D image into a set of curves.

as to what caused edges in the first place? Remember it was mentioned in chapter one that various discontinuities are the main causes of edges. To further understand this reality, Fig. 3.2 illustrates several of these discontinuities [50, 51, 16].

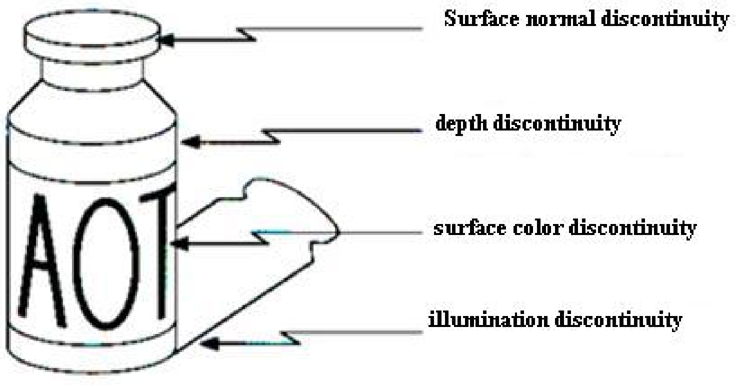

Surface normal discontinuity
depth discontinuity
surface color discontinuity
illumination discontinuity
AOT

Figure 3.2: Various factors responsible for the occurrence of edges.

From the above illustrations, we can see that edges are the places in the object corresponding to the object boundaries, and are, as a result, technically defined as pixels where image brightness changes abruptly [52, 53, 17]. This can be seen by examining Fig. 3.3, which compares and contrasts brightness and spatial coordinates.

As such, an edge can also be described as a property attached to an individual pixel and is calculated from the image function behavior in its neighborhood. It is, therefore, considered a vector variable consisting of magnitude of the gradient and the direction of

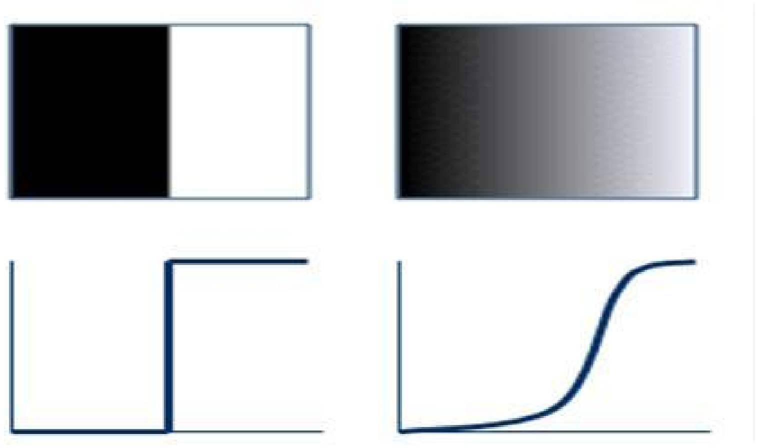


Figure 3.3: Brightness versus spatial coordinates.

an edge [54, 55, 20]. In particular, edge information in a given image is found by looking at the relationship a pixel has with its neighbors. If a pixel's gray-level value is similar to those around it, then there is a good chance that there is no edge presence at that point. By contrast, if a pixel has neighbors with widely varying gray-levels, there is high likelihood of an edge point at that location [56, 57, 58]. An edge point could be any of the available edge types as illustrated by Fig. 3.4.

Note that the gradient magnitude and the gradient direction are continuous image functions consisting of the angle in radians from the x-axis to the point (x,y). Remember it was established earlier that an edge is where change occurred, and this change is measured by a derivative in one-dimensional space. Anytime we encountered the biggest change during this measure, one of two things will be realized: the derivative will have a maximum magnitude or the second derivative will be zero [58, 59, 7]. Note from the chapter two discussion that the former case describes the gradient family of edge detectors while the latter is a typical behavior of the Laplacian edge detection methods [7, 60]. The gradient of an image is given by

$$\nabla f = \left[\frac{\delta f}{\delta x}, \frac{\delta f}{\delta y}\right] \tag{3.1}$$

The image gradient points in the direction of most rapid change in intensity. Possible directions are horizontal, vertical, or diagonal or the gradiennt itself as illustrated by Fig. 3.5.

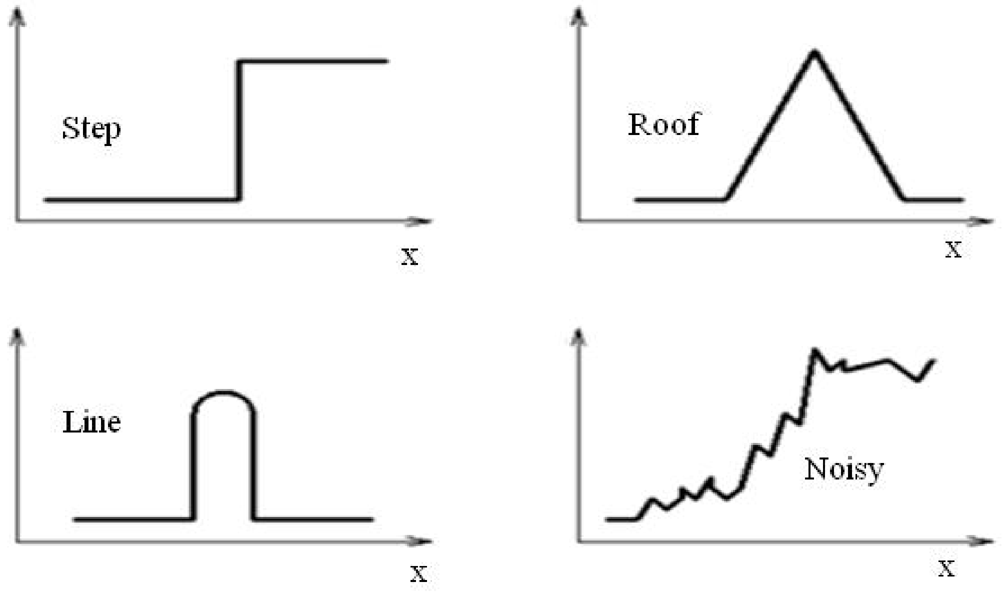


Figure 3.4: Classic edge profiles.

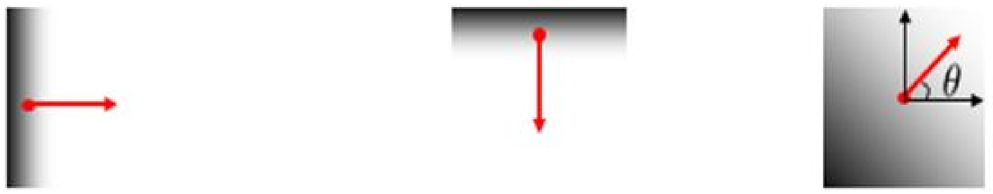


Figure 3.5: Image gradient directions.

$$\nabla f = \left[\frac{\delta f}{\delta x}, 0\right] \tag{3.2}$$

denotes the horizontal direction of the most rapid change in intensity whereas

$$\nabla f = \left[0, \frac{\delta f}{\delta y}\right] \tag{3.3}$$

denotes the vertical direction of the most rapid change in intensity. Finally, the image gradient

$$\nabla f = \left[\frac{\delta f}{\delta x}, \frac{\delta f}{\delta y}\right] \tag{3.4}$$

denotes the diagonal direction of the most rapid change in intensity. It is easy to see that the diagonal direction is the default direction of the image gradient [7, 60]. Finally, the gradient direction and the gradient magnitude are respectively given by

$$\theta = tan^{-1}\left(\frac{\delta f}{\delta y} / \frac{\delta f}{\delta x}\right) \tag{3.5}$$

and

$$||\nabla f|| = \sqrt{\left(\frac{\delta f}{\delta x}\right)^2 + \left(\frac{\delta f}{\delta y}\right)^2} \tag{3.6}$$

3.3 Pioneer Works on Edge Detection

Like many research areas, the field of edge detection is made famous by a number of pioneers who first charted the uncharted water. Among those is John F. Canny, who, until this day, is considered the most influential figure in the field [7, 61, 62, 10, 63]. Other figures, such as Roberts, Prewitt, Sobel, Marr, Hildreth and Hueckel, to mention just a few, have also done substantial work that opened the door for the next generation of researchers [8, 48, 1, 7, 63].

3.4 Edge Detection Operator by Canny

Canny's edge detection operator [1, 7, 5] aims at formulating a set of edge detection criteria that capture as directly as possible the desirable properties of an edge operator. In his work, Canny used variational techniques to find a solution over the space of all the linear shift invariant operators with the objective of detecting edges at the zero-crossings of the second directional derivative of the smoothed image. These vital criteria such as the low probability of error, good localization, and single response to single edge should, in fact, be the main goals of any edge detection operator [5]. Canny's operator accepts digitized image and produces an edge map, which includes information about the position and strength of edges, their orientation, and the scale at which the change took place, as its output. Oftentimes, quite great details, particularly structural information or the details of surface orientation and the material of which the visible surfaces comprise, are needed for a successful detection of given edges. This helps in deciding what technique to apply. In situations where the surfaces are smooth and of uniform reflectance, for instance, shape from shading, by Horn [1, 5], may be applied to obtain surface orientation. Furthermore, in other vision modules such as the shape from motion, shape from contour, shape from texture and stereo; structural properties are inferred from the edge contours [5, 16, 58, 8]. In fact, Canny derives his filter by optimizing a certain performing index that favors true positive, true negative and accurate localization. As mentioned earlier, analysis is, by and large, restricted to linear shift invariant filters that detect smooth one-dimensional continous step edges [1, 5]. It should be mentioned that other justifiable performance criteria, which would lead to different filters, are possible.

3.4.1 Canny's Criteria

As mentioned earlier, Canny's operator, like any good edge detector, attempted to satisfy several desirable criteria. These are good detection, good localization and elimination of multiple responses [1, 5].

- *Good detection:* A good detection is defined as a low probability of not marking real edge points, or falsely marking non-edge points. It is further defined mathematically by taking the signal-to-noise ratio (SNR)

$$SNR = \frac{\left| \int_{-w}^{w} G(-x) f(x) dx \right|}{n_o \sqrt{\int_{-w}^{w} f^2(x) dx}}, \tag{3.7}$$

  where f is the filter and G is the edge signal. The denominator is the root-mean-squared response to noise $n(x)$ only.

- *Good localization:* A good localization dictates that the edge detection operator detects edges that are close to the center of the true edge. It is mathematically defined as

$$Localization = \frac{1}{\sqrt{E\left[x_0^2\right]}} = \frac{\left| \int_{-w}^{w} G^{'}(-x) f^{'}(x) dx \right|}{n_o \sqrt{\int_{-w}^{w} f^{'2}(x) dx}}. \tag{3.8}$$

  It is to be noted from the above equation that the measure increases as localization improves, and it uses the reciprocal of the root-mean-squared (RMS) distance of the marked edge from the center of the true edge. Intuitively, the equation answers the question of 'if we assume the filter's response is the maximum at the edge when there is no noise, what is the expected distance of the local maximum?' Also note that the numerator is the second derivative of the filtered response, indicating how steep the slope of the zero-crossing of the filtered response is. Essentially, the steeper the slope, the sharper is the localization [1, 5].

- *Elimination of multiple responses:* this criterion simply dictates that there is only one response to single edge, which is actually implicit in the first criterion but made explicit here to eliminate multiple responses.

3.4.2 Canny's Miscellaneous Contributions

Of the existing edge operators, Canny's operator is considered the most optimal and versatile. This is because of its adherence to the necessary conditions for an effective edge detector. Notwithstanding this reality, he needed and had to improve it, for no edge detector that has by far been a cured-all. This is because a given condition is only fully satisfied at the expense of another. For example, more smoothing imporves detection but may hurt localization. This is an example of the detection/localization tradeoffs. So in an effort to improve his detector, Canny introduced the adaptive threshold, where he used the statistic of the image itself to set the threshold [5]. He further introduced a histogram, whose value is chosen at some percentile, say, the median, as a reference value of edge strengths. He then set the threshold as a multiple of this value, albeit not as a number but as a slowly varying function on the coarse grid [1].

In addition, Canny further introduced the idea of high and low thresholds. This mechanism is usually referred to as hysteresis. The hysteresis method uses two thresholds; the high and low thresholds. The high threshold is used to find 'seeds' for strong edges where it is required that the strength of the edges should be large enough so as not to ignore any edge. These seeds are allowed to grow into long edges in both direction as long as the edge strengths do not fall below the low threshold [1, 7, 5].

3.5 Roberts Edge Detection Operator

Frustrated by the inability of computers to construct and display three-dimensional array of solid objects from one 2-dimensional photograph, Lawrence Roberts decided to carefully analyze and mechanize the rules and assumptions of depth perceptions in an effort to turn the tide [48]. By assuming that a photograph is a perspective projection of a set of objects which can be constructed from transformation of known three-dimensional models, and that the objects are supported by other visible objects or by a ground plane, Roberts was able to devise a method to reasonably obtain, three-dimensional description from the edge information in a photograph by means of a topological, and mathematical process [1, 48].

Roberts wrote a computer program that can process an image into a line drawing, transform the line drawing into three-dimensional representation, and finally, display the three-dimensional structure from any point of view, with all the hidden lines removed [48]. This methodology was sufficiently generic to handle most of the planar-surfaced objects and has been, to date, quite instrumental in edge detection research. It has proven to be one of the most useful gradient operators today. A typical Robert Operator is given by the following equations starting with first form:

- First form:

$$\sqrt{[I(r,c)-I(r-1,c-1)]^2+[I(r,c-1)-I(r-1,c)]^2} \tag{3.9}$$

- Second form:

$$\left|I(r,c)-I(r-1,c-1)\right|+\left|I(r,c-1)-I(r-1,c)\right| \tag{3.10}$$

This resulted in

$$\left|I(r,c)-I(r-1,c-1)\right|+\left|I(r,c-1)-I(r-1,c)\right| \tag{3.11}$$

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{3.12}$$

and

$$h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{3.13}$$

The two matrices given by the two equations above represent Roberts' Operator (sometimes referred to as Roberts's Cross Operator) along the diagonal. They can equivalently be given as followed:

$$h_1 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \tag{3.14}$$

and

$$h_2 = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \tag{3.15}$$

Note that these matrices are derived to provide the gradient of the least square error planar surface fitted over a $2x2$ window [48]. Obviously, Roberts' operator marks edge points only, and provides no information about the edge orientation. For this reason, it tends to work best with binary images. Its primary disadvantages lies in its use of fewer pixels to approximate gradients and its high sensitivity to noise [48, 1].

3.6 Prewitt Edge Detection Operator

The Prewitt operator, which is a member of the gradient family of operators, on the other hand, calculates the maximum response of a set of convolution kernels to find the local edge orientation for each pixel. In this process, various kernels can be candidates for this operation. The entire set of 8 kernels is produced by taking one of the kernels and rotating its coefficients circularly [48, 1, 14]. In effect, each of the resulting kernels is sensitive to an edge orientation ranging from 0 to 315 degrees in steps of 45, where 0 corresponds to a vertical edge. Note also, that the maximum response for each pixel is the value of the corresponding pixel in the output magnitude image. The values for the output orientation image lie between 1 and 8, depending on which of the 8 kernels produced the maximum response [7, 21]. Some researchers referred to this edge detection method as edge template matching, because a set of edge templates is matched to the image, each representing an edge in a certain orientation. The edge magnitude and orientation of a pixel is then determined by the template that matches the local area of the pixel the best. Moreover, the Prewitt edge detector is an appropriate way to estimate the magnitude and orientation of an edge. While differential gradient edge detection needs a rather time-consuming calculation to estimate the orientation from the magnitudes in the x- and y-directions, the Prewitt edge detection obtains the orientation directly from the kernel with the maximum response. The set of kernels is limited to 8 possible orientations; however experience shows that most direct orientation estimates are not much more accurate [5, 64]. Finally, the Prewitt operator masks are matrices of $3x3$ and $4x4$ respectively in both vertical and horizontal directions.

$$\nabla_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.16}$$

$$\nabla_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{3.17}$$

$$\nabla_x = \begin{bmatrix} -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \end{bmatrix} \tag{3.18}$$

$$\nabla_x = \begin{bmatrix} 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -3 & -3 & -3 & -3 \end{bmatrix} \tag{3.19}$$

The Prewitt operator convolution masks are derived by fitting a least square error quadratic surface over $3x3$ image window, where the fitted surface is then differentiated [7, 20]. The

Prewitt operator, as mentioned above estimates the edge magnitude (likelihood of an edge) and direction as followed:

$$EM = \sqrt{x^2 + y^2} \tag{3.20}$$

$$ED = tan^{-1}\left[\frac{x}{y}\right] \tag{3.21}$$

3.7 Sobel Edge Detection Operator

Quite similar to Prewitt operator, the Sobel operator is a gradient operator and a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. In particular, at each point in the image, the result of the Sobel operator is either the corresponding gradient vector or the norm of this vector. Resembling the Prewitt operator in some sense, the Sobel operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations [5, 21]. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image. In other words, the operator calculates the gradient of the image intensity at each point, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The result therefore shows how "abruptly" or "smoothly" the image changes at that point, and therefore how likely it is that that part of the image represents an edge, as well as how that edge is likely to be oriented. In practice, the magnitude calculation is more reliable and easier to interpret than the direction calculation [7, 20, 58]. As can be expected, the Sobel operator calculates the edge magnitude and direction in similar fashion as Prewitt, but it has different mask coefficients [58].

$$\nabla_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.22}$$

$$\nabla_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{3.23}$$

3.8 Marr-Hildreth Edge Detection Operator

The Marr-Hildreth edge detetction operator was the detector of choice until Canny published his paper [20, 58, 19]. It is a gradient based operator which uses the Laplacian to take the second derivative of an image [19, 8]. The central idea is that if there is a step difference in the intensity of the image, it will be represented in the second derivative by a zero-crossing.

The algorithm proceeds as followed:

1. Smooth the image using a Gaussian. This is done so as to reduce the amount of error found due to noise.

2. Apply a two-dimensional Laplacian to the image:

$$r^2 f = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2} \tag{3.24}$$

   Because the Laplacian will be rotation invariant, it is often called the "Mexican Hat operator" because of its shape [8].

3. Loop through every pixel in the Laplacian of the smoothed image and look for sign changes. If a sign change is encountered and the slope across this sign change is greater than some threshold, mark this pixel as an edge. Alternatively, you can run these changes in slope through a hysteresis, later adapted by Canny, rather than using a simple threshold [8].

3.9 Hueckel Edge Detection Operator

Hueckel operator's main focus was on fast and reliable recognition of edges in the presence of noise. His goal was for his operator to obtain the best fit of an ideal edge element to any empirically obtained edge element [63]. In particular, given an image, Hueckel's detector tried to expand the neighborhood of the image and the step edge using nine basis functions. The trivial case of this approach uses a $2x2$ neighborhood and three basis functions. As a result, elementary methods are applied to explicitly solve this case [63]. Hueckel's operator turned out to be very similar to that of Roberts.

3.10 Other Notable Works on Edge Detection

Since the works of the various authors summarized in the above sections, much has been done in the area of edge detection to improve, supplement or extend many of them. Whether it was Nalwa-Binford approach to improve surface fitting, Sarkar-Boyer to improve the zero-crossing methodology, and the numerous researchers, who, one way or another, contribute a little to the field, the edge detection research continues to thrive [22]. This section, therefore, give a short summary of some of the works that seemed to have stand out from the bunch. This is not, by any mean, an attempt to give exhaustive summary of the works in this area, for it is not the aim of this dissertation. However, interested readers can consult some of the sources cited herein.

In 2002, Hou and Wei [3] proposed a discrete singular convolution (DSC) algorithm for edge detection. Their approach consisted of two classes of edge detectors namely DSC edge detector (DSCED) and DSC anti-noise edge detector (DSCANED) for the detection of multiscale edges. Their experimental results seem to be no worse than those obtained using some of the standard operators discussed earlier. Moreover, a sub -pixel edge detector based on Canny operator is proposed in [15] where a median filter is used to eliminate the salt and pepper noise. After that, a probable position of the image is then determined using Canny operator with the application of maximal variance threshold and a third-order spline interpolation.

Neoh and Hazanchuk [11] implemented an adaptive edge detection for real-time video processing using FGPAs. In their work, a combination of hardware and software components is suggested in the effort to achieve their desired edge detection filter that provides the necessary performance for real-time image and video processing while retaining system flexibility. An improved Canny operator is also proposed in [12] to include the x and y directions as well as the first-order partial finite differences of direction 45 and 135 degrees in calculating the amplitude values. Similarly, Wang and Fan [62] further improved the Canny operator by replacing the Gaussian filter with a self-adaptive filter and applying morphological thinning to thin the edges in order to alleviate some of the deficiencies of the Canny's operator. [6] seemed to have done the same by using the self-adaptive filter to detect cocoon edges.

Furthermore, an evolutionary techniques are proposed by [59] to design what they referred to as “highly parallel” edge detection nodes. Their aim was to implement a new image processing architecture through evolvable hardware that is able to adapt according to the particular images encountered. Zhou et al. [57], in 2009, proposed a vision theory to derive and edge detection algorithm to detect edges of uneven lighting images. An edge detection method, based on local features, to overcome the noise effect in an image by means of applying morphology is investigated in [56], while EdgeFlow, a technique for boundary detection and image segmentation is proposed by Ma and Manjunath [53]. Ant Colony Optimization (ACO), a metaheuristic approach for solving hard optimization problems, is presented by [50] to link disjoint edges produced by the Canny operator while Li et al. [65] introduced grey entropy to detect edges.

Some of the other notable techniques include the detection of edges using adaptive stochastic gradient technique proposed by [45], where the technique is supplemented with noise variance function, block-by-block gradient mask, and Rayleigh distribution threshold. On the other hand, Perona and Malik [4] proposed anisotropic diffusion technique to better detect edges at coarse scales by redefining the scale-space while Huan et al. [44] use mutual information to detect short boundary using Canny operator. In [55], an edge detector based on auto-adaptive double threshold Canny operator to detect tank level imaging is proposed. Finally, Boyer and others [54] used receiver operating characteristic (ROC) curves to evaluate performance of edge detector.

3.11 Edge Detection of Multispectral Images

The edge detection methods discussed thus far are differential operators based and are widely used in the processing of a one-band, sometimes referred to as grayscale, intensity, or monochromatic, images [66, 67]. In contrast, multi-band, multispectral or color images have been seldom studied. Difficulties in dealing with the three components of the red-green-blue (RGB) have foiled the small effort in developing the appropriate color edge detection algorithm. As it was stated in the earlier chapters, one-band or the grayscale image processing is made possible by the application of either the zero-crossing, sometimes referred to as Laplacian or the gradient method [66, 1].

Few researchers who have tried to persist in their quest to develop multispectral edge detection operators are usually met with the sheer difficulties of extending the monochromatic versions of the one band image into a reasonable combination of the RGB components [67].

3.11.1 Challenges

The task of detecting edges of a color or multispectral image has so far proven difficult. While some of the difficulties can be remedied, there has been no clear remedy for small subsets of the challenges, particularly when it comes to combining the results of individual color component into a meaningful and final RGB equivalent. The first known attempt to extend a monochromatic edge detector to a multispectral one, to the author's knowledge, was introduced in 1977 by Professor Ramakant Nevatia of the University of Southern California (USC) in a journal paper where he proposed to extend Hueckel operator, developed four years earlier, to multispectral images [68, 67].

Since then, not many journal publications have been published on the topic and the conference papers devoted to it simply attempt to extend existing grayscale methods, such as those of Hueckel and Canny, to multispectral or multi-band images; a task that is both daunting and, in many cases, too expensive. Although the argument could be made about the fact that it is intuitive to start with the solution to an easier problem in order to attack a more complex one, the obvious difficulties cannot be overlooked [68].

Furthermore, many of the multispectral approaches in literature can be divided into three categories namely; output fusion methods, multi-dimensional gradient methods, and vector methods. While output fusion methods enjoy greater recognition, their underlying characteristics of going through too many stages are utterly unacceptable. These methods proceed sequentially, performing edge detection three times, once each for red, green, and blue, or whatever color space is being used, and then the resulting output is fused (merged) in order to form one edge map. Some of the means typically used to merge the results include summation, the OR operator and weighted sum [68, 66, 65]. This scenario is better illustrated by Fig. 3.6.

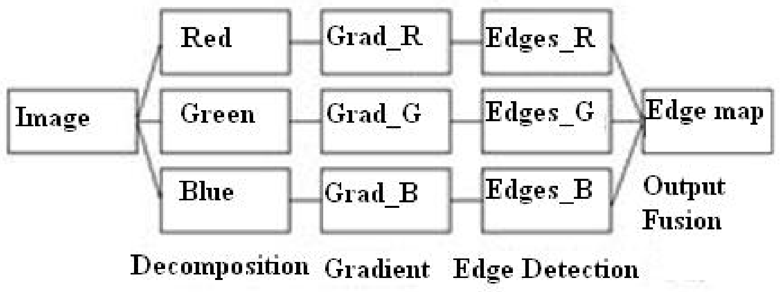


Figure 3.6: Fusion methods for multi-band image edge detection.

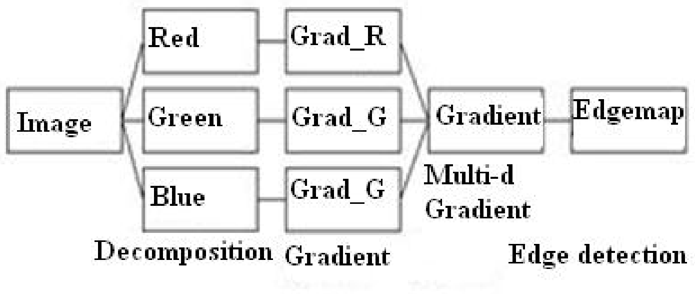


Figure 3.7: Multi-dimensional gradient methods for multi-band image edge detection.

On the other hand, multi-dimensional gradient methods exhibit somewhat the same characteristic as the output fusion methods, but they short-circuited the procedure by combining the three gradients into one where edge detection is only carried out once [68, 69].

Fig. 3.7 illustrates this procedure. Lastly, vector methods treat color of the pixels as vector throughput. For the most part, this is accomplish by either finding the best axis in the color space to project the image data so as to create a single-band image or using vector statistics to compute some statistical measures for edge detection [70, 68, 71]. Various works toward multispectral image are reviewed in the following section.

3.11.2 Brief Analysis

Of the methods mentioned above, the output fusion methods are the most popular. This is, in part, because of their potential to successfully detect edges of multispectral images, albeit at the expense of computational cost. As such, a good portion of the past literature review fall under these methods [68, 72, 73, 74]. Nonetheless, multi-dimensional gradient and vector methods, sometimes referred to as vector order statistic (VOS), have gained ground lately because their computational complexity is much more reasonable [75, 76, 77, 78].

Having touched on the various categories color edge detection operators are grouped into, it should be noted that some literatures also categorize these edge detectors according to those techniques that embed the variations of all color channels in a single measure and those that compute the gradient in each channel and then combine them according to some criteria [74, 68]. However, only the review of the works that fall under output fusion, multi-dimensional gradient and vector methods are summarized below. It should also be noted that the VOS edge detectors were inspired, in part, by the monochrome morphological edge operators. According to Evans and Liu, the development of monochrome morphological edge operators also resulted in the development of a new vector method for color edge detection termed as the color morphological gradient (CMG), which was later improved to robust CMG (RCMG) [74].

3.11.3 Output Fusion Methods

As mentioned early, output fusion methods apply single-channel edge detection techniques to each color plane and then combine the results using any of the available techniques mention previously. Gevers and Smeulders [79] introduced an output fusion method when they proposed a color image retrieval method using color constant ratio gradient for image segmentation and similarity of texture objects. Furthermore, Chunjiang and Yong [80] solved the problem of fusing the gradient by introducing a theory of belief function called Dempster-Shafer theory as a color image edge detection techniques. Madasu et al. [81] introduced a fuzzy edge and corner detector that could be extended to color images. Their main contribution is the application of fuzzy logic and the adaptive thresholding to obtain the image edge map. A somewhat similar approach is proposed in [82] for color edges enhancement and license plate segmentation. An alternative approach is investigated in [83] to detect color edges in visible human dataset. A color edge extraction techniques is presented in [84].

3.11.4 Multi-dimensional Gradient Methods

In multi-dimensional gradient methods, the gradients from the individual channels are recombined before the edge decision, giving rise to a single edge estimate [74]. In 2005, a multi-dimensional gradient method, referred to as Adaptive Color Image Segmentation (ACIS), utilizing neural network was introduced in [73]. The main contribution of that work is the use of multisigmoid activation function for the segmentation where the thresholds depend on the number of the image clusters. Gao [72] proposed an algorithm to detect human faces in a color image. The algorithm uses chrominance information to segment skin color regions. Wei and Rong [85] proposed a perceptual color edge detection algorithm. In this work, an Edge-Preserving Gaussian Filter (EPGF) is used as the smoothing operation to maintain the real edges. Cumani [66] also used a variation of the multi-dimensional gradient method in his edge detection operator. Koschan [86] presented a good comparison of some of the multi-dimensional gradient methods.

3.11.5 Vector Methods

The vector methods of multispectral images edge detection have lately received major attention because of relatively better computational complexity. As mentioned previously, the main problem with both output fusion and multidimensional gradient methods is how to combine the channels to give a final result. Christophe and Pearlman [77] extended the 3D-SPIHT (set partitioning in hierarchical trees) image compression algorithm that enables random access decoding of any specified region of the image volume at a given spatial resolution and given bit rate from a single codestream. This makes it possible to extract various qualities of the images while reading a minimum amount of bits from the coded data. Darbon et al. [76] presented an efficient algorithm for nonlocal image filtering with application in electron cryomicroscopy. It is based on property of neighborhood filtering in order to offer a fast parallel and vectorized implementation in contemporary shared memory computer architectures while reducing the theoretical computational complexity of the nonlocal mean filter. However, this computational complexity improvement doesn't seem to pertain to multispectral images. Although Liu and Pearlman [75] attempted to extend SPIHT coding algorithm by combining it with lattice vector quantization (LVG) in order to reduce computational load and design complexity, their experimental results only show promise for hyperspectral, rather than multipectral images.

A morphological gradient approach to color edge detection by Evans and Liu [74] based on vector differences shown good results, but it appeared that it has not greatly improve the high computation complexity that most, including vector, methods suffer from. In addition, it only provides estimate of the color gradient. In their work, the maximum distance between the vectors within a mask is given as he final output and they have extended the color morphological gradient (CMG) to robust CMG (RCMG) in order to make it more efficient. Another interesting approach was proposed by Tao and Huang [67] when they exploit cluster analysis where they used global color information to guide local gradient computation. Zou and Mel [87] combined cue with co-localized edge detectors to detect edges in natural scenes. Ruzon and Tomasi [68] proposed a compass operator to detect step edges with no assumption of constant regional color on each side. mainly, the compass operator substitutes the vector quantization and earth mover's distance (EMD) for Gaussian smoothing. One key shortcoming with this approach is the independent treatment of windows, which could possible leads to instability or even loss of edges. Several vector based approaches, with varying advantages and disadvantages, are investigated by [70], [69], [71], and [88].

Finally, and given the inadequacies of the various edges and corners detection techniques, especially in regard to computational time and image quality, a new and fresh idea is necessary to sort of get away from the usual. One important fact to note about all of the techniques, be they grayscale or multispectral, discussed thus far is that they all seem to be slight variations of the already existing and expensive ones in addition to each others. With the staggering need to decode image data, formulating a technique that would aid in the necessary features extraction, including edges, corners and curves, in a timely manner, is greatly needed. To inexpensively do this, we carefully analyze a new and emerging technique in data and signal processing, known as the empirical mode decomposition (EMD) developed by Huang et al. [89] in 1998. Because of its adaptability and efficiency, this method shows a great potential in successfully addressing most of the issues underlined above, in regard to image analysis in general and feature extraction in particular, given the right expansion. Chapters 5 and 6 present the EMD and its extension to two-dimensional data, called the bidimensional empirical mode decomposition (BEMD). Its extension and application to image analysis and feature extraction is given in Chapter 7 while Chapter 8 demonstrates its usefulness by presenting experimental results.

3.12 Definition of Corners

In the area of image processing, edge detection is not the only necessary operation, for many other, some of which were summarized in Chapter 2, operations play pivotal roles [90]. Interest points detection is an important image task and has been explored in the literature. Many applications require linking two or more images for the purpose of extracting information from them. If two successive frames in a video sequence taken from a moving camera can be related, for instance, it is possible to extract information regarding the depth of objects in the environment and the speed of the camera.

Although there exist brute force methods of comparing every pixel in the two images, they are computationally complex [91]. Intuitively, one can imagine relating two images by matching only locations of interest in the image. Such locations are referred to as "interest points" and are located using an interest point detector [91, 92]. Finding a relationship between images is then performed using only these points. This is more efficient as it significantly reduces the required computation time [92].

Many different interest point detectors have been proposed with a broad range of definitions for what points in an image are interesting. Various detectors find points of high local symmetry; some find areas of highly varying texture, while others locate corner points. Corner points are interesting as they are formed from two or more edges and edges usually define the boundary between two different objects or parts of the same object [92]. For this reason, this research incorporates an adaptive corner detection methodology to go along with and supplement the underlying edge detection technique. It may be tempting to assume that corner detection is synonymous with edge detection. However, they are in fact different in some sense. While there exist some inherent similarities between the two operations, there are mathematical distinctions in the approaches. As a result, edge detectors tend to fail at corners and vice versa. The underlying intuition is that gradient is not well defined at the corner and has two distinct values near it [92, 90]. To understand the similarities and dissimilarities, a list of desirable properties for a corner detector is provided below [90]:

- All “true corner” should be detected
- No “false corner” should be detected
- Corner points should be well localized
- Detector should have a high repeatability rate, or good stability
- Detector should be robust with respect to noise
- Detector should be computationally efficient

From the above properties, it is easy to reason that most, if not all, of the properties are common to both edge and corner detectors. The mathematical formula to find the corner, therefore, is

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{3.25}$$

where a sum of a hypothetical corner is taken over a small region and two gradients with respect to *y* and *x* respectively, are multiplied. Note that the matrices are symmetries of each other. So in essence, detecting the corners of an image is carried out by filtering the image, computing the gradient of magnitude everywhere, and constructing a C window [90, 93].

To illustrate the importance of some of the desired properties outlined above, namely repeatability and localization, Fig. 3.8 shows a situation where the detector is able to detect all the corners the first time it encounter the object but fails to detect all the corners of the same object the second time around [93]. Such a detector is said to have a poor repeatability rate. Similarly, Fig. 3.9 illustrates the idea of localization, that is the ability of a detector to detect the desired feature at or near the center of the object. Note that the detecor used to detect the corner of the left-most profile has a poor localization while the one used on the right has the desirable localization. Finally, the idea of corner profiles is demonstrated in Fig. 3.10 [90, 93].

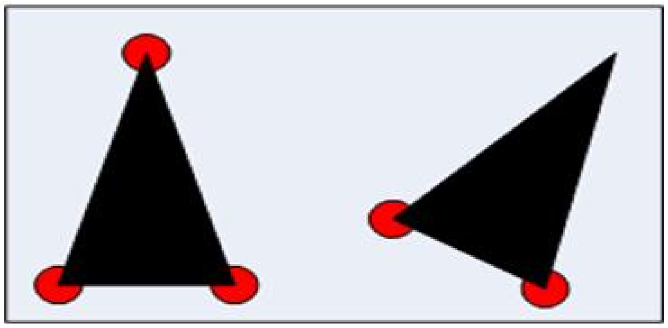


Figure 3.8: Example of a detector with a good and poor repeatability and robustness.

3.13 HARRIS Corner Detector

The HARRIS [94] corner detector, sometimes known as PLESSEY, was developed in 1988 by Harris and Stephens. This detector, as is typically the case, was developed with the intention to address the limitations of some of the earliest detection operators such as Moravec operator. This resulted in a more desirable detection operator in terms of good

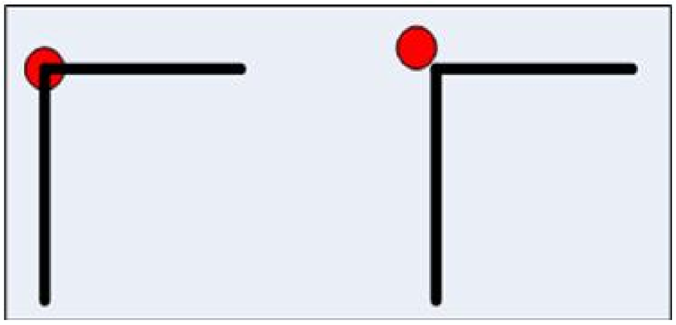


Figure 3.9: Example of a detector with a good and poor localization.

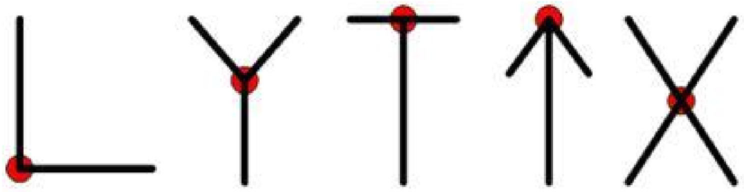


Figure 3.10: Various edge profiles (from left to right: L-junction, Y-junction, T-junction, Arrow-junction, and X-junction.

detection and repeatability rate. Nevertheless, this was at the cost of requiring significantly more computation time. Due to the lack of alternative however, the HARRIS operator was widely used during its time [90, 94].

The basic idea behind HARRIS' operator is the matching of corresponding points in consecutive image frames while tracking corners between frames. During this process, the operator tries to find points with large corner response function R, where R is greater than some given threshold. Then the points of local maxima of R are taken to define the corner [94]. This produces the change of intensity for the shift [u, v]

$$E(u,v) = \sum_{x,y} w(x,y)\,[I(x+u,y+v) - I(x,y)]^2 \tag{3.26}$$

where $w(x,y)$ is the window function, $I(x+u,y+v)$ is shift intensity and $I(x,y)$ is the intensity.

3.14 SUSAN Corner Detector

In 1995, Smith and Brady [90] proposed a new approach for low level image processing, especially for corner and structure preserving noise reduction. Because it is based on brightness comparison within a circular mask, they called it SUSAN (Smallest Univalue Segment Assimilating Nucleus) because it assumes that within a relatively small circular region, pixels belonging to a given object will have relatively uniform brightness. The pixels computation is achieved by calculating the number of pixels with similar brightness together with those at the center of the mask, which is the nucleus of the mask. These pixels are called the USAN (Univalue Segment Assimilating Nucleus) of the mask. Corners are then detected by applying the mask to all pixels in the image and then finding the local minima in this new USAN map [90].

Fig. 3.11 illustrates the circular mask applied to different positions of a black rectangle with the USAN shown in red color.

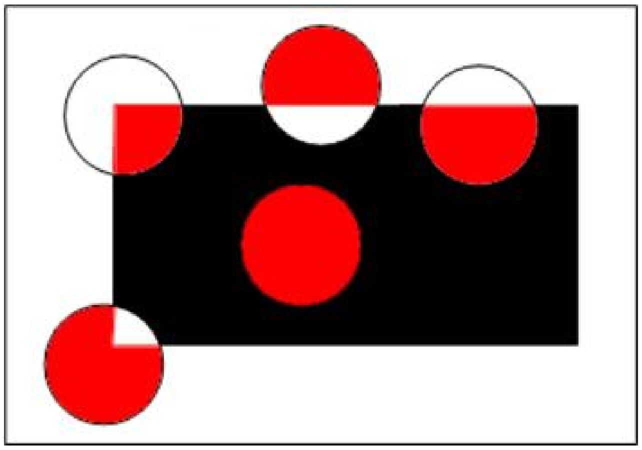


Figure 3.11: Illustration of USAN for different circular masks on a uniform rectangle.

Similarly, the SUSAN algorithm is fairly straightforward:

1. Determine a circular mask, typically of 37 pixels around a nucleus for each point within the image
2. Calculate the difference in brightness between each pixel of the mask and that of its

nucleus and

$$C(\overrightarrow{r},\overrightarrow{r}_0)=\begin{cases}1, & if \quad |I(\overrightarrow{r})-I(\overrightarrow{r}_0)|\leq t\\ 0, & otherwise\end{cases} \tag{3.27}$$

3. Sum the number of pixels within the circular mask which have similar intensity levels to that of the nucleus

$$n(\overrightarrow{r}_0)=\sum_{\overrightarrow{r}_0} C(\overrightarrow{r},\overrightarrow{r}_0) \tag{3.28}$$

   where r is a given pixel location within the mask and r_0 is the position of the mask nucleus in the image frame. $I(r)$ refers to the intensity of the pixel at location r, t is the brightness threshold, and c is the output of the comparison in intensities at locations r and r_0. The sum of the comparison outputs c is then taken and that represents the total number of pixels in the USAN region, in other words, the USAN area.

4. Compare n with g, the geometric threshold which is set to half of the maximum value that n can be $(n_{max}/2)$.

5. At a perfect corner (where two straight edges intersect) the USAN area will always be less than half the size of the mask area, and will be a local minimum

$$R(\overrightarrow{r}_0)=\begin{cases}g-n(\overrightarrow{r}_0), & if \quad n(\overrightarrow{r}_0)<g\\ 0 & otherwise\end{cases} \tag{3.29}$$

   The above rule is a simple formulation of the SUSAN principle since the response is inversely proportional to the USAN area.

3.15 Definition of Curves

Curves, in some sense, are similar to edges and corners and, as such, share number of similarities. However, curves represent boundaries and contours of the objects of interest. They tend to allow both smooth and sharp turning. And like edges and corners, curves detection is an important operation in image processing and computer vision [95]. Their specific application included road following, object tracking, fracture detection in borehole images and extraction of interesting features from medical images [96].

In general, the process involving the decision of whether a pixel constitutes a curve point or not is to see if it lies on some curves in the image and to link the points to form a curve. Furthermore, curvilinear structure in an image is then modeled as a curve that exhibits a characteristic of one-dimensional line profile in the direction perpendicular to the line [97, 96]. This situation is illustrated by Fig. 3.12.

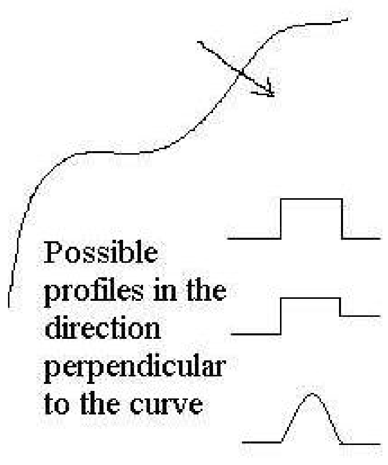


Figure 3.12: Traditional curve points classification.

The one-dimensional profile is characterized by a vanishing first derivative, which is the result of achieving the local maximum. When the direction perpendicular to the curve is given and is known to be lying on a curve, the first directional derivative in that direction vanishes while the second directional derivative achieves the largest absolute value [97]. In the context of an image, pixels are considered to be on an integer grid where a pixel has a boundary defined by the unit square

$$\left[x-\frac{1}{2},x+\frac{1}{2}\right] x \left[y-\frac{1}{2},y+\frac{1}{2}\right] \tag{3.30}$$

Thus, a pixel in an image is classified as a curve point if the first derivative along the given direction vanished within a unit square centered around the pixel, or within the pixel boundaries. The problem of computing the direction perpendicular to the curve is given by

$$H = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix} \tag{3.31}$$

The partial derivatives $I_{xx}, I_{xy}, I_{yx}, I_{yy}$ are computed numerically using partial differences, usually after convolving the image with the a Gaussian smoothing kernel. This kernel is necessary because the problem of estimating derivative of an image, particularly noisy one, is ill-posed. The use of the smoothing kernel , under very general assumptions, make this problem well posed and generally leads to scale-space description of the image [96].

3.16 Radon Transform

Curve detection, just like that of edges and corners, also play a role, though limited, in the general area of feature detection. And although there isn't as much literature on the subject as there is on edges and corners, it is still an important topic to examine. Of the few curve detection algorithms out there, one has shown greater potential warranting discussion, though brief, in this dissertation. The Radon Transform in general and the Generalized Radon Transform (GRT) in particular have been useful.

The Generalized Radon Transform (GRT) Rag [96] introduced is able to extract lines out of an image. In its basic form, a line can be represented by two parameters, such as the slope and the slope-intercept, and by varying these two parameters, it can obtain various lines. As the result, the Radon Transform of a continuous two-dimensional function $g(x,y)$ is found by stacking or integrating values $g(x,y)$ along slanted lines. In this process, the location of the line is determined from the line parameters (i.e., slope p and τ).

Similarly, GRT algorithm detects salient curves from the image by locating inner product maxima in the parameter space. In the case of first order Radon Transform, the inner product of each line with the image is calculated to give rise to a two-dimensional parameter representation in the (p,τ) rather than (x,y) domain of the image. Peaks in the parameter domain correspond to the salient lines present in the image [95].

The Generalized Radon Transform (GRT) can also be use effectively to extract faint curves from a noisy image. In given situations, GRT can handle higher order polynomial and other general functions as well as curve tracing. The GRT curve extraction described above, where curves are extracted from the image by locating inner product maxima in the parameter space, is given by

$$\hat{g}(p,\tau) = \int_{-\infty}^{\infty} g(x, px+\tau)dx \tag{3.32}$$

3.17 Conclusion

From the above definition of edges and corners, it easy to see that the task of detecting them is indeed imperative in order to understand the underlying image better. Furthermore, various operations, some of whom are preprocessing step in edge and corner detection tasks, also play a major role in the overall image processing. In addition, it should be noted that edge profile determines the level of complexity that an edge detection operator has to be in order to detect a given edge [1, 46].

The discussion of various approaches and improvements to detect edges and corners has certainly made it clear that a good bit of work has been done in the area. Nonetheless, it also shows that successfully detecting edges and other features of an object has proven to be a complex process both from computational standpoint as well as in algorithm development as numerous factors have to be considered. This explains the continuous effort in finding new ways, or improving the existing ones [1, 48, 90].

The works in this area has indeed matured, but it continues to be bolstered by the enthusiasm and the aggressive nature approach of the young researchers. The brief summary provided above indicates that desire to always find a better, be it in terms of computational time or efficiency, way to detect and process edges and corners [45, 4, 93].

Chapter 4

ROLE OF INTERPOLATION METHODS IN EDGE DETECTION

4.1 Introduction

Image interpolation is a common processing step in all digital images. It occurs anytime you resize or remap or distort an image from one pixel grid to another. While image resizing is necessary when there is a need to increase or decrease the total number of pixels, remapping can occur under a wider variety of circumstances. Example of these would be correcting lens distortion, changing perspective, and rotating an image [98, 99]. As such, interpolation works by using known data to estimate values at unknown points. If one wants to know the temperature at noon, but only measured it, say at 11AM and 1PM, for example, performing linear interpolation on the data would estimate the desired value. The overall concept will be clearer in the subsequent sections as the importance of the role interpolation plays in general and in edge detection in particular is emphasized and explained.

4.2 Interpolation Model

As partly mentioned above, image interpolation is a robust data estimator. It works in two directions, and tries to achieve the best approximation of a pixel's color and intensity based on the values at surrounding pixels. The importance of interpolation algorithms is underscore by their profound coverage in the literature as many of the image processing steps require some forms of interpolation [98, 100]. For this reason, common interpolation methods, including the radial basis functions (RBF) based methods, whose variance is used in this dissertation, are presented in this chapter, albeit in varying depth.

It is worth mentioning further that common interpolation algorithms can be grouped into two categories namely adaptive and non-adaptive. Adaptive methods adjust depending on whether they are interpolating sharp edges as opposed to smooth texture, whereas non-adaptive methods treat all pixels equally [98, 101].

4.3 Non-Adaptive Interpolation Algorithms

Non-adaptive algorithms are very popular, for they are practical and indiscriminate in nature. In these types of interpolation algorithms, certain computations are performed indiscriminately to the whole image for interpolation regardless of its contents [102, 103]. Depending on their complexity, these use anywhere from 0 to 256, and in some instances, more adjacent pixels during the interpolation. In effect, the more adjacent pixels they include, the more accurate they can become. However, this comes at the expense of much longer processing time [102, 98]. Many of these algorithms are heavily utilize in various commercial image processing tools or freeware graphic viewers such as the Adobe Photoshop CS2 software and IrfanView [102]. Though there are numerous non-adaptive interpolation algorithms in existence, only the most popularly used ones; namely nearest neighbor, bilinear, and bicubic are briefly discussed in this section while filtering-based method are briefly mentioned.

4.3.1 Nearest Neighbor Interpolation

One of the most basic interpolation algorithms is the nearest neighbor. Because it only considers one pixel that is the closest to the interpolated point, it is simple and requires the least processing time of all the interpolation algorithms. This simplicity makes it very inefficient because the resulting pixelization or blocky effect results in a poor image quality that is almost unacceptable for most high quality imaging applications [102].

4.3.2 Bilinear Interpolation

Bilinear interpolation, on the other hand, considers the closest 2 x 2 neighborhood of known pixel values surrounding the unknown pixel. In other words, it is the weighted average of four neighboring pixel values. This process of taking the weighted average to arrive at the final interpolated value produces much smoother looking images than nearest neighbor [102, 104]. Fig. 4.1 illustrates the case where all known pixel distances are equal, and the interpolated value is simply their sum divided by four.

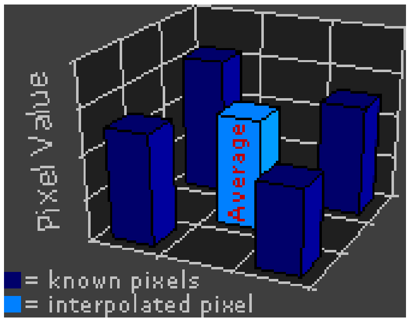


Figure 4.1: Bilinear Interpolation with weighted average.

4.3.3 Bicubic Interpolation

The bicubic interpolation, by contrast, uses sixteen neighboring pixels for estimation. As such, it goes one step beyond bilinear by considering the closest $(4x4)$ neighborhood of known pixels for a total of 16 pixels. Since these pixels are at various distances from the unknown pixel, closer pixels are given a higher weighting in the calculation. Therefore, bicubic produces visibly sharper images than the previous two methods, and is perhaps the ideal combination of processing time and output quality. For this reason it is a standard in many image editing programs, including Adobe®, Photoshop®, printer drivers and in-camera interpolation [102]. Fig. 4.2 shows a pictorial representation of the bicubic interpolation.

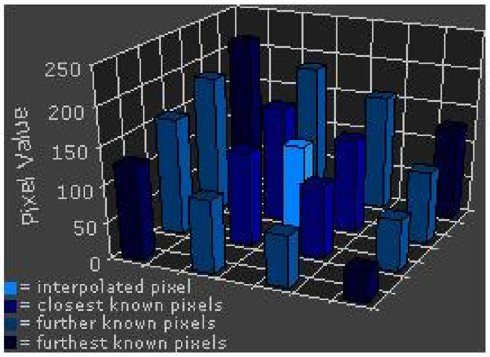


Figure 4.2: Bicubic Interpolation with 16 pixels.

4.3.4 Filtering-based Interpolation

In addition to the interpolation algorithms discussed above, there exist filtering-based interpolation methods. These methods are also known as re-sampling methods because they transform discrete image pixels defined at one coordinate system to a new coordinate system of a different resolution [102]. Nonetheless, these methods will not be discuss further in this dissertation.

4.4 Adaptive Interpolation Algorithms

Various adaptive interpolation techniques have been proposed in the literature [102]. Quite a few of the adaptive interpolation are proposed based on logics of local structure of the image and intensity variations that are indistinguishable by human eyes. However, adaptive interpolation algorithms include many proprietary algorithms in licensed software such as Qimage, PhotoZoom Pro, Genuine Fractals and many more. Many of the software vendors apply a different version of their algorithm on a pixel-by-pixel basis when detecting the presence of an edge with the aim of minimizing unsightly or unpleasant interpolation artifacts in regions where they are most apparent [102, 105, 101].

4.5 Surface Interpolation Algorithms

Surface interpolation methods take into account vector-valued surface and other visual cues for surface reconstruction when interpolating. These types of interpolators take more surrounding pixels into consideration, and are thus also much more computationally intensive. Moreover, they retain the most image information after an interpolation. As a result, they are extremely useful when the image requires multiple rotations or distortions in separate steps. However, for single-step enlargements or rotations, these higher-order algorithms provide diminishing visual improvement as processing time is increased [102, 105, 100].

4.6 Radial Basis Function (RBF)

Surface interpolation from two-dimensional scattered data, typically the local maxima and minima points, is a crucial part of many image analysis methods and particularly the bidimensional empirical mode decomposition (BEMD) method utilize in this research.

Typically, the choice of the appropriate interpolation method depends on the specific structure of the problem [101, 102]. With a scattered data, global interpolation algorithm is necessary to interpolate the data. As such, radial basis function (RBF) based interpolation methods are good candidates for global interpolation methods for scattered data points. This is based, in part, on the fact that RBF methods impose fewer restrictions on the geometry of the interpolation centers and are consequently suited to problems where the interpolation centers do not form a regular grid as is the case for local maxima or minima maps existing in the BEMD process. Moreover, RBF methods are among the most elegant schemes from a mathematical point of view [101]. Some of the most popular RBFs; namely Linear (L), Cubic Spline (CS), Thin Plate Splines (TPS), Multiquadrics (M), Inverse Multiquadrics (IM), Exponential Spline (ES), Guassian Spline (GS) and Compactly Supported Spline (CSS) are given in Table 4.1.

For illustration purposes, consider, $f : R^d \to R$ a real valued function of d variables that is to be approximated by $s : R^d \to R$, given the values $f(x_i) : i = 1,2,3,...,N$ where $x_i : i = 1,2,3,...,N$ is a set of distinct points in R^d call the nodes of operations. Then RBF approximation s is given by

$$s(x) = p_m(x) + \sum_{i=1}^{N} \lambda_i \phi(||x - x_i||), x \in R^d, \lambda_i \in R \tag{4.1}$$

where s is the radial basis function (RBF); p_m is the low degree polynomial, which is a member of $m-th$ degree polynomials in d variables; $||.||$ denotes the Euclidean norms; the λ_i's are the RBF coefficients; ϕ is a real-valued function called the basis function, and the x_i's are the RBF centers. Table 4.1 provides some of the typical choices of ϕ for the various RBFs.

Note that the constant c in Table 4.1 maybe selected based on specific need and perfor-

Table 4.1: Various Choices of ϕ for Different RBFs.

RBFs	Choices of ϕ
Linear (L)	$\phi(r) = r$
Cubic Spline (CS)	$\phi(r) = r^3$
Thin Plate Splines (TPS)	$\phi(r) = r^2 log(r)$
Multiquadrics (M)	$\phi(r) = \sqrt{r^2 + c^2}$
Inverse Multiquadrics (IM)	$\phi(r) = 1/\sqrt{r^2 + c^2}$
Exponential Spline (ES)	$\phi(r) = e^{-cr}$
Gaussian Spline (GS)	$\phi(r) = e^{-cr^2}$
Compactly Supported Spline (CSS)	$\phi(r) = (1 - r^m) + p(r)$

mance for the cases of exponential splines and Gaussian splines [101].

4.7 Miscellaneous Interpolation Methods

In addition to the interpolation methods presented above, there exist still other methods suggested in the literature that perform well under specific situations. Nonetheless, most, if not all, of them are typically expensive in terms of computational time. For example, spline and sinc are higher interpolation algorithms that are useful when the image requires multiple rotations or distortion in separate steps [101, 102]. Several authors have respectively suggested the use of some XY interpolation techniques and bivariate interpolation to handle irregularly distributed data points [102].

Dai et al. [98] compared several interpolation algorithms in a network-based GPS techniques. In that comparison, Linear Combination Model (LCM), Distance-based linear Interpolation method (DIM), Linear Interpolation Method (LIM), Low-order Surface model (LSM), and Least Squares Collocation (LSC) are thoroughly discussed.

4.8 Conclusion

This chapter has presented and discussed some of the major interpolation algorithms and issues associated with them. It was emphasized that the non-adaptive interpolation algorithms play a major role in many commercial imaging software due to their practicality and generic nature. Moreover, the existence of adaptive interpolation algorithms, and the general interest in utilizing them was pointed out. Surface interpolation algorithms as well as radial basis functions (RBFs) for two-dimensional scattered data interpolation, whose variance is use in this dissertation, were also presented [98, 101, 102].

Chapter 5

EMPIRICAL MODE DECOMPOSITION (EMD)

5.1 Introduction

Empirical mode decomposition (EMD), developed by Norden E. Huang et al. [89] in 1998 is a method of breaking down a signal without leaving a time domain and, in some sense, serves the same purpose as the Fourier Transforms and wavelet decomposition image analysis techniques. It is, however useful for analyzing natural signals, which are, by and large, non-stationary, time-varying and non-linear [89, 106]. The Fourier Transform and wavelet decomposition methods deal strictly with stationary, periodic and linear data and signals. Because EMD decomposes a complex signal into finite and oscillatory modes known as intrinsic mode functions (IMFs), it is self-adaptive and efficient and, as a result, gained major impetus in various fields of study outside of computer science [89, 106, 107]. This popularity stemmed from the fact that the basic functions used to decompose a signal are not predefined but adaptively derived from the signal itself [108, 109, 110, 99, 100].

Huang and others were motivated by the inability of the existing data analysis techniques to effectively handle nonlinear and non-stationary data and signals. Because data analysis is important in research and practical applications, it is necessary to have a reliable and efficient mechanism of examining the data so that we can make sense of them so they argued [89, 111, 112, 113, 114]. Since one of the main goals of data analysis is to determine the parameters needed to construct the necessary model and to confirm the model that is constructed to represent a phenomenon, it is imperative that some of the inherent problems are addressed. Data, whether from physical measurement or numerical modeling will exhibit one or more of the following problems: The data span will be too short; the data are non-stationary; the data represent a nonlinear process [115, 116, 117, 118, 119, 120, 121, 122].

Because the existing methods, which are briefly summarized in the next section, do not address some of the above problems, at least in the pragmatic standpoint, EMD attempts to empirically bridge the gap [123, 89, 2]. Historically, Fourier spectral analysis has provided a general method for examining the global energy-frequency distributions, according to Huang et al. [89], and as a result, the term "spectrum" has become almost synonymous with the Fourier transform of the data. The Fourier transform, often abbreviated FT, is an operation that transforms one complex-valued function of a real variable into another [123, 89, 106]. In its simplest form, the Fourier transform changes the original function in its time domain to the frequency domain of the new function and is often referred to as the frequency domain representation of the original function. This is so because it describes which frequencies are present in the original function [107]. Since its introduction, it has become the data analysis technique of choice and has thus been applied to many types of data [111].

Nonetheless, and as mentioned previously, some important restrictions namely; application to only linear systems, and strict application to periodic or stationary data, of the Fourier spectral analysis, still need to be overcome, or else the resulting spectrum will make little physical sense [89]. It is worth mentioning that the stationarity requirement, which is defined below, is not particular to the Fourier spectral analysis but common to most of the available data analysis techniques.

According to the traditional definition, a time series, *X(t)*, is stationary in the wide sense, if, for all t,

$$\left.\begin{array}{c} E(|X(t)^2|) < \infty, \\ E(X(t)) = m, \\ C(X(t_1),(X(t_2)) = C(X(t_1+\tau),X(t_2+\tau)) = C(t_1-t_1) \end{array}\right\} \tag{5.1}$$

in which E(.) is the expected value defined as the ensemble average of the quantity, and C(.) is the covariance function. Stationarity in the wide sense is also known as weak stationarity, covariance stationarity or second-order stationarity [107, 89]. Furthermore, time series, *X(t)*, is strictly stationary, if the joint distribution of

$$[X(t_1),(X(t_2),...,(X(t_n)] \text{ and } [X(t_1+\tau),(X(t_2+\tau),...,(X(t_n+\tau)] \qquad (5.2)$$

are the same for all t_i and τ. Thus, a strictly stationary process with finite second moments is also weakly stationary, but the inverse is not true [111].

5.2 Non-Stationary Data Processing Methods

The development of the empirical mode decomposition (EMD) by Huang et al. has indeed inspired the data processing research community in general and image processing in particular. In justifying the merit of their method, the authors gave a good, albeit brief survey of the methods available for processing non-stationary data [89]. In doing so, they fittingly noted that most of the methods still depend on Fourier analysis, but emphasized the fact that they are limited to linear systems only. It is also important to mention that of the few methods available, the adoption of any method is almost strictly determined according to the special field in which the application is made [89, 111]. The summary of the available methods is provided below.

5.2.1 The Spectrogram

Of all the methods, the spectrogram is considered the most basic method, for it is nothing more than a limited time window-width Fourier spectral analysis. By successively sliding the window along the time axis, one can get a time-frequency distribution. Given that it relies on the traditional Fourier spectral analysis, it is traditionally assumed that the data are piecewise stationary [89, 111]. However, this assumption is not always justified in non-stationary data, for even though the data are piecewise stationary, it is difficult, if possible at all, to guarantee that the window size adopted always coincides with the stationary time scales. Huang et al. [89], therefore pondered the following questions in regard to the spectrogram data processing method: What can we learn about the variations longer than the local stationary time scale? Will the collection of the locally stationary pieces constitute some longer period phenomena?

Moreover, the authors further noted that there are practical difficulties in applying the method. For instance, in order to localize an event in time, the window width must be narrow, while, on the other hand, the frequency resolution requires longer time series. These conflicting requirements render this method of limited usage. It is, however, extremely easy to implement with the fast Fourier transform. As a result, it has attracted a wide following and has been applied to speech pattern analysis, among other applications [89, 123].

5.2.2 The Wavelet Analysis

The wavelet approach is fundamentally an adjustable window Fourier spectral analysis with the following general definition:

$$W(a,b;X,\psi) = |a|^{-1/2} \int_{-\infty}^{\infty} X(t)\psi^{*}\left(\frac{t-b}{a}\right) dt, \tag{5.3}$$

in which $\psi^{*}(.)$ is the basic wavelet function that satisfies certain very general conditions, a is the dilation factor and b is the translation of the origin. Although time and frequency do not appear explicitly in the transformed result, the variable $1/a$ gives the frequency scale and b, the temporal location of an event. An intuitive physical explanation of equation (6.3) is very simple: $W(a;b;X;)$ is the "energy" of X of scale a at $t = b$ [89, 111].

Because of this basic form of $at+b$ involved in the transformation, it is also known as affine wavelet analysis. For specific applications, the basic wavelet function $\psi^{*}(.)$, can be modied according to special needs, but the form has to be given before the analysis. In most common applications, however, the Morlet wavelet is defined as Gaussian enveloped sine and cosine wave groups with 5.5 waves [89]. Generally, $\psi^{*}(.)$ is not orthogonal for different a for continuous wavelets. Although one can make the wavelet orthogonal by selecting a discrete set of a, this discrete wavelet analysis will miss physical signals having scale different from the selected discrete set of a. Continuous or discrete, the wavelet analysis is basically a linear analysis. A very appealing feature of the wavelet analysis is that it provides a uniform resolution for all the scales. Limited by the size of the basic wavelet function, the downside of the uniform resolution is uniformly poor resolution [89, 111, 120].

Although wavelet analysis has been available only in the last twenty years or so, it has become particularly popular. Indeed, it proved its usefulness in analyzing data with gradual frequency changes. Since it has an analytic form for the result, it has attracted far-reaching attention of the applied mathematicians. Nonetheless, most of its applications have been in edge detection and image compression [111, 120].

Versatile as the wavelet analysis is, the problem with the most commonly used Morlet wavelet is its leakage generated by the limited length of the basic wavelet function, which makes the quantitative definition of the energy-frequency-time distribution difficult, according to Huang et al. [89]. Sometimes, the interpretation of the wavelet can also be counterintuitive. For instance, to define a change occurring locally, one must look for the result in the high-frequency range, for the higher the frequency the more localized the basic wavelet will be. If a local event occurs only in the low-frequency range, one will still be forced to look for its effects in the high-frequency range. Such interpretation will be dicult if not impossible [121, 122, 89]. Another difficulty of the wavelet analysis is its non-adaptive nature. Once the basic wavelet is selected, one will have to use it to analyze all the data. Since the most commonly used Morlet wavelet is Fourier based, it also suffers the many shortcomings of Fourier spectral analysis: it can only give a physically meaningful interpretation to linear phenomena; it can resolve the interwave frequency modulation provided the frequency variation is gradual, but it cannot resolve the intrawave frequency modulation because the basic wavelet has a length of 5.5 waves. In spite of all these problems, wavelet analysis is still the best available non-stationary data analysis method so far [121, 122, 89].

5.2.3 The Wigner-Ville Distribution

The Wigner-Ville distribution is sometimes also referred to as the Heisenberg wavelet. By definition, it is the Fourier transform of the central covariance function. For any time series, X(t), the central variance can be defined as

$$C_c(\tau,t) = X(t - \frac{1}{2}\tau)X^*(t + \frac{1}{2}\tau). \tag{5.4}$$

Then the Wigner-Ville distribution is

$$V(\omega,t) = \int_{-\infty}^{\infty} C_c(\tau,t)e^{-i\omega\tau}d\tau. \tag{5.5}$$

This transform has found quite a following in the literature for the last several years. It has particularly been extremely popular with the electrical engineering community. The difficulty with this method, however, is the severe cross terms as indicated by the existence of negative power for some frequency ranges. Although this shortcoming can be eliminated by using the Kernel method, the result is, then, basically that of a windowed Fourier analysis and would, therefore, inherit all the limitations of the Fourier analysis. Although some nice extensions to the method have been proposed in the literature for application to complicated data, they require great amount of judgment [121, 122, 89, 120].

5.2.4 Evolutionary Spectrum

The basic idea behind the evolutionary spectrum was to extend the classic Fourier spectral analysis to a more generalized basis: from sine or cosine to a family of orthogonal functions $\{\phi(\omega,t)\}$ indexed by time, t, and defined for all real ω, the frequency. Then, any real random variable, $X(t)$, can be expressed as

$$X(t) = \int_{-\infty}^{\infty} \phi(\omega,t)dA(\omega,t), \tag{5.6}$$

in which $dA(\omega,t)$, the Stieltjes function for the amplitude, is related to the spectrum as

$$E(|dA(\omega,t)|^2) = d\mu(\omega,t) = S(\omega,t)d\omega, \tag{5.7}$$

where $\mu(\omega,t)$ is the spectrum, and $S(\omega,t)$ is the spectral density at a specific time t, also designated as the evolutionary spectrum. If for each fixed ω, $\phi(\omega,t)$ has a Fourier transform

$$\phi(\omega,t) = a(\omega,t)e^{i\Omega(\omega)t}, \tag{5.8}$$

then the function of $a(\omega,t)$ is the envelope of $\phi(\omega,t)$, and $\Omega(\omega)$ is the frequency. If, further, we can treat $\Omega(\omega)$ as a single valued function of ω, then

$$\phi(\omega,t) = a(\omega,t)e^{i\omega t}. \tag{5.9}$$

Thus, the original data can be expanded in a family of amplitude modulated trigonometric functions [118, 119, 89]. The evolutionary spectral analysis is very popular in the earthquake community. The difficulty of its application is to find a method to define the basis, $\{\phi(\omega,t)\}$. In principle, for this method to work, the basis has to be defined *a posteriori*. So far, no systematic way has been offered; therefore, constructing an evolutionary spectrum from the given data is impossible. However, the earthquake community change the application of this method from data analysis to data simulation in order to utilize it. As such, the evolutionary spectrum analysis has never been very useful in its current form [119, 89].

5.2.5 The Empirical Orthogonal Function Expansion (EOF)

The empirical orthogonal function expansion (EOF) is also known as the principal component analysis, or singular value decomposition method. The essence of EOF is briefly summarized as follows: for any real $z(x,t)$, the EOF will reduce it to

$$z(x,t) = \sum_{1}^{n} a_k(t) f_k(x), \tag{5.10}$$

in which

$$f_j . f_k = \delta_{jk}. \tag{5.11}$$

The orthonormal basis, $\{f_k\}$, is the collection of the empirical eigenfunctions defined by

$$C.f_k = \lambda_k f_k, \tag{5.12}$$

where C is the sum of the inner products of the variable.

According to Huang et al. [89], EOF represents a radical departure from all the above methods, for the expansion basis is derived from the data; therefore, it is a *posteriori*, and highly efficient. The critical flaw of EOF is that it only gives a distribution of the variance in the modes defined by f_k, but this distribution by itself does not suggest scales or frequency content of the signal. Although it is tempting to interpret each mode as independent variations, this interpretation should be viewed with great care, for the EOF decomposition is not unique. A single component out of a non-unique decomposition,even if the basis is orthogonal, does not usually contain physical meaning [116, 117, 89]. Nonetheless, the EOF method has been very popular, especially in the oceanography and meteorology communities because of its adaptability.

5.2.6 Other Miscellaneous Methods

In addition to the methods mentioned above, there are also several methods such as least square estimation of the trend, smoothing by moving averaging, and differencing to generate stationary data. These methods, though useful, are too specialized to be of general use [116, 117]. Furthermore, it is to be noted that all the above methods are designed to modify the global representation of the Fourier analysis, but they all failed in one way or the other. Having discussed the methods, we can summarize the necessary conditions for the basis to represent a nonlinear and non-stationary time series: (a) complete; (b) orthogonal; (c) local; and (d) adaptive. The first condition guarantees the degree of precision of the expansion; the second condition guarantees positivity of energy and avoids leakage. The first two requirements are standard requirements for all the linear expansion methods. For nonlinear expansions, the EMD methodology subtly modifies the orthogonality condition. The details of how it does it will be discussed in the next section. It should be mention here that some of the above mentioned methods do not meet the two standard requirements. Furthermore, the locality and adaptivity conditions are particular to the nonlinear and non-stationary data [89].

The requirement for locality is the most crucial for non-stationarity because in such data, there is no time scale; therefore, all events have to be identified by the time of their occurrences. Accordingly, the EMD method requires both the amplitude (or energy) and the frequency to be functions of time. The requirement for adaptivity is also crucial for both nonlinear and non-stationary data, for only by adapting to the local variations of the data can the decomposition fully account for the underlying physics data [89, 116, 117].

Thus, the EMD employs a two-step process to analyze the data. The first step involves preprocessing the data using the empirical mode decomposition method, with which the data are decomposed into a number of intrinsic mode function components, essentially expanding the data in a basis derived from the data themselves. The second step is to apply the Hilbert transform to the decomposed IMFs and construct the energy-frequency-time distribution, designated as the Hilbert spectrum, from which the time localities of events will be preserved [89].

5.3 Intrinsic Mode Functions

An intrinsic mode function (IMF) is a function that satisfies two conditions: (1) in the whole data set, the number of extrema and the number of zero crossings must either equal or differ at most by one; and (2) at any point, the mean value of the envelope defined by the local maxima and the envelope defined by the local minima is zero [89, 124]. The specifics of these conditions are summarized below.

5.3.1 IMFs and Residue Properties

As mentioned above, the principle of the EMD lie in the decomposition of a signal into its one-dimensional (1D) intrinsic mode functions (IMFs), designated as $f([n])_{n\in\mathbb{Z}}$, and a 1D residue based on the local frequency or oscillation information [125, 126, 101, 104, 127]. The process by which IMFs are constructed is referred to as a "sifting process". The principle of EMD is based on the characterization of f through its decomposition in intrinsic mode functions (IMF) that are defined as followed [125, 104, 103, 128]:

Definition 1: A function is considered an intrinsic mode function if the number of extrema equals the number of zero-crossings and if it has a zero local mean. Based on this definition, the following is the fashion in which it proceeds [129, 130, 125]:

1. Initialization: $r_0 = f, k = 1$
2. Computation of the kth IMF, d_k(SP)

   a Initialization: $h_0 = r_{k-1}, j = 1$

   b Identify all the local extrema of h_{j-1}

   c Interpolate the local minima (or maxima) to get the envelope, $Env_{min,j-1}$ or $(Env_{max,j-1})$

   d Compute the mean of these envelops as $Env_{mean,j-1}(t) = \frac{1}{2}(Env_{min,j-1}(t) + Env_{max,j-1}(t))$

   e $h_j[n] = h_{j-1}[n] - Env_{mean,j-1}(n)$

   f If the stopping criterion is fulfilled then $d_k = h_j$ otherwise $j = j+1$
3. $r_k[n] = r_k - 1[n] - d_k[n]$
4. if r_k is not monotonic, go to step 2 else the decomposition is complete

At that point, f becomes:

$$f[n] = \sum_{k=1}^{K} d_k[n] + r_K[n], K \in \mathbb{N}^* \tag{5.13}$$

With this presentation, we can see that the key point of the algorithm is the sifting process (SP), which is entirely defined by an interpolation method (most commonly cubic spline interpolation) and by a stopping criterion. Furthermore, it should be noted, to the knowledge of author, that there doesn't seem to be a mathematical proof of the convergence of the algorithm [125, 131, 132].

5.4 Classical EMD Process Overview

Having summarized the IMFs algorithms and its propeties, it is now appropriate to briefly introduce the classical EMD process overview in order to facilitate the upcoming discussion.

From the previous discussion, it was established that the EMD method is the ideal candidate for processing non-stationary and nonlinear signal or data, for the currently available methods only fit to properly deal with linear and stationary data and signals [89, 124]. For the purpose of discussing the EMD, let the original 1D signal/data be denoted by $d(x)$, the ith IMF by $f_i(x)$, and the the residue (R) by $r(x)$. For a signal with m data points, $x \in 1 : m$. In a classical and other standard EMD processes, multiple iterations are required to obtain an IMF, $f_i(x)$, where the intermediate state of an IMF (ISIMF) in the jth iteration can be denoted by $f_{i,j}(x)$. In the process, $f_i(x)$ is obtained from its source signal/data $s_i(x)$. Given the above definitions, the basic steps of a classical EMD method can be summarized as [89, 124, 2]

(i) Set $i = 1$, and $s_i(x) = d(x)$. If $s_i(x)$ (i.e., $d(x)$) is not the only component, with $s_i(x)$ having the residue (R) properties, then go to step two.

(ii) Set $j = 1$, and $f_{i,j}(x) = s_i(x)$.

(iii) Obtain the local maxima points of $f_{i,j}(x)$, which is known as the maxima map and denoted by $p_{i,j}(x)$. Similarly, obtain the local minima points of $f_{i,j}(x)$, which is known as the minima map and denoted by $q_{i,j}(x)$. Note that the data points in the maxima and minima maps corresponding to the local maxima and minima points in the ISIMF, $f_{i,j}(x)$, will be non-zero or zero whereas the other points in the maxima and minima maps will always be zero [2].

(iv) Generate the upper envelope (UE), $u_{i,j}(x)$, and the lower envelope (LE), $l_{i,j}(x)$, of intermediate state IMF (ISIMF), $f_{i,j}(x)$, from the maxima points in $p_{i,j}(x)$ and minima points in $q_{i,j}(x)$, respectively.

(v) Find the mean envelope (ME) as $m_{i,j}(x) = \frac{u_{i,j}(x)+l_{i,j}(x)}{2}$.

(vi) Calculate $f_{i,j}(x)$ as $f_{i,j+1}(x)$ = $f_{i,j}(x)$ - $m_{i,j}(x)$.

(vii) Check to see whether $f_{i,j}(x)$ follows the properties of an IMF.

(viii) If $f_{i,j+1}(x)$ meets the IMF properties as per step (vii), then take $f_i(x)$ = $f_{i,j+1}(x)$; set $s_{i+1}(x)$ = $s_i(x)$ - $f_i(x)$, and $i = i+1$; go to step (ix). Otherwise, set $j = j+1$, go to step (iii) and continue up to step (viii).

(ix) Find out the number of extrema points (maxima and minima together) (NEP), denoted as ε_i^1, in $s_i(x)$. If ε_i^1 is less than the extrema threshold (ET), ε^{1T}, the residue (R), $r(x)$ = $s_i(x)$; and the decomposition is complete. Otherwise, go to step (ii) and continue up to step (ix) [101, 104, 89, 124].

As illustrataed previously, the process of extracting the IMFs one by one is known as sifting process (SP) or simply as "sifting" [101]. It is to be noted that this is the same technique employed by various extensions of EMD as will be evident in the subsequent chapters. The sifting algorithm gradually separates the intrinsic image components having the highest to the lowest local spatial variations/oscillations of the data. Furthermore, the second and subsequent source image/data are also known as intermediate residues(IRs), $r_i(x)$, and for that reason, if there are k IMFs in a decomposition, then $s_{i+1}(x)$ = $r_i(x)$ for $i \in 1 : k$ and $r_k(x)$ = $r(x)$ [104, 89].

To simplify the EMD discussion and the subsequent ones, we will designate the first IMF, $f_1(x)$, as IMF-1; the second IMF, $f_2(x)$ as IMF-2, and so on. In a similar fashion, the first ISIMF of the first IMF, $f_{1,1}(x)$, will be designated as ISIMF-1:1; the second ISIMF of the first IMF, $f_{1,2}(x)$, as ISIMF-1:2; the first ISIMF of the second IMF, $f_{2,1}(x)$, as ISIMF-2:1, and so on. On the other hand, the first iteration, j = 1, is called Iteration-1; the second iteration, j = 2, is called Iteration-2, and so on. Several authors, particularly Bhuiyan and Huang [104, 89] also employed these notations. Furthermore, IMFs and the residue of a signal are collectively referred to as empirical mode functions (EMFs), denoted by $c_i(x)$, where $c_i(x)$ = $f_i(x)$ for i = 1 : k and $c_i(x)$ = $r(x)$ for i = k + 1. In this case, the first EMF, $c_1(x)$, is called EMF-1; the second EMF, $c_2(x)$, is called EMF-2, and so on. If no processing is done on the EMFs, the summation of all the EMFs returns the original data/signal back as given by

$$\sum_{i=1}^{k+1} c_i(x) = \widetilde{d}(x), \tag{5.14}$$

where ideally, $\widetilde{d}(x) = d(x)$. Practically, these two quantities will differ only by the round-off and truncation errors introduced by the execution environment, manipulating software or computer [104]. In addition, an orthogonality index (OI), denoted by θ^1, can be defined for the EMFs as done by Huang et al. [89], and it is given by

$$\theta^1 = \sum_{x=1}^{m} \left\{ \sum_{i=1}^{k+1} \sum_{j=1}^{k+1} \frac{c_i(x)c_j(x)}{\widetilde{d}(x)^2} \right\}. \tag{5.15}$$

It is to be noted here that a low value of OI indicates a good decomposition in terms of local orthogonality among the EMFs. In general, OI values less than or equal to 0.1 are acceptable [104, 106].

Going back to the discussion of the classical EMD process, obtaining the maxima and minima maps was one of the key steps. A close look at step (iii) of the process would reveal how this is accomplished. In fact, maxima and minima maps are generally obtained using a sliding window, which is known in the literature as the neighboring window method (NWM). In this method, a data point is considered a local maximum if its value is strictly higher than all other data points within the window. In the same manner, a data point is considered a local minimum if its value is strictly lower than all other data points within the window [104, 106, 107]. Even though the maxima and minima maps are dependent on the window size, a 1×3 window produces the most optimum maxima and minima maps. In addition to the neighboring window method, local derivatives can also be employed to find a local maximum and/or a local minimum point. All other points, except the maxima or minima points, of the maxima or minima maps are filled with zeros. On the other hand, an interpolation method known as the cubic spline (CS) is most widely used to interpolate the local maxima and minima points in order to obtain the UE and LE, respectively, in step (iv) of the EMD process [104, 106, 107, 89].

In the EMD process above, some of the steps can be accomplished using more than one methodologies. For instance, although various methodologies can be employed to verify the properties of an IMF in step (vii) of the above EMD process, the most used method is to calculate a deviation ratio (DR), denoted by $\delta^1_{i:j,j+1}$ [89, 2]. The DR, which is, in some sense, similar to mean squared error (MSE), can be defined in several ways as demonstrated below [109, 89]

$$\delta^1_{i:j,j+1} = \sum_{x=1}^{m} \frac{\left|f_{i,j+1}(x) - f_{i,j}(x)\right|^2}{\left|f_{i,j}(x)\right|^2}, \tag{5.16}$$

$$\delta^1_{i:j,j+1} = \sum_{x=1}^{m} \frac{\left|f_{i,j+1}(x) - f_{i,j}(x)\right|^2}{\left|f_{i,j+1}(x)\right|^2}, \tag{5.17}$$

$$\delta^1_{i:j,j+1} = \frac{\sum_{x=1}^{m} \left|f_{i,j+1}(x) - f_{i,j}(x)\right|^2}{\sum_{x=1}^{m} \left|f_{i,j}(x)\right|^2}. \tag{5.18}$$

A quick note on the above three equations is that even though all of them provide a global measure of the DR, The last equation causes the fastest convergence while the second one results in the slowest convergence. It should be further noted here that the definition of DR affects the number of required iterations to achieve a given DR threshold (DRT) as well as the amount of sifting per each IMF. It, however, does not have any contribution to the calculation of UE, LE or ME in a particular iteration [89, 2, 110]. In addition, if the calculated DR in step (vii) is within the DRT, δ^{1T}, then the corresponding IMF is assumed to have the required IMF properties. This reality will make more sense in the subsequent discussions. Typically, a low DRT, δ^{1T} , (e.g., below 0.5 for the second equation, and below 0.05 for the first and last quation) is chosen to ensure nearly zero envelope mean of the IMF [2, 124]. In some cases, sifting for each IMF is completed before fully satisfying the DRT criterion in order to avoid unexpected IMF characteristics due to over sifting (e.g., constant amplitude frequency modulated signals) [124, 101]. In such cases, the sifting is done up to a maximum number of allowable iterations (MNAI), denoted as j^m. On the other hand, the most appropriate value for the extrema threshold (ET) for 1D signal, ε^{1T} , is 2 [124, 101, 104].

5.5 Huang's Stopping Criteria and Boundary Adjustment

As mentioned previously, the Empirical mode decomposition (EMD), developed by Norden E. Huang et al. [89] in 1998 is a method of breaking down a signal without leaving a time domain and, in some sense, serves the same purpose as the Fourier Transforms and wavelet decomposition image analysis techniques. It became popular in the signal processing and time analysis domains. This popularity stemmed from the fact that the basic functions used to decompose a signal are not predefined but adaptively derived from the signal itself [108, 109, 110, 99, 100].

Nevertheless, caution has to be taken when any attempt is made to switch domains of application. This is due, in large part, to the fact that important questions, application domain being one of these, have to answered and adequately addressed. In addition to the domain question, interpolation algorithm, along with the associated stopping criteria and boundary effect has to be addressed. Huang et al. used cubic splines fitting and proposed interesting stopping criteria and boundary adjustment [89].

In his work, Huang considered the sifting process to be crucial to the effectiveness of the EMD. In fact, it is the sifting algorithm that determines the quality of the output. However, since his goal was to allow it to separate the finest local mode from the data first based only on the time scale, he defined two important desired t effects: First is the elimination of the riding waves and the second is the smoothing of uneven amplitude. The two conditions are necessary for a meaningful instantaneous frequency and a large disparity in the neighboring wave amplitudes, respectively. To guarantee that the IMF components retain enough physical sense of both amplitude and frequency modulation, Huang proposed a stopping criterion using a standard deviation. In essence, he limited the size of standard deviation, SD, computed from the two consecutive sifting results as

$$SD = \sum_{t=0}^{T} \left[\frac{(}{h_{1(k-1)}(t)} - h_{1k}(t))h_{1(k-1)}^{2}(t) \right], \qquad (5.19)$$

where the two optimal values for SD can be set between 0.2 and 0.3.

Another issue that has to be dealt with is the boundary effect. During the interpolation procedure, data interpolation gets complicated when it gets to the end of the given window. This is because the values at the end do not have enough neighbors that they can be compared to, and as such create a situation where, if not handled properly, can negatively affect the interpolation output. Huang proposed and defined two extremas amplitude at the end of the window to deal with this effect, and this seems to work well with his domain of choice, which is time domain [89].

5.6 Conclusion

This chapter summarizes the empirical mode decomposition so as to demonstrate its worth in image analysis and feature extraction. In doing so, various non-stationary data processing methods are reviewed and their weaknesses and strengths pointed out [99, 100, 105, 133]. It was mentioned that most of the non-stationary data are useful to a certain degree depending on the specific need. Nonetheless, the efficiency and adaptability of the empirical mode decomposition made it stand out from the rest [134, 135, 136, 137]. In addition, extending it to two-dimensional and combining it with the appropriate interpolation technique makes it suitable for application to image processing, particularly feature extraction [138, 139, 140, 141]. Fundamentally, the significance of the empirical mode decomposition lies in the decomposition of a given signal into its intrinsic mode function through the sifting process [142, 143, 144].

Chapter 6

BIDIMENSIONAL EMPIRICAL MODE DECOMPOSITION

6.1 Introduction

Bidimensional empirical mode decomposition (BEMD) describes the empirical mode decomposition (EMD) technique for image or two dimensional data processing. Based on the properties of the empirical mode decomposition (EMD) and characteristics of images, the EMD technique has been extended to analyze images or two dimensional data. For this reason, the bidimensional empirical mode decomposition (BEMD) is also known as image EMD (IEMD), 2D EMD, and directional (DEMD), among other designations [2, 89]. The BEMD decomposes an image or two-dimensional (2D) data into its 2D or bidimensional IMFs (BIMFs) and a 2D or bidimensional residue (BR), which represents the characteristic of local spatial scales at various levels of the image or 2D data, defined by the BIMFs or the BR. Because of the inherent nonlinearity and non-stationarity in images, BEMD presents a legitimate promise for image processing tool [2, 89].

In regard to the sifting algorithm for bidimensional signals, a similar algorithm as that described in Chapter 6 can be applied while the fundamental question to answer is what interpolation technique to use in the sifting process (SP) and how many iterations to consider in the SP to build the IMFs [132, 127, 125]? For the purpose of this dissertation, an efficient and robust interpolation method, known as the bicubic interpolation, is applied. Details of this algorithm were given in Chapter 4.

6.2 Bidemsional IMFs and Residue

In this section, the two dimensional effect of the intrinsic mode functions (IMFs) and residue (R) is discussed and the sifting process, that reflects the two dimensional behavior, is revisited. It suffice, therefore, to note that the properties of the bidimensional IMFs

(BIMFs) are slightly different from the IMFs discussed in Chapter 6. As such, the bidimensional intrinsic mode functions (BIMFs) only adheres to the last two properties of IMFs [132, 101].

The two properties that BIMFs followed are summarized below:

(i) At any point, the mean value ($Env_{mean,j-1}(t)$), of the upper envelope ($Env_{max,j-1}(t)$) and lower envelope ($Env_{min,j-1}(t)$), obtained from the local maxima points and the local minima points of the BIMF respectively, is zero or nearly zero.

(ii) The BIMFs are locally orthogonal among each other and as a set

Furthermore, it should be noted here that, due to the properties of an image and the BEMD process, the first two properties (i and ii) given for the IMFs in chapter 6 cannot be satisfied because the maxima and minima points are defined in a two dimensional setting for an image. For the same reason, it is also difficult, if not impossible, to define and/or to achieve any characteristic relationships between the number of maxima points and the number of minima points for BIMFs [101]. As is the case for IMFs, the orthogonality property is supplemental and/or desirable feature, whose fulfillment is not vital. Furthermore, the bidimensional residue (BR) provides the trend of the image or two dimensional data that usually does not have more than one local maximum and/or one local minimum point [132, 101, 104].

6.3 Process of the Classical BEMD

To simplify the discussion of the BEMD process, let the original image or 2D data be denoted by $I(x,y)$, the ith BIMF by $F_i(x,y)$, and the BR by $R(x,y)$. For a $m \times n-$pixel image, $x \in 1:m$ and $y \in 1:n$, where m and n are the total number of pixels in the horizontal and vertical directions, respectively. In order to obtain BIMF, $F_i(x,y)$ in the classical and/or other standard BEMD processes, multiple iterations are required [101]. Additionally, the intermediate state of a BIMF (ISBIMF) in the jth iteration can be denoted as $F_{i,j}(x,y)$. During this process, $F_i(x,y)$ is obtained from its source image $S_i(x,y)$. With the above definitions, the basic steps of a classical EMD method can be extended and/or modified in order to help us define the steps of a classical BEMD process as given below [132, 101].

(i) Set $i = 1$, and $S_i(x,y)$ = $I(x,y)$. If $S_i(x,y)$ (i.e., the given image $I(x,y)$) is not the only component, with $S_i(x,y)$ having the BR properties, then go to step two.

(ii) Set $j = 1$, and $F_{i,j}(x,y)$ = $S_i(x,y)$.

(iii) Obtain the local maxima points of $F_{i,j}(x,y)$, which is known as the 2D maxima map and denoted by $P_{i,j}(x,y)$. Similarly, obtain the local minima points of $F_{i,j}(x,y)$, which is known as the 2D minima map and denoted by $Q_{i,j}(x,y)$.

(iv) Generate the upper envelope (UE), $U_{i,j}(x,y)$, and the lower envelope (LE), $L_{i,j}(x,y)$, of ISBIMF, $F_{i,j}(x,y)$, from the maxima points in $P_{i,j}(x,y)$ and minima points in $Q_{i,j}(x,y)$, respectively.

(v) Find the mean envelope (ME) as $M_{i,j}(x,y) = \frac{U_{i,j}(x,y)+L_{i,j}(x,y)}{2}$

(vi) Calculate $F_{i,j}(x,y)$ as $F_{i,j+1}(x,y)$ = $F_{i,j}(x,y)$ - $M_{i,j}(x,y)$.

(vii) Check to see whether $F_{i,j}(x,y)$ follows the properties of a BIMF.

(viii) If $F_{i,j+1}(x,y)$ meets the BIMF properties as per step (vii), then take $F_i(x,y)$ = $F_{i,j+1}(x,y)$; set $S_{i+1}(x,y)$ = $S_i(x,y)$ - $F_i(x,y)$, and $i = i+1$; go to step (ix). Otherwise, set $j = j+1$, go to step (iii) and continue up to step (viii).

(ix) Find out the number of extrema points (maxima and minima together) (NEP), denoted as ε_i^2, in $S_i(x,y)$. If ε_i^2 is less than the extrema threshold (ET), ε^{2T}, the BR, $R(x,y)$ = $S_i(x,y)$; and the decomposition is complete. Otherwise, go to step (ii) and continue up to step (ix) [101, 104].

As illustrataed previously, the process of extracting the BIMFs one by one is known as sifting process (SP) or simply as "sifting" [101]. It is to be recalled that this is the same technique employed by EMD. The sifting algorithm gradually separates the intrinsic image components having the highest to the lowest local spatial variations/oscillations of the data. Furthermore, the second and subsequent source image/data are also known as intermediate bidimensional residues(IBRs), $R_i(x,y)$ [104]. As such, if there are k BIMFs in a decomposition, then $S_{i+1}(x,y) = R_i(x,y)$ for $1 \in 1 : k$ and $R_k(x,y) = R(x,y)$

To simplify the BEMD discussion and the subsequent ones, we will designate the first BIMF, $F_1(x,y)$, as BIMF-1; the second BIMF, $F_2(x,y)$ as BIMF-2, and so on. In a similar fashion, the first ISBIMF of the first BIMF, $F_{1,1}(x,y)$, will be designated as ISBIMF-1:1; the second ISBIMF of the first BIMF, $F_{1,2}(x,y)$, as ISBIMF-1:2; the first ISBIMF of the second BIMF, $F_{2,1}(x,y)$, as ISBIMF-2:1, and so on. In the similar manner, the first source image, $S_1(x,y)$, is called SI-1, the second source image, $S_2(x,y)$, is called SI-2, and so on. Furthermore, the first IBR, $R_1(x,y)$, is called IBR-1, the second IBR, $R_2(x,y)$, is called IBR-2, and so on. Several authors, particularly Huang and Bhuiyan [104, 89] also employed these notations.

In addition, BIMFs and the residue of an image, or 2D data matrix, are collectively referred to as bidimensional empirical mode functions (BEMFs), denoted by $C_i(x,y)$, where $C_i(x,y) = F_i(x,y)$ for i = 1 : k and $C_i(x,y) = R(x,y)$ for i = k + 1. In this case, the first BEMF, $C_1(x,y)$, is called BEMF-1; the second BEMF, $C_2(x,y)$, is called BEMF-2, and so on. It should be noted that BEMF-1 is considered as the lowest order component and BEMF-k (final BEMF or the BR) is considered as the highest order component of the image derived by BEMD method [104]. If no processing is done on the BEMFs, the summation of all the BEMFs returns the original data/signal back as given by

$$\sum_{i=1}^{k+1} C_i(x,y) = \widetilde{I}(x,y), \tag{6.1}$$

where ideally, $\widetilde{I}(x,y) = I(x,y)$. Practically, these two quantities will differ only by the round-off and truncation errors introduced by the execution environment, manipulating software or computer [104]. In addition, an orthogonality index (OI), denoted by θ^2, can be defined for the BEMFs as done by Huang et al. [89], and it is given by

$$\theta^2 = \sum_{x=1}^{m} \sum_{y=1}^{n} \left\{ \sum_{i=1}^{k+1} \sum_{j=1}^{k+1} \frac{C_i(x,y)C_j(x,y)}{\widetilde{I}(x,y)^2} \right\}. \tag{6.2}$$

It is to be noted here that a low value of OI indicates a good decomposition in terms of local orthogonality among the BEMFs. In general, OI values less than or equal to 0.1 are acceptable for BEMFs as they were for EMFs [104, 106, 131].

Going back to the discussion of the classical BEMD process above, obtaining the maxima and minima maps was one of the key steps. A close look at step (iii) of the process would reveal how this is accomplished. In fact, and as was the case for the EMD method, maxima and minima maps are obtained using a 2D sliding window, a process known in the literature as the neighboring window method (NWM). In this method, a data point is considered a local maximum if its value is strictly higher than all other data points within the window. In the same manner, a data point is considered a local minimum if its value is strictly lower than all other data points within the window [104, 106, 107]. Even though the maxima and minima maps are dependent on the window size, a 3×3 window produces the most optimum maxima and minima maps. Nonetheless, a higher window size may be used in some applications, although this may result in a lower, if not equal, number of local extrema points for a given data matrix. All other points, except the maxima or minima points, of the maxima or minima maps are filled with zeros. On the other hand, a family of interpolators commonly referred to as radial basis function (RBF), such as the thin-plate spline (TPS) are often used to interpolate the local maxima and minima points in order to obtain the UE and LE, respectively, in step (iv) of the BEMD process [104, 106, 107, 89].

In the EMD process above, some of the steps can be accomplished using more than one methodologies. For instance, although various methodologies can be employed to verify the properties of a BIMF in step (vii) of the above BEMD process, the most used method is to calculate the deviation ratio (DR), denoted by $\delta^2_{i:j,j+1}$ [89, 2]. The DR for BEMD, which is, in some sense, similar to mean squared error (MSE), can be defined in several ways as demonstrated below [109, 89]

$$\delta^2_{i:j,j+1} = \sum_{x=1}^{m}\sum_{y=1}^{n} \frac{\left|F_{i,j+1}(x,y) - F_{i,j}(x,y)\right|^2}{\left|F_{i,j}(x,y)\right|^2}, \tag{6.3}$$

$$\delta^2_{i:j,j+1} = \sum_{x=1}^{m}\sum_{y=1}^{n} \frac{\left|F_{i,j+1}(x,y) - F_{i,j}(x,y)\right|^2}{\left|F_{i,j+1}(x,y)\right|^2}, \tag{6.4}$$

$$\delta^2_{i:j,j+1} = \frac{\sum_{x=1}^{m}\sum_{y=1}^{n}\left|F_{i,j+1}(x,y) - F_{i,j}(x,y)\right|^2}{\sum_{x=1}^{m}\sum_{y=1}^{n}\left|F_{i,j}(x,y)\right|^2}. \tag{6.5}$$

A quick note on the above three equations is that even though all of them provide a global measure of the DR, The last equation causes the fastest convergence while the second one results in the slowest convergence. It should be further noted here that the definition of DR affects the number of required iterations to achieve a given DR threshold (DRT) as well as the amount of sifting per each BIMF. It, however, does not have any contribution to the calculation of UE, LE or ME in a particular iteration [89, 2, 110].

In addition, if the calculated DR in step (vii) is within the DRT, δ^{2T}, then the corresponding BIMF is assumed to have the required BIMF properties. This reality will make more sense in the subsequent discussions. Typically, a low DRT, δ^{2T} , (e.g., below 0.5 for the second equation, and below 0.05 for the first and last quation) is chosen to ensure nearly zero envelope mean of the IMF [2, 124]. In some cases, sifting for each BIMF is completed before fully satisfying the DRT criterion in order to avoid unexpected BIMF characteristics due to over sifting (e.g., constant amplitude frequency modulated signals) [124, 101]. In such cases, the sifting is done up to a maximum number of allowable iterations (MNAI), denoted as j^m, to serve as an additional stopping criterion.

It is worth mentioning that the surface interpolation algorithms require at least three maxima or three minima points to estimate the corresponding UE or LE. Therefore, the sifting or iterations are terminated for a BIMF before achieving the DRT when the maxima or minima map contains less than three maxima or minima. On the other hand, the most appropriate value for the extrema threshold (ET) for 2D data or image, ε^{2T}, is 2 [124, 101, 104].

6.4 The FABEMD Approach

Upon realizing the benefit of Huang's proposed methodology, several authors including Adhami et al., Nunes and Bhuiyan [101, 145, 2] investigated its cross domain application and what they found wasn't disappointing.

Nunes et al. [145] tested it with image texture extraction and Bhuiyan et al. [101] followed suit by proposing an order statistic filters (OSFs)-based approach to analyze images. Because of it potential to be fast, they called this methodology Fast and Adaptive Bidimensional Empirical Mode Decomposition (FABEMD) [101]. The essence of the FABEMD is its application of a statistical methodology known as the order statistic filter. His interpolation method proceeds as followed:

For each local maximum point in $P_i^S(x,y)$ (i.e., for each non-zero element in $P_i^S(x,y)$), the Euclidean distance to the nearest other local maximum point (non-zero element) is calculated and stored in an array, called neighboring distance vector of the maxima, and it is denoted as λ_i^U. Similarly, for each local minimum point in $Q_i^S(x,y)$ (i.e., for each non-zero element in $Q_i^S(x,y)$), the Euclidean distance to the nearest other local minimum point (non-zero element) is calculated and stored in an array, called neighboring distance vector of the minima, and it is denoted as λ_i^L. The number of elements in λ_i^U or λ_i^L is equal to the number of maxima or minimum points in $P_i^S(x,y)$ or $Q_i^S(x,y)$, respectively. The distance values in λ_i^U and λ_i^L maybe arranged in ascending or descending order for easy selection of the quantity later. And instead of taking the distance to the nearest neighbor, other distances, which may have not yet been explored or tested properly, may also be utilized. But according to Bhuiyan, only the use of distance to the nearest neighbor for each local maximum or minimum points yield the desired separation of the characteristic local spatial scales or variation of an image into its BEMFs [101].

6.5 Conclusion

This chapter gives a reasonable description of the bidimensional empirical mode decomposition (BEMD). It should be recalled that the BEMD is the extension of the empirical mode decomposition (EMD) to two or more dimensions [125]. With this expansion, the sifting process of the EMD is slightly modified as a result. The BEMD method is a major part of this dissertation and its role in the research is clearly defined in the next chapter.

Chapter 7

BIDIMENSIONAL EMPIRICAL MODE DECOMPOSITION FOR EDGES, CORNERS, AND CURVES (BEMDEC)

7.1 Overview

Feature detection is a fundamental problem in the field of image processing because detecting feature of an image reveals more information that would otherwise be lost. Current approaches have, however, focused only on improving and tweaking some of the old and well known techniques in an effort to address their shortcomings. Though this is a decent endeavor, it is sometimes necessary to think outside the box in order to find new ways to address old problems. Because of the computational complexities and inaccuracies of these methodologies, attacking them from different angles is vital. This is because image use has been on the rise in recent years and successfully extracting their feature makes other data decoding easier [2, 70, 66, 65].

In recent years, the general interest in image processing has increased. As the result of this reality, successful extraction of feature is important, for it makes objects identification and extraction of information from natural images and scenes easier. Due to the nonlinearity and non-stationarity in images, bidimensional empirical mode decomposition (BEMD) techniques seem to hold great potential for image processing, because of their inherent ability to decompose an image into its bidimensional intrinsic mode functions (BIMFs) and a bidimensional residue (BR). Although a number of BEMD methods, effective to certain degrees, have been suggested and proposed, many of them seem to solve one problem at the expense of others. While it has been generally the goal to develop a scheme that provides an acceptable tradeoff, achieving it has been difficult. This reality motivates the underlying concept and formulation of bidimensional empirical mode decomposition for edges, corners and curves (BEMDEC) method presented in this dissertation.

7.2 BEMDEC Outline

The underlying concept of this dissertation is motivated by the lack of adaptive feature detection operator in general and a robust one in particular. In recent years, the utilization of image data has significantly increased, but the methods to analyze and understand such data are seriously deficient. The trends indicate that the increase in the image usage in various applications is permanent [66, 68]. For this reason, fresh concepts such as the one proposed herein should be considered good starting point but certainly not the end of the quest. Detailed explanation of the BEMDEC methodology, including the EMD/BEMD from which it was derived, and the primary interpolation algorithm, are given below.

7.3 EMD/BEMD Algorithm

It was emphasized in the previous two chapters that the EMD/BEMD algorithm plays a major role in the formulation of BEMDEC. Nonetheless, important details, such as how it works and what its limitations are, were intentionally reserved. Though the EMD/BEMD algorithm cannot be entirely reproduced due to the space constrains, simple explanation and illustrative examples are called for to understand where it fits in the overall scheme of the proposed method. For starter, a detailed version of the "sifting" process, which was introduced in Chapters 5 and 6, is given in the following itemized list and in the tables in algorithmic form. It should be noted that the original sifting process was summarized in aforementioned chapters [89, 106, 107]. The algorithm proceeds as followed for extraction of two IMFs in a series of iterations:

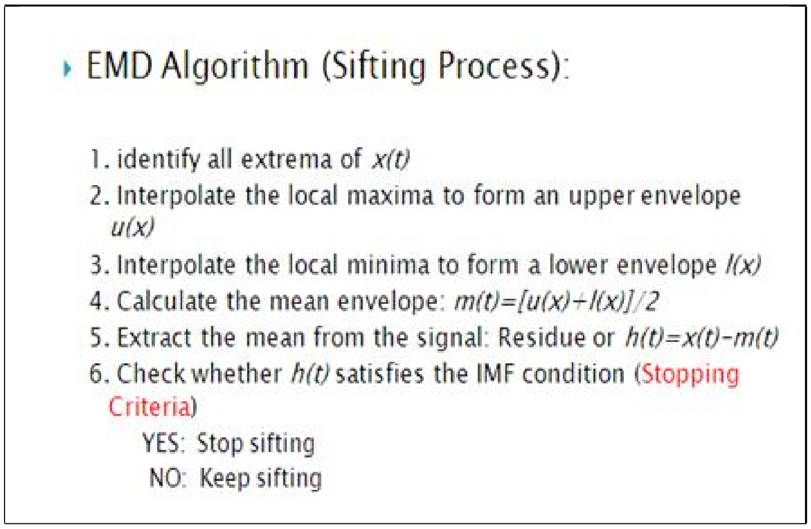
- Identify the positive peaks (maxima) and negative peaks (minima) of the original signal/image
- Construct the lower and the upper envelopes of the signal using some interpolation method
- .Calculate the mean values by averaging the upper envelope and the lower envelope

- Subtract the mean from the original signal to produce the first intrinsic mode function (IMF) component

- Calculate the first residual component by subtracting the first IMF from the original signal. Note that this residual component is treated as new data and is subjected to the same processes described above to calculate the next IMF

- Repeat the above steps until the final residual component becomes a monotonic function where no more IMFs can be extracted

Table 7.1: EMD Algorithm (Sifting Process - SP.)



EMD Algorithm (Sifting Process):

1. identify all extrema of $x(t)$
2. Interpolate the local maxima to form an upper envelope $u(x)$
3. Interpolate the local minima to form a lower envelope $l(x)$
4. Calculate the mean envelope: $m(t)=[u(x)+l(x)]/2$
5. Extract the mean from the signal: Residue or $h(t)=x(t)-m(t)$
6. Check whether $h(t)$ satisfies the IMF condition (Stopping Criteria)
   YES: Stop sifting
   NO: Keep sifting

Table 7.2: SP at the Initialization Where IMF = 1 and Iteration = 0.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
  while Residue not equal zero or not monotone
    while $I_i$ has non-negligible local mean
      U(t) = spline through local maxima of $I_i$
      L(t) = spline through local minima of $I_i$
      Av(t) = 1/2 (U(t) + L(t))
      $I_i(t)$ = $I_i(t)$ - Av(t)
      i = i + 1
    end
    $IMF_k(t)$ = $I_i(t)$
    Residue = Residue - $IMF_k$
    k = k+1
  end
```

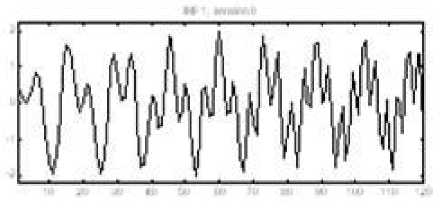


Table 7.3: SP at the Local Maxima Where IMF = 1 and Iteration = 0.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
  while Residue not equal zero or not monotone
    while $I_i$ has non-negligible local mean
      U(t) = spline through local maxima of $I_i$
      L(t) = spline through local minima of $I_i$
      Av(t) = 1/2 (U(t) + L(t))
      $I_i(t)$ = $I_i(t)$ - Av(t)
      i = i + 1
    end
    $IMF_k(t)$ = $I_i(t)$
    Residue = Residue - $IMF_k$
    k = k+1
  end
```

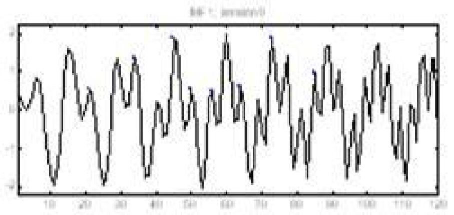


Table 7.4: SP at the End of Local Maxima Where IMF = 1 and Iteration = 0.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t)$ = $I_i(t)$ - Av(t)
            i = i + 1
        end
        $IMF_k(t)$ = $I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

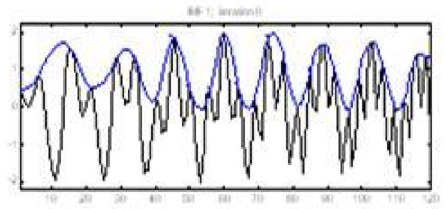


Table 7.5: SP at the Local Minima Where IMF = 1 and Iteration = 0.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t)$ = $I_i(t)$ - Av(t)
            i = i + 1
        end
        $IMF_k(t)$ = $I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

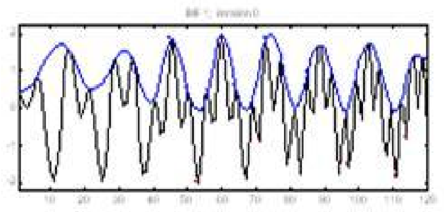


Table 7.6: SP at the End of the Local Minima Where IMF = 1 and Iteration = 0.

```
Residue = s(t)
I_1(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while I_i has non-negligible local mean
            U(t) = spline through local maxima of I_i
            L(t) = spline through local minima of I_i
            Av(t) = 1/2 (U(t) + L(t))
            I_i(t) = I_i(t) - Av(t)
            i = i + 1
        end
        IMF_k(t) = I_i(t)
        Residue = Residue - IMF_k
        k = k+1
    end
```

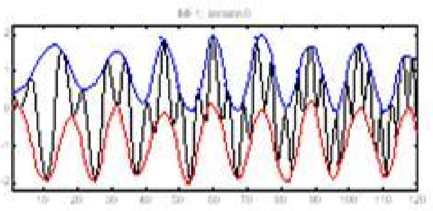


Table 7.7: SP at the Local Mean Where IMF = 1 and Iteration = 0.

```
Residue = s(t)
I_1(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while I_i has non-negligible local mean
            U(t) = spline through local maxima of I_i
            L(t) = spline through local minima of I_i
            Av(t) = 1/2 (U(t) + L(t))
            I_i(t) = I_i(t) - Av(t)
            i = i + 1
        end
        IMF_k(t) = I_i(t)
        Residue = Residue - IMF_k
        k = k+1
    end
```

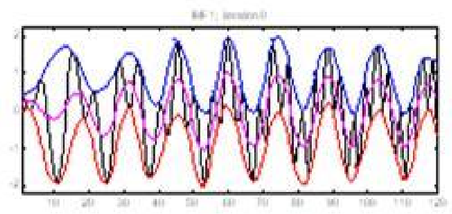


Table 7.8: SP with the Mean Substracted.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_{i}(t) = I_i(t) - Av(t)$
            i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

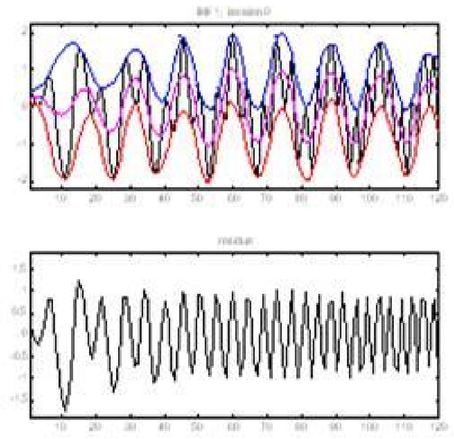


Table 7.9: SP Increment Step Where IMF = 1 and Iteration = 1.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t) = I_i(t) - Av(t)$
            i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

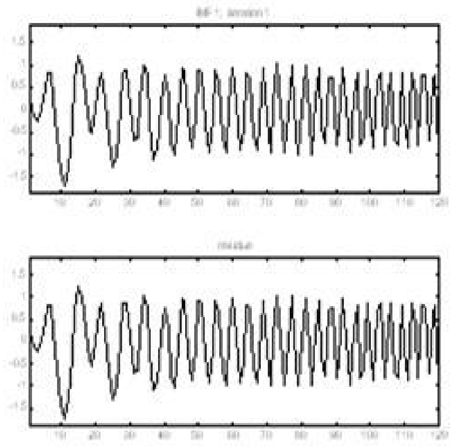
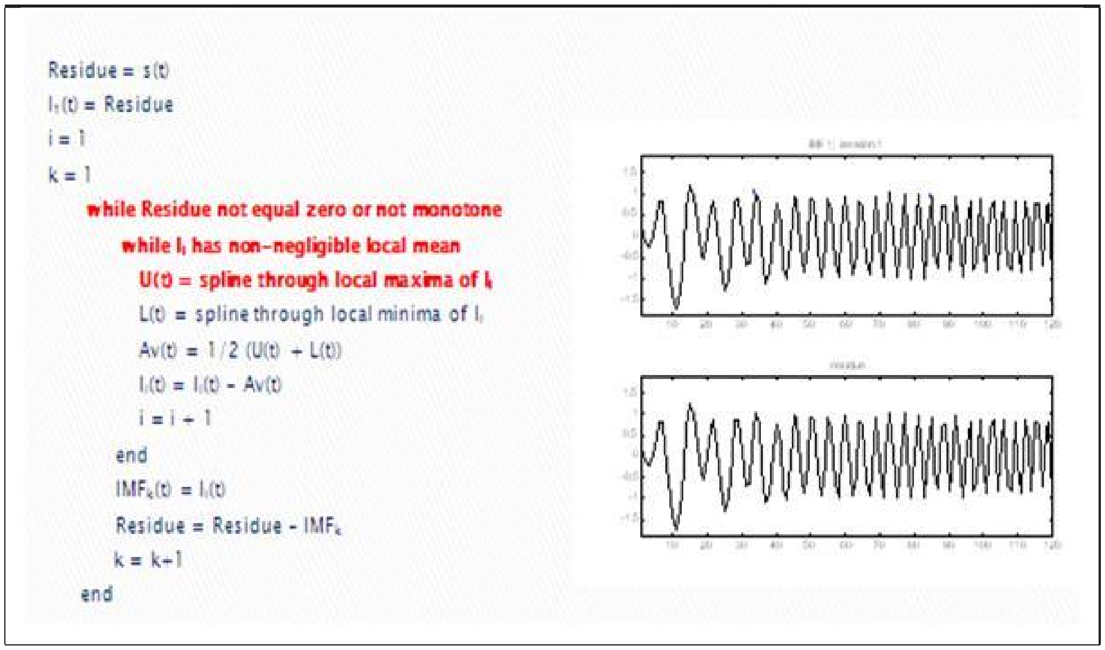
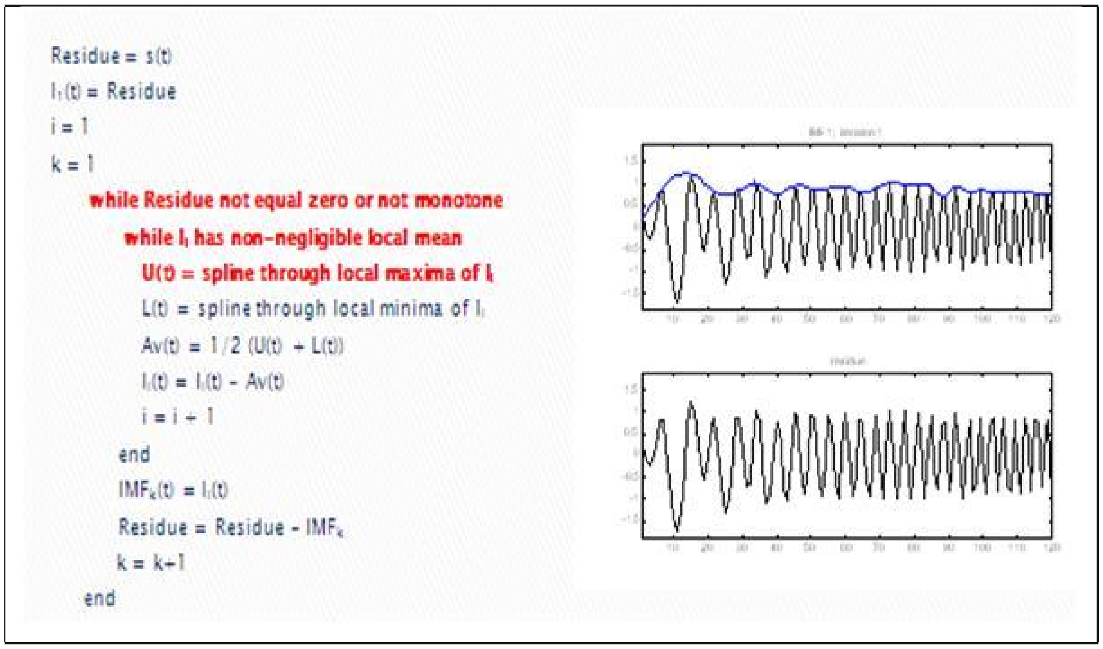


Table 7.10: SP at the Maxima with IMF = 1 and Iteration = 1.



```
Residue = $s(t)$
$l_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $l_i$ has non-negligible local mean
            $U(t)$ = spline through local maxima of $l_i$
            $L(t)$ = spline through local minima of $l_i$
            $Av(t) = 1/2\ (U(t) + L(t))$
            $l_i(t) = l_i(t) - Av(t)$
            i = i + 1
        end
        $IMF_k(t) = l_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

Table 7.11: SP at the End of the Maxima with IMF = 1 and Iteration = 1.



```
Residue = $s(t)$
$l_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $l_i$ has non-negligible local mean
            $U(t)$ = spline through local maxima of $l_i$
            $L(t)$ = spline through local minima of $l_i$
            $Av(t) = 1/2\ (U(t) + L(t))$
            $l_i(t) = l_i(t) - Av(t)$
            i = i + 1
        end
        $IMF_k(t) = l_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

Table 7.12: SP at the Minima with IMF = 1 and Iteration = 1.

```
Residue = s(t)
$I_1$(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
          U(t) = spline through local maxima of $I_i$
          L(t) = spline through local minima of $I_i$
          Av(t) = 1/2 (U(t) + L(t))
          $I_i$(t) = $I_i$(t) - Av(t)
           i = i + 1
        end
        $IMF_k$(t) = $I_i$(t)
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

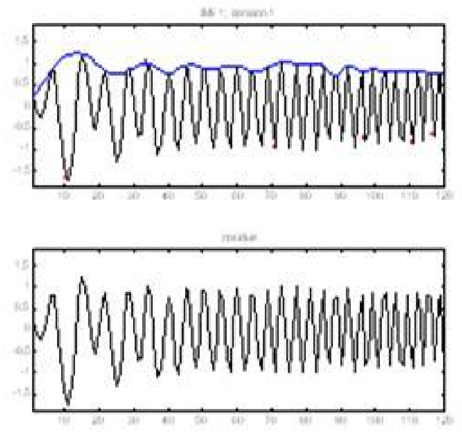


Table 7.13: SP at the End of the Minima with IMF = 1 and Iteration = 1.

```
Residue = s(t)
$I_1$(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
          U(t) = spline through local maxima of $I_i$
          L(t) = spline through local minima of $I_i$
          Av(t) = 1/2 (U(t) + L(t))
          $I_i$(t) = $I_i$(t) - Av(t)
          i = i + 1
        end
        $IMF_k$(t) = $I_i$(t)
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

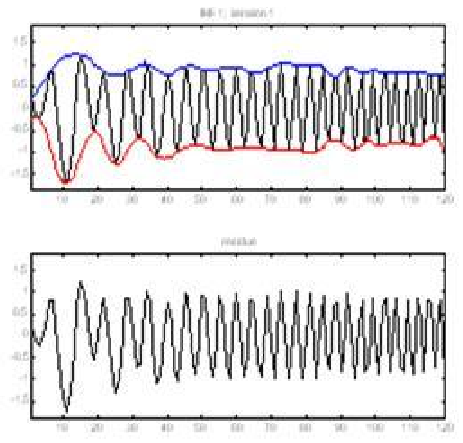


Table 7.14: SP at the Local Mean with IMF = 1 and Iteration = 1.

```
Residue = $s(t)$
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
          $U(t)$ = spline through local maxima of $I_i$
          $L(t)$ = spline through local minima of $I_i$
          $Av(t) = 1/2\ (U(t) + L(t))$
          $I_i(t) = I_i(t) - Av(t)$
          i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

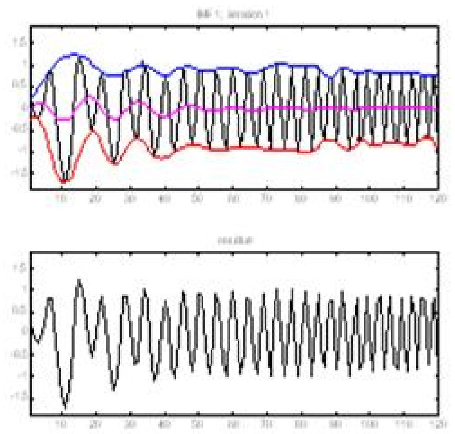


Table 7.15: SP where the Mean is Substracted with IMF = 1 and Iteration = 1.

```
Residue = $s(t)$
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
          $U(t)$ = spline through local maxima of $I_i$
          $L(t)$ = spline through local minima of $I_i$
          $Av(t) = 1/2\ (U(t) + L(t))$
          $I_i(t) = I_i(t) - Av(t)$
          i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

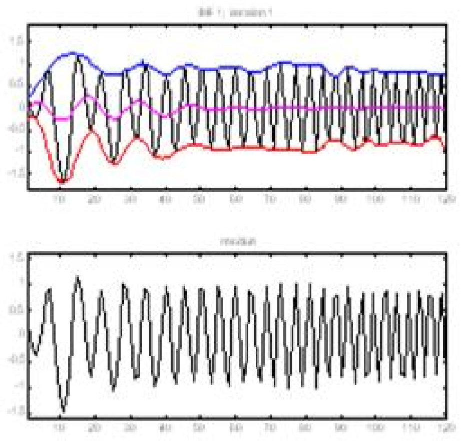
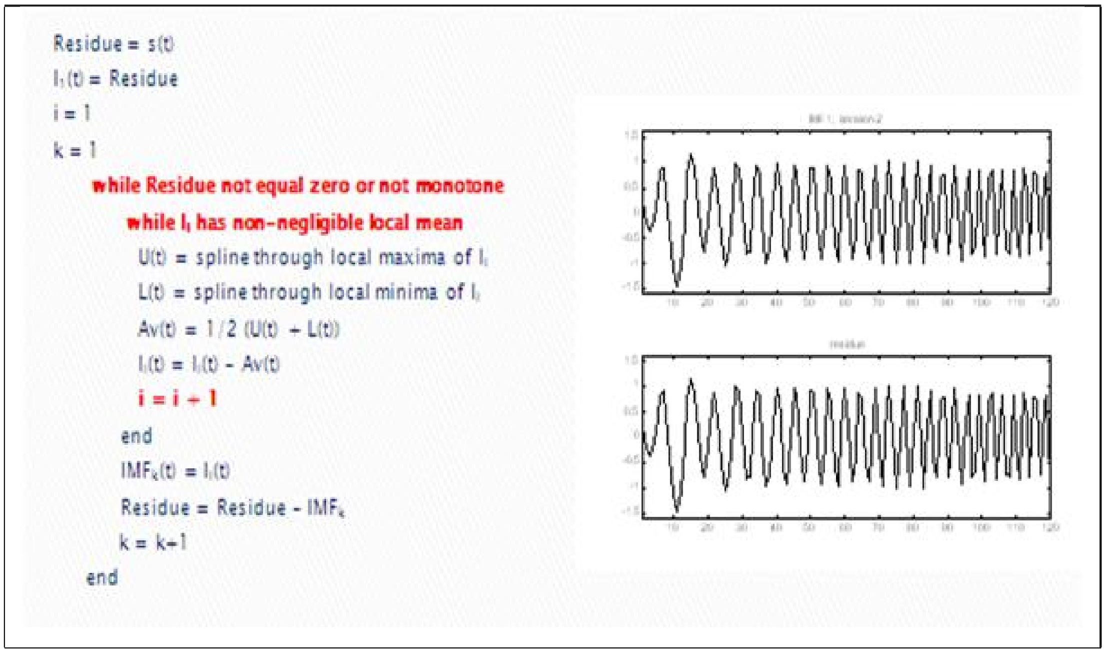
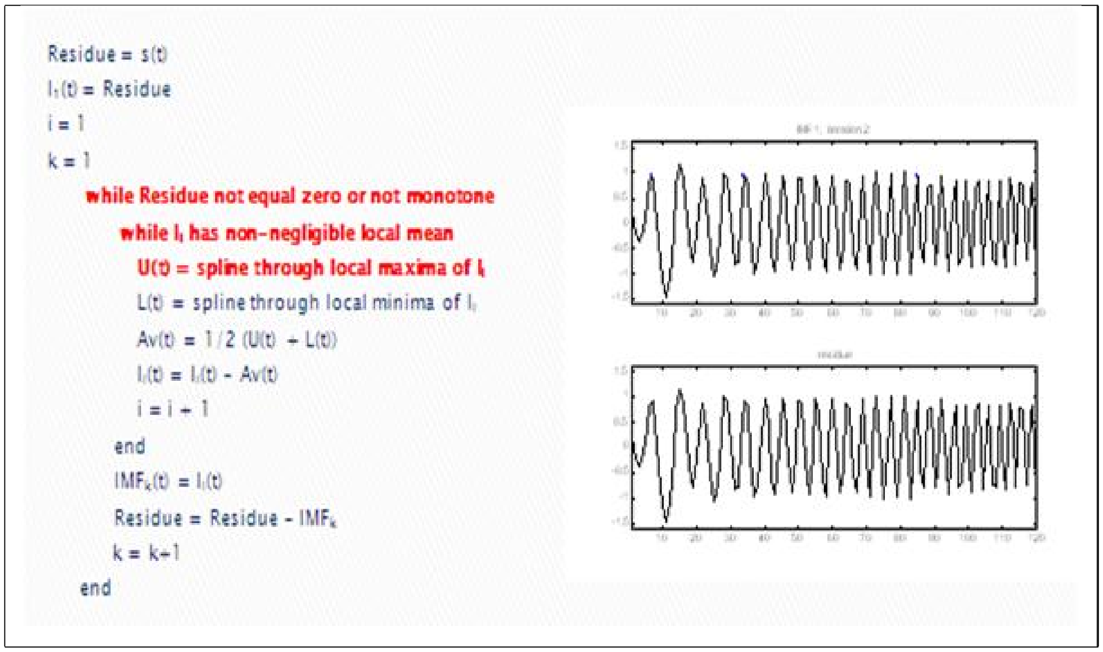


Table 7.16: SP Increment Step Where IMF = 1 and Iteration = 2.



```
Residue = s(t)
I_1(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while I_i has non-negligible local mean
          U(t) = spline through local maxima of I_i
          L(t) = spline through local minima of I_i
          Av(t) = 1/2 (U(t) + L(t))
          I_i(t) = I_i(t) - Av(t)
          i = i + 1
        end
        IMF_k(t) = I_i(t)
        Residue = Residue - IMF_k
        k = k+1
    end
```

Table 7.17: SP at the Local Maxima with IMF = 1 and Iteration = 2.



```
Residue = s(t)
I_1(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while I_i has non-negligible local mean
          U(t) = spline through local maxima of I_i
          L(t) = spline through local minima of I_i
          Av(t) = 1/2 (U(t) + L(t))
          I_i(t) = I_i(t) - Av(t)
          i = i + 1
        end
        IMF_k(t) = I_i(t)
        Residue = Residue - IMF_k
        k = k+1
    end
```

Table 7.18: SP at the End of the Local Maxima with IMF = 1 and Iteration = 2.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t)$ = $I_i(t)$ - Av(t)
            i = i + 1
        end
        $IMF_k(t)$ = $I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

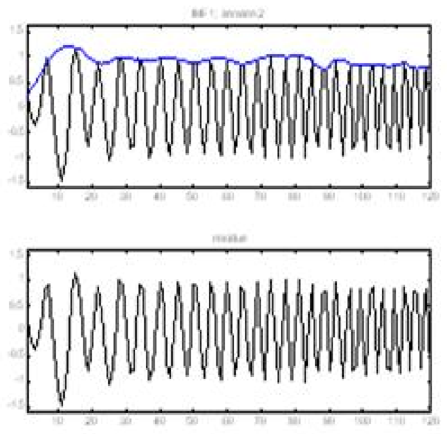


Table 7.19: SP at the Beginning of the Local Minima with IMF = 1 and Iteration = 2.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t)$ = $I_i(t)$ - Av(t)
            i = i + 1
        end
        $IMF_k(t)$ = $I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

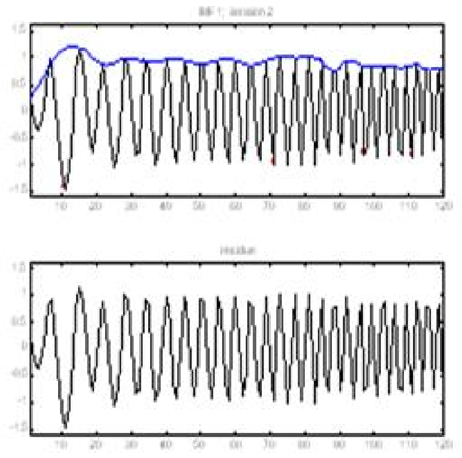
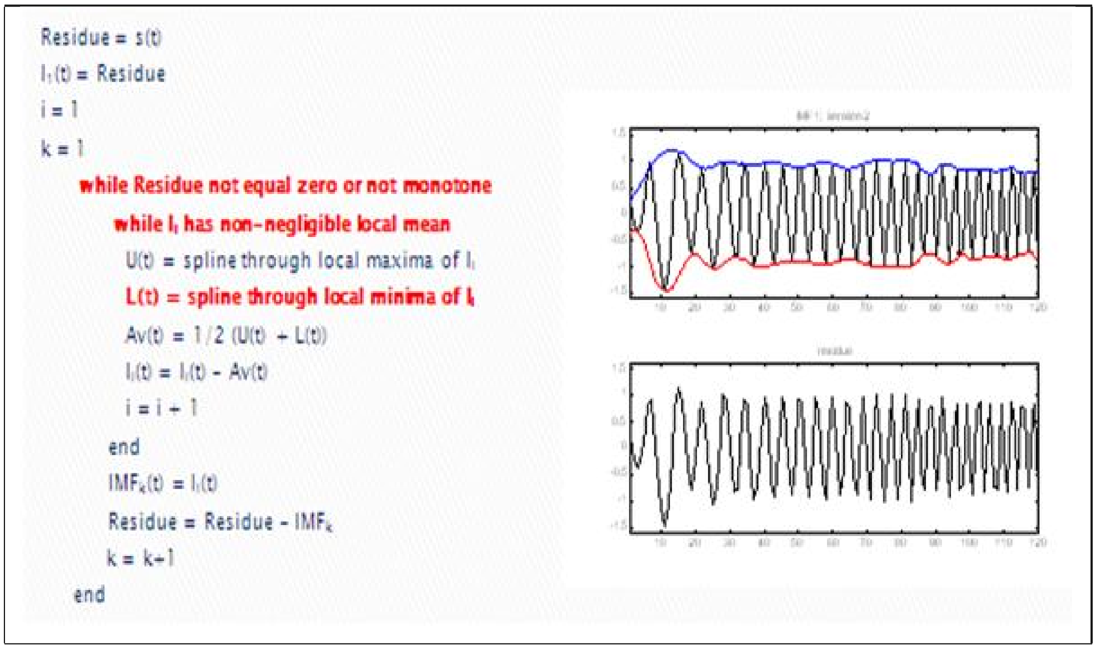
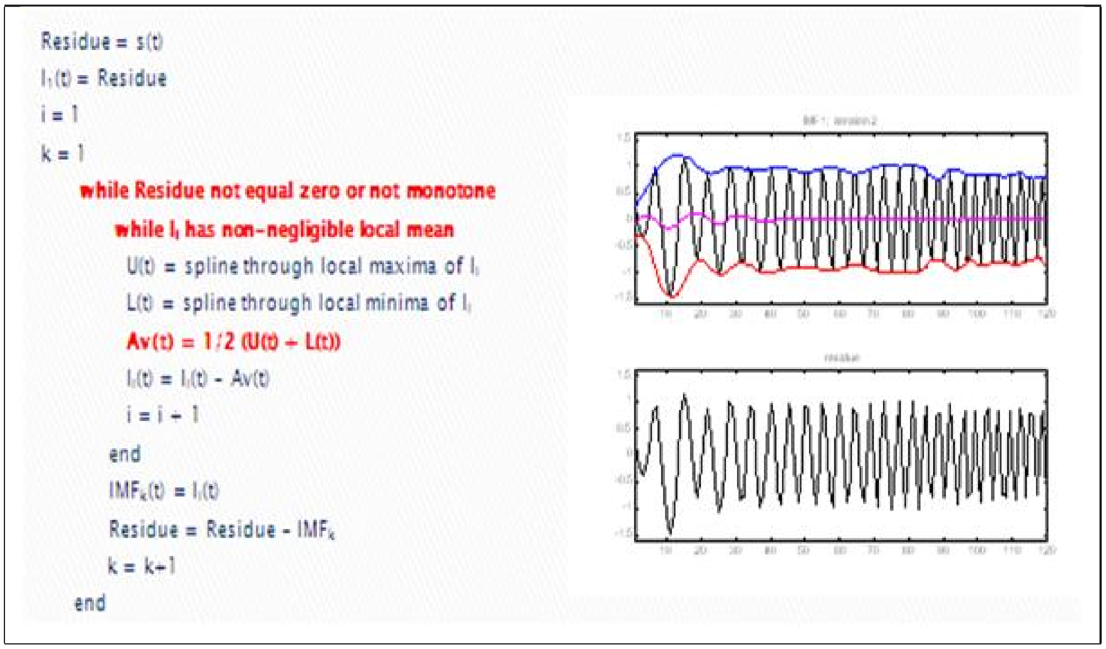


Table 7.20: SP at the End of the Local Minima with IMF = 1 and Iteration = 2.



```
Residue = s(t)
$l_1(t)$ = Residue
i = 1
k = 1
   while Residue not equal zero or not monotone
      while $l_i$ has non-negligible local mean
         U(t) = spline through local maxima of $l_i$
         L(t) = spline through local minima of $l_i$
         Av(t) = 1/2 (U(t) + L(t))
         $l_i(t) = l_i(t)$ - Av(t)
         i = i + 1
      end
      $IMF_k(t) = l_i(t)$
      Residue = Residue - $IMF_k$
      k = k+1
   end
```

Table 7.21: SP at the Local Mean with IMF = 1 and Iteration = 2.



```
Residue = s(t)
$l_1(t)$ = Residue
i = 1
k = 1
   while Residue not equal zero or not monotone
      while $l_i$ has non-negligible local mean
         U(t) = spline through local maxima of $l_i$
         L(t) = spline through local minima of $l_i$
         Av(t) = 1/2 (U(t) + L(t))
         $l_i(t) = l_i(t)$ - Av(t)
         i = i + 1
      end
      $IMF_k(t) = l_i(t)$
      Residue = Residue - $IMF_k$
      k = k+1
   end
```

Table 7.22: SP with the Mean Subtracted with IMF = 1 and Iteration = 2.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            $U(t)$ = spline through local maxima of $I_i$
            $L(t)$ = spline through local minima of $I_i$
            $Av(t) = 1/2\,(U(t) + L(t))$
            $I_i(t) = I_i(t) - Av(t)$
            i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

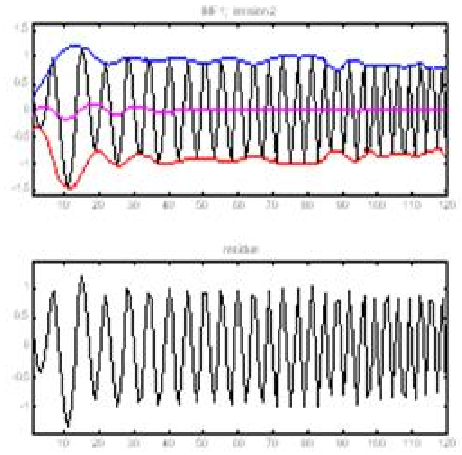


Table 7.23: SP at the End of the Loop with IMF = 1 and New Iteration = 3.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            $U(t)$ = spline through local maxima of $I_i$
            $L(t)$ = spline through local minima of $I_i$
            $Av(t) = 1/2\,(U(t) + L(t))$
            $I_i(t) = I_i(t) - Av(t)$
            i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

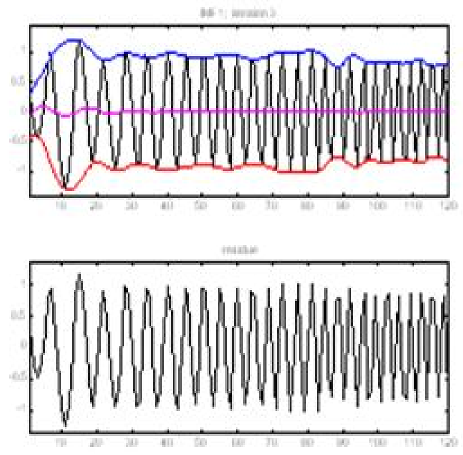
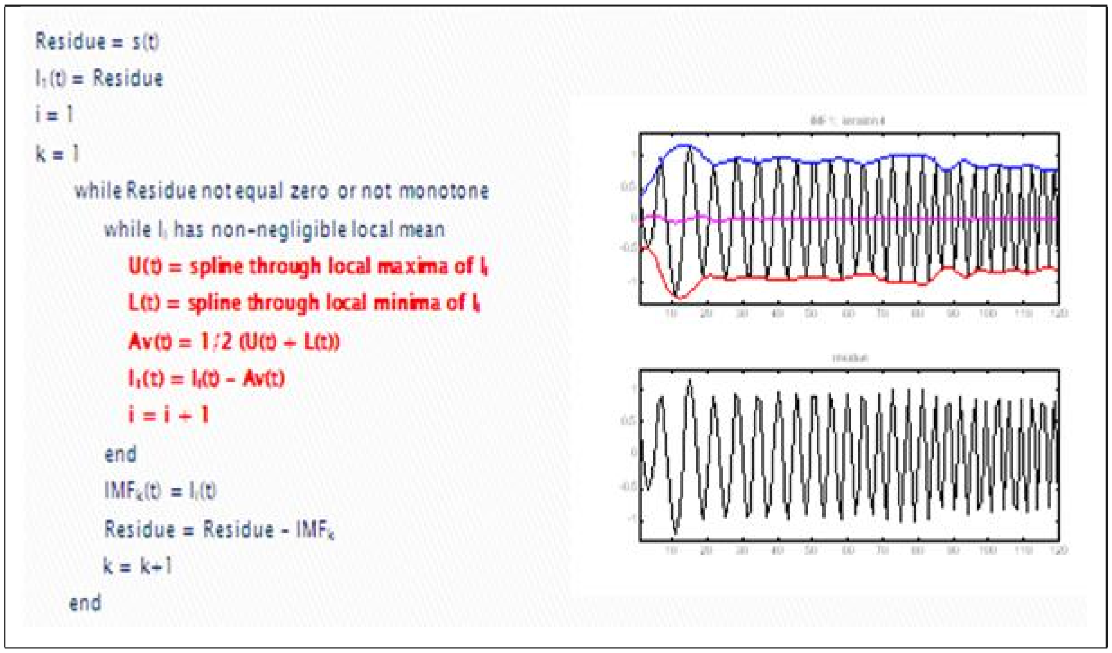
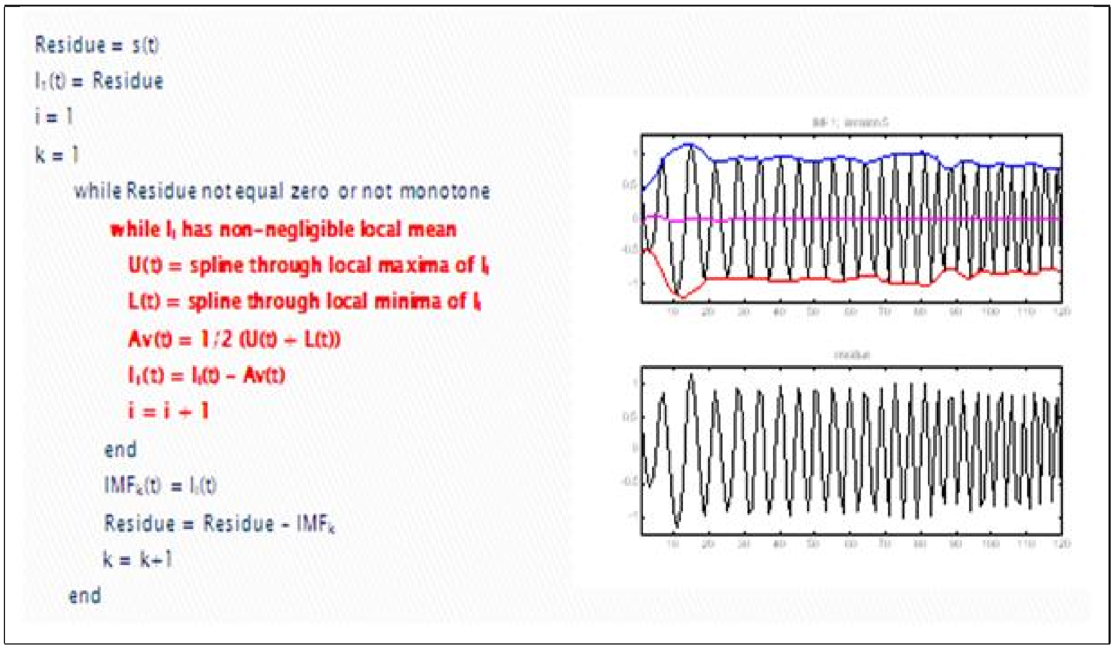


Table 7.24: SP at Iteration 4.



```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t) = I_i(t)$ - Av(t)
            i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

Table 7.25: SP at Iteration 5.



```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
            U(t) = spline through local maxima of $I_i$
            L(t) = spline through local minima of $I_i$
            Av(t) = 1/2 (U(t) + L(t))
            $I_i(t) = I_i(t)$ - Av(t)
            i = i + 1
        end
        $IMF_k(t) = I_i(t)$
        Residue = Residue - $IMF_k$
        k = k+1
    end
```

Table 7.26: SP at Iteration 6.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
          U(t) = spline through local maxima of $I_i$
          L(t) = spline through local minima of $I_i$
          Av(t) = 1/2 (U(t) + L(t))
          $I_i(t)$ = $I_i(t)$ - Av(t)
          i = i + 1
       end
       $IMF_k(t)$ = $I_i(t)$
       Residue = Residue - $IMF_k$
       k = k+1
    end
```

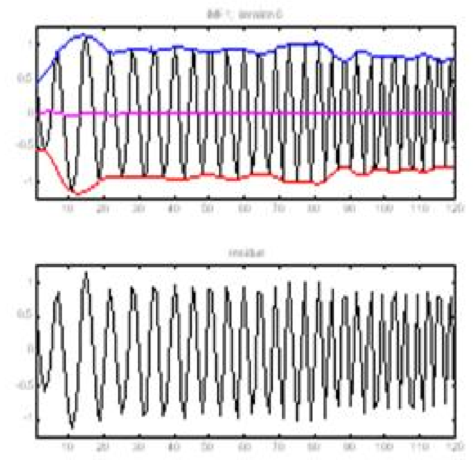


Table 7.27: SP at Iteration 7.

```
Residue = s(t)
$I_1(t)$ = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while $I_i$ has non-negligible local mean
          U(t) = spline through local maxima of $I_i$
          L(t) = spline through local minima of $I_i$
          Av(t) = 1/2 (U(t) + L(t))
          $I_i(t)$ = $I_i(t)$ - Av(t)
          i = i + 1
       end
       $IMF_k(t)$ = $I_i(t)$
       Residue = Residue - $IMF_k$
       k = k+1
    end
```

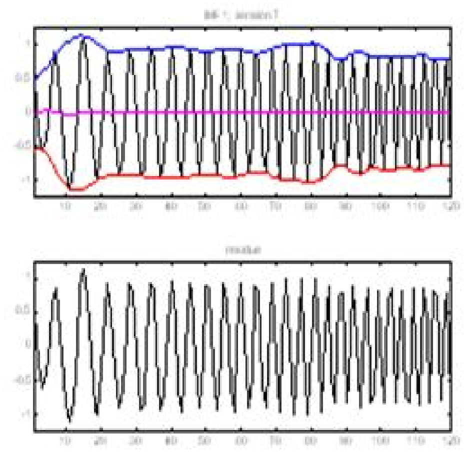


Table 7.28: SP at Iteration 8.

```
Residue = s(t)
I_1(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while I_i has non-negligible local mean
          U(t) = spline through local maxima of I_i
          L(t) = spline through local minima of I_i
          Av(t) = 1/2 (U(t) + L(t))
          I_i(t) = I_i(t) - Av(t)
          i = i + 1
        end
        IMF_k(t) = I_i(t)
        Residue = Residue - IMF_k
        k = k+1
    end
```

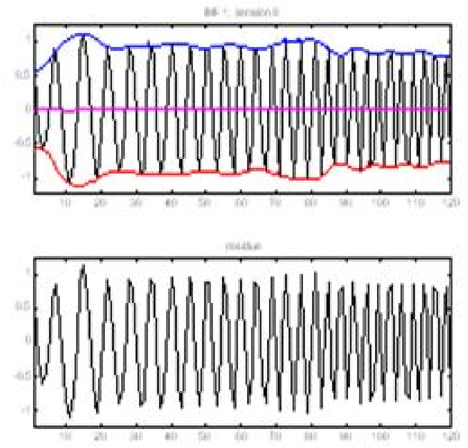
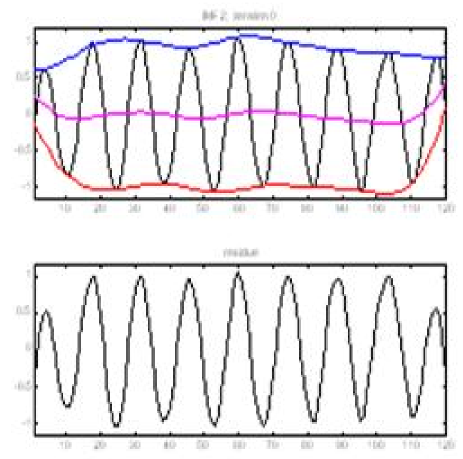


Table 7.29: SP at the End of the Iteration with the Desired Residue, Envelopes and Mean.

```
Residue = s(t)
I_1(t) = Residue
i = 1
k = 1
    while Residue not equal zero or not monotone
        while I_i has non-negligible local mean
          U(t) = spline through local maxima of I_i
          L(t) = spline through local minima of I_i
          Av(t) = 1/2 (U(t) + L(t))
          I_i(t) = I_i(t) - Av(t)
          i = i + 1
        end
        IMF_k(t) = I_i(t)
        Residue = Residue - IMF_k
        k = k+1
    end
```

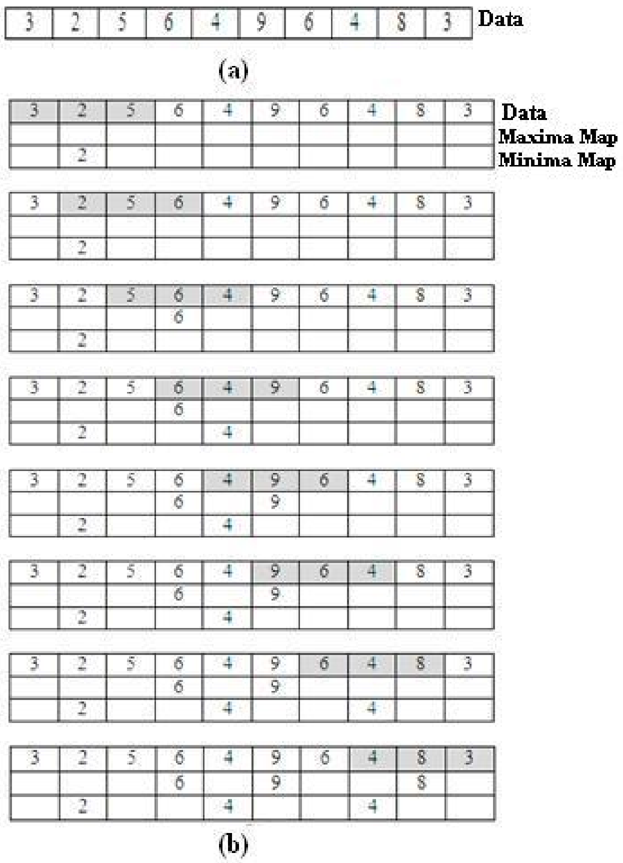


Essentially, the above algorithm is a detailed version of what was introduced previously in Chapters 5 and 6. Next section provides an illustrative example of extrema points detection and interpolation of the Cubic Spline (CS) interpolation method without the proposed boundary adjustment.

7.4 Interpolating and Detecting the Extrema Points

Given the 1 x 10 data vector in Fig. 7.1(a), the step-by-step accumulation of the local maxima and minima points in the corresponding maps for various positions of 1 x 3 window is given in Fig. 7.1(b). As is usually done in most cases for this kind of interpolation, the endpoints of the data in the window were ignored. The process proceeds by comparing the center data with other data points within the window in order to obtain the local maximum or minimum point, if one exists, corresponding to the endpoints of the data. This scenario is shown in Fig. 7.2(a). Note that once the algorithm reaches the end point, it places the window center on the endpoints and compares it with the available data only. To avoid what is traditionally referred to as a "boundary effect," [125, 104] a pseudo-extrema (ignoring the endpoints) is assumed in this example whenever such a situation is encountered at the end of a data vector. The data vector and final maxima and minima maps are given in

Fig. 7.2(b).



3	2	5	6	4	9	6	4	8	3	Data

(a)

3	2	5	6	4	9	6	4	8	3	Data
										Maxima Map
	2									Minima Map

3	2	5	6	4	9	6	4	8	3
	2								

3	2	5	6	4	9	6	4	8	3
			6						
	2								

3	2	5	6	4	9	6	4	8	3
			6						
	2			4					

3	2	5	6	4	9	6	4	8	3
			6		9				
	2			4					

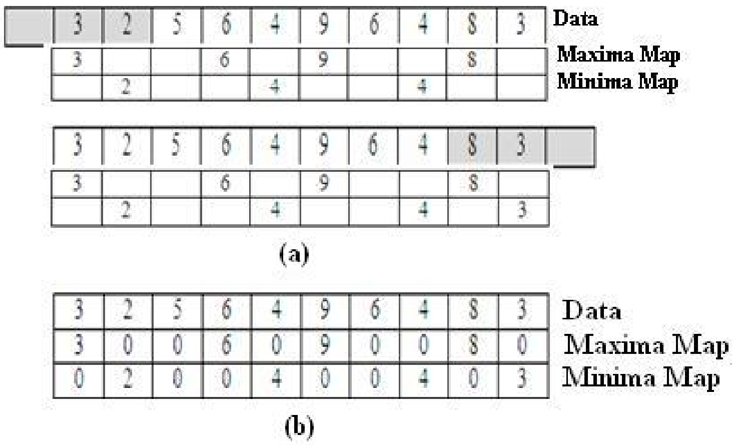
3	2	5	6	4	9	6	4	8	3
			6		9				
	2			4					

3	2	5	6	4	9	6	4	8	3
			6		9				
	2			4			4		

3	2	5	6	4	9	6	4	8	3
			6		9			8	
	2			4			4		

(b)

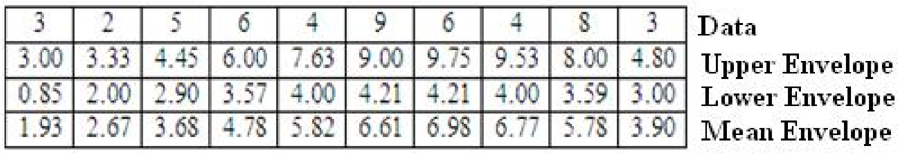
Figure 7.1: A 1 x 10 data vector is shown in (a); Accumulation of the local maxima and minima points for different positions of a 1 x 10 window is shown in (b).



3	2	5	6	4	9	6	4	8	3	Data
3	0	0	6	0	9	0	0	8	0	Maxima Map
0	2	0	0	4	0	0	4	0	3	Minima Map

Figure 7.2: (a) Shows the accumulation of the local maxima and minima points at the ends; and (b) shows the data vector and the final maxima and minima maps.

Application of the cubic spline interpolation algorithm to the maxima and minima maps of Fig. 7.2(b) results in the upper envelope (UE) and lower envelope (LE) data vectors, repectively, given in Fig. 7.3. The mean envelope (ME) data, also given in Fig. 7.3, is obtained by performing point-wise averaging between UE and LE data. Although the mean envelope data is not the same as the local mean/average of the original data, it has some similarity with the true local average, which is generally obtained by averaging the data within a sliding window [103, 128].



3	2	5	6	4	9	6	4	8	3	Data
3.00	3.33	4.45	6.00	7.63	9.00	9.75	9.53	8.00	4.80	Upper Envelope
0.85	2.00	2.90	3.57	4.00	4.21	4.21	4.00	3.59	3.00	Lower Envelope
1.93	2.67	3.68	4.78	5.82	6.61	6.98	6.77	5.78	3.90	Mean Envelope

Figure 7.3: Original data of Fig. 7.1(a), and data vectors corresponding to the UE, LE and ME of the original data.

7.5 Cubic Spline Algorithm

Interpolation is a method of constructing new data points within the range of a discrete set of known data points. It is inspired by the fact that real world numerical data is usually difficult to analyze, and any function which would effectively correlate the data would be difficult to obtain and highly cumbersome. To this end, the idea of the cubic spline was developed. Using this process, a series of unique cubic polynomials can be fitted between each of the data points, with the stipulation that the curve obtained be continuous and appear smooth [104]. These cubic splines can then be used to determine rates of change and cumulative change over an interval.

In addition, cubic splines are a popular choice for curve fitting because of their ease of data interpolation, integration, differentiation, and they are normally very smooth. The basic idea behind cubic spline interpolation is based on the engineer's tool used to draw smooth curves through a number of points. This spline consists of weights attached to a flat surface at the points to be connected. A flexible strip is then bent across each of these weights, resulting in a pleasingly smooth curve. The mathematical spline is similar, in principle, to the cubic spline. The points, in this case, are numerical data. The weights are the coefficients on the cubic polynomials used to interpolate the data. These coefficients "bend" the line so that it passes through each of the data points without any erratic behavior or breaks in continuity [104, 102, 101]. Because of its efficiency with respect to time and performance, coupled with its accuracy, the cubic spline data interpolation algorithm stood out amongst the rest of the radial basis functions (RBFs) family of functions. This explains its utilization in this dissertation.

In mathematical terms, the fundamental idea, as explained above, is to fit a piecewise function of the form

$$S(x) = \begin{cases} s_1(x) & if x_1 \leq x < x_2 \\ s_2(x) & if x_2 \leq x < x_3 \\ & . \\ & . \\ & . \\ s_{n-1}(x) & if x_{n-1} \leq x < x_n \end{cases} \tag{7.1}$$

where s_i is a third degree polynomial defined by

$$s_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \tag{7.2}$$

for i = $1, 2, ..., n-1$.

Note that the first and second derivatives of these $n-1$ equations are key to this process, and they are expressed as

$$s_i^{'}(x) = 3a_i(x - x_i)^2 + 2b_i(x - x_i) + c_i \tag{7.3}$$

$$s_i^{''}(x) = 6a_i(x - x_i) + 2b_i \tag{7.4}$$

for i = $1, 2, ..., n-1$ for both equations.

Furthermore, the cubic splines have to comform to the following properties:

1. The piecewise function $S(x)$ will interpolate all data points.

2. $S(x)$ will be continuous on the interval $[x_1, x_n]$

3. $S^{'}(x)$ will be continuous on the interval $[x_1, x_n]$

4. $S^{''}(x)$ will be continuous on the interval $[x_1, x_n]$

7.6 The BEMDEC Methodology

Having identified the benefits of EMD and its BEMD variances, the proposed methodology is geared toward utilizing the positives aspects of EMD/BEMD while mitigating the negatives. To gain the necessary inherent strengths of the EMD/BEMD, several questions have to be answered properly. These are the domain in which to apply it and the interpolation technique to use. Furthermore, several concerns, such as the stopping criteria and boundary adjustments have to be addressed within the interpolation routine. Having carefully studied the various approaches by several authors, particularly Huang, Nunes and Bhuiyan et al. [2, 89, 145] in dealing with these issues, the methodology proposed herein takes into consideration the aforementioned questions and attempts to address them in a unique fashion or circumbent them in some sense, when necessary. Fig. 7.4 depicts the proposed method called Bidimensional Empirical Mode Decomposition for edges, corners and curves (BEMDEC).

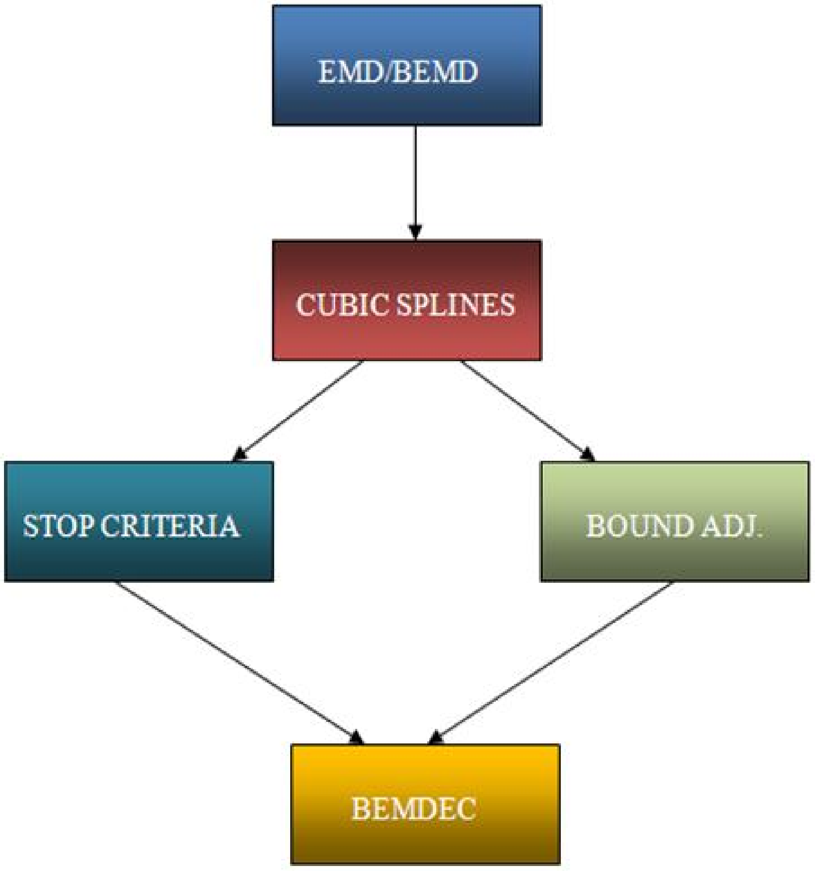


Figure 7.4: Proposed method known as BEMDEC.

From Fig. 7.4, it is easy to see that the EMD/BEMD plays a pivital role in the BEMDEC scheme, for it is the basis from which it was derived. The addition of the cubic spline interpolation algorithm, a flexible stopping criteria, and an adaptive boundary adjustment make the BEMDEC methodology consistent and versatile with respect to detecting edges, corners and curves. It should be noted however, though the detecton of the corners and curves are supplementary operations, their addition makes BEMDEC a complete feature detector.

7.6.1 Putting It All Together

The combination of a non-stationary signal/data handling technique in EMD/BEMD and the cubic spline algorithm interpolation algorithm forms the foundation of the BEMDEC methodology. The strength of BEMDEC, as mentioned earlier, comes from the proposed interpolation algorithm, stopping criteria, and boundary adjustments. Having described the Cubic Splines interpolation algorithm earlier, it will only be revisited briefly below while detailed summaries of the proposed stopping criteria and boundary adjustment are given.

7.6.2 The Cubic Splines (CS) Algorithm Revisited

As mentioned earlier, the potential and effectiveness of the BEMD methodologies depend on adequately answering two key questions in order to achieve an acceptable operational cost, quality, and robustness. That is what domain will it be apply to? And what interpolation method will be used? In addition, our interpolation method of choice will be affected further by our handling of the stopping criteria and the boundary effect; two conditions typically encountered during the interpolation routine of the sifting process. The stopping criteria and the boundary adjustment will be explained in details in the subsequent sections.

Furthermore, the designation of the cubic spline (CS) as the interpolation method of choice in this work is not arbitrary. As explained earlier, it has the desirable qualities needed to properly interpolate scattered data, and it generally demonstrates consistent and stable behaviors while doing so. Of all the available interpolation techniques, CS incorporates the ease of interpolation, integration, and differentiation with efficiency and flexibility. To demonstrate the effectiveness of the CS interpolation, Fig. 7.5 and Fig. 7.6 show the effective interpolation of the CS algorithm against polynomial interpolation, one of its closest rivals. As can be seen, the data interpolation of the CS algorithm is much more effective. The next two sections briefly revisited the Huang and FABEMD [2, 89] approaches to addressing the aforementioned questions, after which, the proposed stopping criteria and the boundary adjustment of this dissertation will be presented.

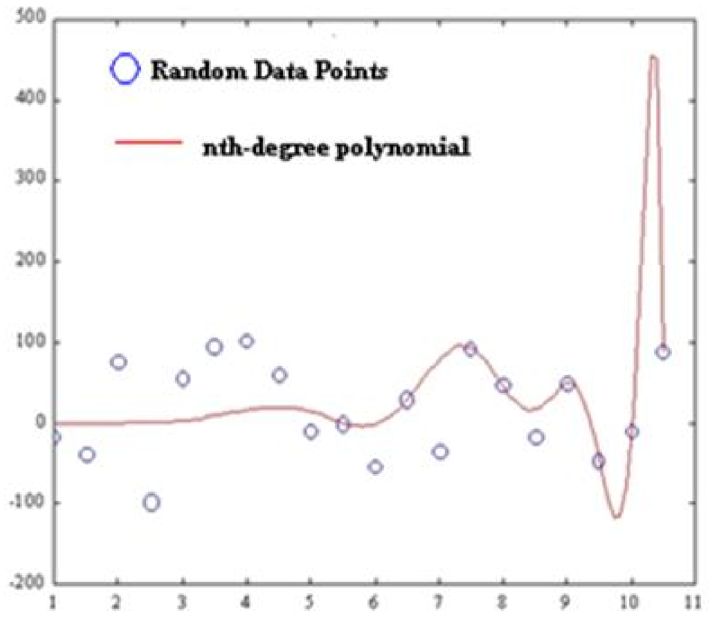


Figure 7.5: Example of polynomial interpolation with scattered random data points.

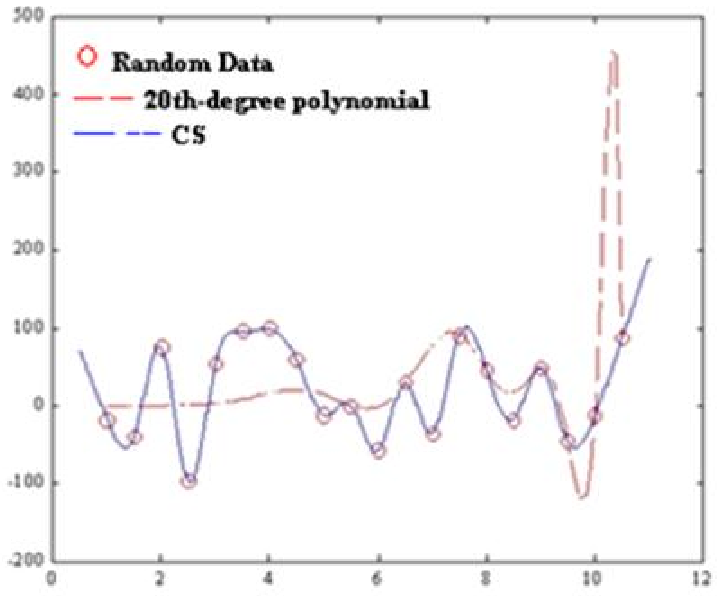


Figure 7.6: Example of polynomial interpolation vs. Cubic Spline (CS) with the same scattered random data points.

7.6.3 Huang's Approach Revisited

As mentioned previously, the Empirical mode decomposition (EMD), developed by Norden E. Huang et al. [89] in 1998 is a method of breaking down a signal without leaving a time domain and, in some sense, serves the same purpose as the Fourier Transforms and wavelet decomposition image analysis techniques. It became popular in the signal processing and time analysis domains. This popularity stemmed from the fact that the basic functions used to decompose a signal are not predefined but adaptively derived from the signal itself [108, 109, 110, 99, 100].

Nevertheless, caution has to be taken when any attempt is made to switch domains of application. This is due, in large part, to the fact that important questions, application domain being one of these, have to be answered and adequately addressed. In addition to the domain question, interpolation algorithm, along with the associated stopping criteria and boundary effect has to be addressed. Huang et al. used cubic splines fitting and proposed interesting stopping criteria and boundary adjustment [89].

In his work, Huang considered the sifting process to be crucial to the effectiveness of the EMD. In fact, it is the sifting algorithm that determines the quality of the output. However, since his goal was to allow it to separate the finest local mode from the data first based only on the time scale, he defined two important desired effects: First is the elimination of the riding waves and the second is the smoothing of uneven amplitude. The two conditions are necessary for a meaningful instantaneous frequency and a large disparity in the neighboring wave amplitudes, respectively. To guarantee that the IMF components retain enough physical sense of both amplitude and frequency modulation, Huang proposed a stopping criterion using a standard deviation. In essence, he limited the size of standard deviation, SD, computed from the two consecutive sifting results as

$$SD = \sum_{t=0}^{T} \left[\frac{|(h_{1(k-1)}(t) - h_{1k}(t))|^2}{h^2_{1(k-1)}(t)} \right], \tag{7.5}$$

where the two optimal values for SD can be set between 0.2 and 0.3.

Another issue that has to be dealt with is the boundary effect. During the interpolation procedure, data interpolation gets complicated when it gets to the end of the given window. This is because the values at the end do not have enough neighbors that they can be compared to, and as such create a situation where, if not handled properly, can negatively affect the interpolation output. Huang proposed and defined two extremas amplitude at the end of the window to deal with this effect, and this seems to work well with his domain of choice, which is time domain [89].

However, some concerns still exist. For instance, the overshoot and undershoot problems seem to have been exacerbated by the choice of the stopping criteria with cubic spline fitting, and the author contemplated using a different, though expensive, method (higher order spline). However, that method, upon initial test, only showed marginal improvement. The case of weak signals imbedded in stronger ones is another cause of concerns because this situation causes the extrema to be invisible in the final signal notwithstanding being picked up by the sifting process. In addition, there are cases where the individual components of the IMF do not guarantee a well defined physical meaning for all decomposition, especially in methods with a priori basis. Nonetheless, the Huang's EMD remains a power tool in regard to handling nonlinear, non-periodic and non-stationary data in the time domain [89].

Huang's methodology shown some promise, but because of its domain of application, his stopping criteria and the boundary adjustment only work well with time domain. As such, they were not flexible with a cross domain application. This warrant proposal of unique stopping criteria and boundary adjustment in this dissertation.

7.6.4 The FABEMD Approach Revisited

Upon realizing the benefit of Huang's proposed methodology, several authors, especially Adhami, Nunes and Bhuiyan [101, 145, 2] investigated its cross domain application and what they found was not disappointing.

Nunes et al. [145] tested it with image texture extraction and Bhuiyan et al. [101] followed suit by proposing an order statistic filters (OSFs) based approach to analyze images. Because of it potential to be fast, they called this methodology Fast and Adaptive Bidimensional Empirical Mode Decomposition (FABEMD) [101].

In addition, the basis of FABEMD's algorithm is the proposal of a new envelope estimation method. Otherwise, most of the algorithmic steps remain similar to that of the classical EMD/BEMD described earlier and will not be reproduced here. Also, this methodology adopts single iteration for reach BIMF, which improves its computational time efficiency. One of the keys to the FABEMD algorithm is the determination of the order statistic filter width, which then allows the application of maximum and minimum filters to the source image in order to obtain the upper and lower envelopes respectively [101]. These envelopes are then smoothed by smoothing averaging filters.

As mentioned above, the essence of FABEMD is its application of a statistical methodology known as the order statistic filter. His interpolation method proceeds as followed:

For each local maximum point in $P_i^S(x,y)$ (i.e., for each non-zero element in $P_i^S(x,y)$), the Euclidean distance to the nearest other local maximum point (non-zero element) is calculated and stored in an array, called neighboring distance vector of the maxima, and it is denoted as λ_i^U. Similarly, for each local minimum point in $Q_i^S(x,y)$ (i.e., for each non-zero element in $Q_i^S(x,y)$), the Euclidean distance to the nearest other local minimum point (non-zero element) is calculated and stored in an array, called neighboring distance vector of the minima, and it is denoted as λ_i^L. The number of elements in λ_i^U or λ_i^L is equal to the number of maxima or minimum points in $P_i^S(x,y)$ or $Q_i^S(x,y)$, respectively. The distance values in λ_i^U and λ_i^L maybe arranged in ascending or descending order for easy selection of the quantity later. And instead of taking the distance to the nearest neighbor, other distances, which may have not yet been explored or tested properly, may also be utilized. But according to Bhuiyan, only the use of distance to the nearest neighbor for each local maximum or minimum points yield the desired separation of the characteristic local spatial scales or variation of an image into its BEMFs [101].

Though the FABEMD approach share the same domain of application with BEMDEC proposed in this dissertation, their envelop estimation methods are different and the two puruse different stopping criteria and boundary adjustments. While FABEMD proposed an Extrema Threshold (ET) and Number of Extrema Points (NEP) for his stopping criteria, some inherent issues, such as undefined duration and a limiting range of the two parameters were a cause of concern. Furthermore, the use of false extrema for boundary adjustment raised some concerns.

7.6.5 Stopping Criteria

It has been emphasized that sifting process (SP) is a vital part of the BEMD method regardless of the application domain. Nevertheless, it can be overdone and, in some cases, not done enough. This typically results in two conditions known as overshoot and undershoot, which can adversely affect the desired outcome. As such, carefully establishing when to stop or continue is of utmost importance. This is controlled by the stopping criteria.

In this dissertation, the proposed stopping criteria is simple and straight forward. Given the sifting algorithm, three parameters are defined. These are a global threshold, denoted by θ_1, an evaluation function denoted by $\delta(t)$ and the iteration duration denoted by $1-\delta$. Both the global threshold and the evaluation function start out at 0.05, a value that is observed to be optimal. The sifting process then iterates until evaluation function is less than the global threshold at which point we have reached our stopping criteria.

Specifically, the following key parameters and functions/routines, each with an associated syntax, imbedded in the sifting process, control and guide the proposed stopping criteria in the BEMDEC implementation:

- *BEMDEC*: This parameter computes the bidimensional empirical mode decomposition.
- *BIMF*: This parameter describes one bidimensional imperial mode function and has the following syntax: BIMF = BEMDEC (X) where X is a real vector computing the Bidimensional Empirical Mode Decomposition of X, which results in a matrix BIMF containing 1 BIMF per row, the last one being the residue. The sifting process stops when the evaluation function is less than the global threshold.
- *SP_STOP*: vector of a customized stopping parameters (Evaluation_Function, Global_Threshold, Tolerance). The length of the default stopping vector is 3 and if it is found to be less, then it is set to defaults of (0.05, 0.05, 0.6).
- *FIXED*: Is an integer parameter value which, if invoked, disables the given stopping criteria in order to do exactly fixed number of iterations for each BIMF.
- *Num_Directions*: Is an image gradient or direction in which lower and upper envelopes are computed. The default direction is 3 (vertical, horizontal, diagonal).
- *TIME*: Is sampling time.
- *MAX_ITERATIONS*: Maximum number of iterations in the sifting process for the computation of each BIMF.
- *MAX_BIMF*: Maximum number of BIMFs extracted during each iteration.
- *DISPLAY*: Is a Boolean variable giving the option to display the steps in the sifting process. This option is disabled when the size of the image is too large.

- *Inter_Method*: Is the interpolation method. This parameter allows a choice to use a different interpolation method beside the default (Cubic Spline).
- *Stop_BEMDEC*: BEMDEC function to process extrema points, where the default is three extrema points.
- *Sift_Stop*: Function controlling the stopping criteria
- *Bound_Condition*: Function controlling the proposed boundary condition
- *Envelopes_Cal*: Function responsible for the calculation of the various envelopes (i.e., mean, upper and lower envelopes).
- *Display_BEMDEC*: Function to display the progression of the BEMDEC decomposition with the default criteria.

Having adopted some of the notational conventions of [89, 145, 101], essentially, the original image $I(x,y)$ is decomposed first into a set of BIMFs, $F_i(x,y)$, and a residue $R(x,y)$, where $i = 1,2,...k$, and k is the total number of BIMFs, using BEMDEC. In the subsequent steps, the $F_i(x,y)$ binarized to $B_i(x,y)$ with a suitable threshold $F_i(x,y)$, where t_i is computed as

$$t_i = max\{F_i(x,y)\} \quad x \quad \gamma_i \tag{7.6}$$

where $maxF_i(x,y)$ denotes the maximum value of the pixels in the ith BIMF, $F_i(x,y)$; and $0 \leq \gamma_i \leq 1$. Note that the value of γ_i is chosen as needed. Furthermore, the binarization operation can be computed as followed:

$$B_i(x,y) = \left(\begin{array}{l} 1, \quad if \quad F_i(x,y) > t_i \\ \quad 0, \quad otherwise \end{array} \right. \tag{7.7}$$

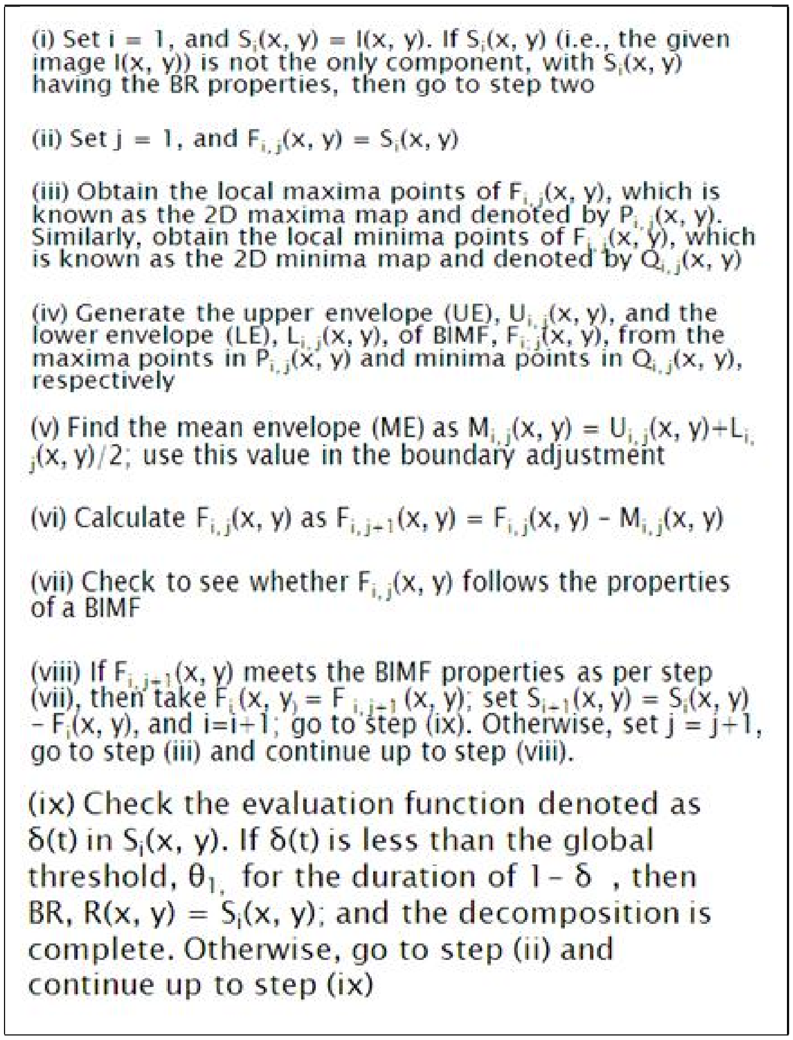
where (x,y) represents the coordinate of the corresponding BIMF or the binary image. It should be noted that the non-zero pixels in the binary image are the candidate edge pixels.

Upon obtaining the binary image $B_i(x,y)$, thinning operation is applied to get the single pixel width skeleton image $T_i(x,y)$. The purpose of thinning and skeletonization operation (these are summarized later) is to removes pixels on the boundaries of objects while keeping it in intact [101]. The remaining pixels make up the image skeleton. This process is repeated until no changes occur in the image. Because the above process may identify some unnecessary edges, it is important that some of these disjointed, unconnected or undesired edges are removed from the skeleton image $T_i(x,y)$ by further processing. The result is the final cleaned edge image $T_i(x,y)$. This removal operation of undesired edges can be controlled by a threshold parameter ζ_i, which is the desired maximum length in pixels of the unconnected edges to be removed. Any edge segment having length equal to ζ_i or lower is removed from the skeleton image $T_i(x,y)$. For convenient and to show how the sifting process works in BEMDEC, two tables, the first containing the parameters (and their meanings) used in the algorithm and the second containing the complete BEMDEC algorithm, are provided below. Fig. 7.7 show a pictorial view of the final and modified BEMDEC diagram with the supplementary corners and curves depicted.

Table 7.30: Parameters Used in the BEMDEC Algorithm (x and y are image gradients while i and j are image rows and columns).

Parameter	Description
$I(x,y)$	The image
$R(x,y)$	Bidimensional Residue (BR)
$F_{i,j}(x,y)$	Intermediate state of BIMF
$P_{i,j}(x,y)$	2D maxima map
$Q_{i,j}(x,y)$	2D minima map
$U_{i,j}(x,y)$	Upper envelope
$L_{i,j}(x,y)$	Lower envelope
$M_{i,j}(x,y)$	Mean envelope
$\delta(t)$	Evaluation function
θ_1	Global threshold

Table 7.31: Complete BEMDEC Algorithm.



(i) Set $i = 1$, and $S_i(x, y) = I(x, y)$. If $S_i(x, y)$ (i.e., the given image $I(x, y)$) is not the only component, with $S_i(x, y)$ having the BR properties, then go to step two

(ii) Set $j = 1$, and $F_{i,j}(x, y) = S_i(x, y)$

(iii) Obtain the local maxima points of $F_{i,j}(x, y)$, which is known as the 2D maxima map and denoted by $P_{i,j}(x, y)$. Similarly, obtain the local minima points of $F_{i,j}(x, y)$, which is known as the 2D minima map and denoted by $Q_{i,j}(x, y)$

(iv) Generate the upper envelope (UE), $U_{i,j}(x, y)$, and the lower envelope (LE), $L_{i,j}(x, y)$, of BIMF, $F_{i,j}(x, y)$, from the maxima points in $P_{i,j}(x, y)$ and minima points in $Q_{i,j}(x, y)$, respectively

(v) Find the mean envelope (ME) as $M_{i,j}(x, y) = U_{i,j}(x, y)+L_{i,j}(x, y)/2$; use this value in the boundary adjustment

(vi) Calculate $F_{i,j}(x, y)$ as $F_{i,j+1}(x, y) = F_{i,j}(x, y) - M_{i,j}(x, y)$

(vii) Check to see whether $F_{i,j}(x, y)$ follows the properties of a BIMF

(viii) If $F_{i,j+1}(x, y)$ meets the BIMF properties as per step (vii), then take $F_i(x, y) = F_{i,j+1}(x, y)$; set $S_{i+1}(x, y) = S_i(x, y) - F_i(x, y)$, and $i=i+1$; go to step (ix). Otherwise, set $j = j+1$, go to step (iii) and continue up to step (viii).

(ix) Check the evaluation function denoted as $\delta(t)$ in $S_i(x, y)$. If $\delta(t)$ is less than the global threshold, θ_1, for the duration of $1 - \delta$, then BR, $R(x, y) = S_i(x, y)$; and the decomposition is complete. Otherwise, go to step (ii) and continue up to step (ix)

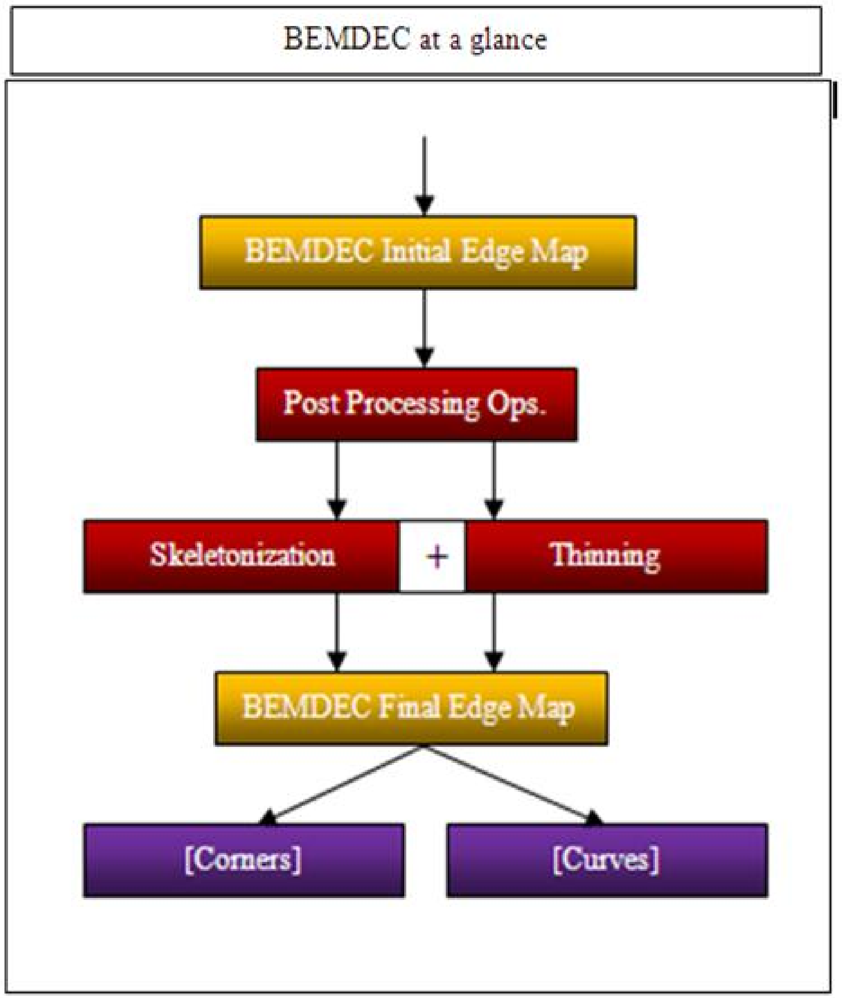


Figure 7.7: Modified BEMDEC algorithm with corners and curves.

7.6.6 Skeletonization

The post processing steps of skeletonization and thinning in Fig. 7.7 are typically a direct result of morphological operators. Morphology is a broad set of image processing operations that process images based on shapes. As described in the Matlab manual and demo, morphological operations apply a structuring element to an input image, creating an output image of the same size. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors. By choosing the size and shape of the neighborhood, you can construct a morphological operation that is sensitive to specific shapes in the input image.

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image. In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as dilation or erosion.

The skeletonization operation - as the name implies, therefore, reduces all objects in an image to lines, while leaving the essential structure of the original image. Because of its ability to provide region-based shape feature, it has become a common preprocessing step in raster-to-vector conversion and many pattern recognition applications. Three skeletonization techniques are detecting ridges in distance map of the boundary points, calculating the Voronoi diagram generated by the boundary, and layer by layer erosion. Fig. 7.8 demonstrates the idea of skeletonization showing the before and after image.

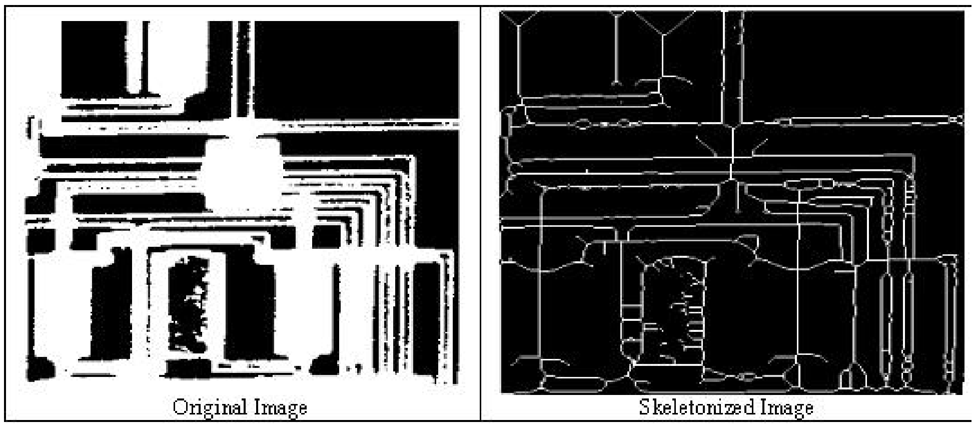


Figure 7.8: Example of skeletonization.

7.6.7 Thinning

Like skeletonization, thinning is a morphological operation that is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. It can be used for several applications, but it is particularly useful for skeletonization. In this mode, it is commonly used to organize the output of edge detectors by reducing all lines to single pixel thickness. Thinning is normally only applied to binary images, and produces another binary image as output as is the case in this dissertation. The thinning operation is related to the hit-and-miss transform, which is described below. Like other morphological operators, the performance of the thinning operation is determined by a structuring element.

As described by Fisher et al. the binary structuring elements used for thinning are of the extended type described under the hit-and-miss transform, meaning they can contain both ones and zeros. The thinning operation is related to the hit-and-miss transform and can be expressed quite simply in terms of it. The thinning of an image I by a structuring element J is:

$$Thin(I,J) = I - hit - and - miss(I,J) \tag{7.8}$$

where the subtraction is a logical subtraction defined by $X - Y = X \bigcap NOTY$

In another word, the thinning operation is calculated by translating the origin of the structuring element to each possible pixel position in the image, and at each such position comparing it with the underlying image pixels. If the foreground and background pixels in the structuring element exactly match foreground and background pixels in the image, then the image pixel underneath the origin of the structuring element is set to background (zero). Otherwise it is left unchanged. Note that the structuring element must always have a one or a blank at its origin if it is to have any effect. The choice of structuring element determines under what situations a foreground pixel will be set to background, and hence it determines the application for the thinning operation.

This is the effects of a single pass of a thinning operation over the image. In fact, the operator is normally applied repeatedly until it causes no further changes to the image (i.e. until convergence). On the other hand, in some applications such as pruning, the operations may only be applied for a limited number of iterations. It should be further note that thinning is the dual of thickening, i.e. thickening the foreground is equivalent to thinning the background.

One of the most common uses of thinning is to reduce the thresholded output of a given edge detector, such as BEMDEC, to lines of a single pixel thickness, while preserving the full length of those lines (i.e. pixels at the extreme ends of lines should not be affected). This is accomplished as described below:

- Consider all pixels on the boundaries of foreground regions, thus foreground points that have at least one background neighbor.
- Delete any such point that has more than one foreground neighbor, as long as doing so does not locally disconnect (i.e. split into two) the region containing that pixel.
- Iterate until convergence. This procedure erodes away the boundaries of foreground objects as much as possible, but does not affect pixels at the ends of lines.

The effect of this procedure is illustrated by Fig. 7.9.

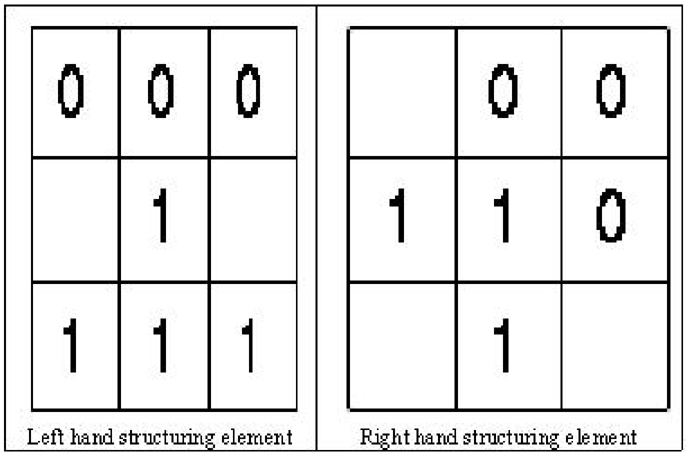


Figure 7.9: Thinning the given image by left and right hands structuring elements.

7.6.8 Boundary Adjustment

Boundary adjustment is another vital consideration in the sifting process. Fig. 7.10 demonstrates the boundary effect in 1x3 sliding window. Note from figure 7.1 (b) that the interpolation method in the sifting process considers a 1x3 sliding window at a time comparing the middle value with its neighbors in determining the maxima and minima maps. During this process, the middle value is compared with its neighbors, and if it is greater than both of the neighbors, it is placed in the maxima map. If the middle value is less than both of the neighbors, it is placed in minima map. Otherwise, it is ignored. Looking at Fig. 7.10, it is easy to see that this is the case when 1 is being considered. The problem occurs when the algorithm gets to the boundary values (2 and 3 in this case). Clearly, these two values have one neighbor each and comparing either 2 or 3 with its current neighbor only tells half of the story. This situation is known as the boundary effect, and it has been shown that it has a negative effect on the final output, be it image or a signal [89]. Properly handling this boundary effect is therefore paramount.

The proposed boundary adjustment is even simpler. Having already computed the values of the lower and upper envelopes in the interpolation routines, instead of ignoring the values at the boundaries, the average of this two values is taken and is used as the boundary value. This designation has been termed as "extended averaging"(EA) in this work. The results of this scheme seem to be very favorable compare to those obtained when the boundaries values are ignored or false ones are used.

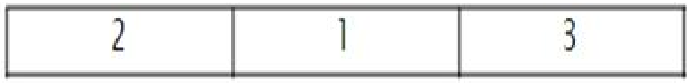


Figure 7.10: 1 x 3 window demonstrating the boundary effect.

Futhermore, the extension of BEMDEC scheme to corner and curves and the associated algorithm are presented in the next section. It is worth mentioning that the addition of the corner/curve detection is supplementary in this dissertation, and, as such, indepth coverage of these two feature has not been conducted. It suffices to say, however, that two popular corner detection operators, namely HARRIS and SUSAN, and a curve algorithm, known as Generalized Radon Transform (GRT), are briefly summarized below for comparison purposes. Nonetheless, the proposed algorithm produced some very positive results. Details of the algorithms are provided following the brief discussion of HARRIS, SUSAN and the

Generalized Radon Transform.

7.6.9 HARRIS Operator Revisited

As mentioned earlier, the HARRIS [94] corner detector, sometimes known as PLESSEY, was developed in 1988 by Harris and Stephens. This detector, as is typically the case, was developed with the intention to address the limitations of some of the earliest detection operators such as Moravec operator. This resulted in a more desirable detection operator in terms of good detection and repeatability rate. Nevertheless, this was at the cost of requiring significantly more computation time. Due to the lack of alternative however, the HARRIS operator was widely used during its time [90, 94].

The basic idea behind HARRIS' operator is the matching of corresponding points in consecutive image frames while tracking corners between frames. During this process, the operator tries to find points with large corner response function R, where R is greater than some given threshold. Then the points of local maxima of R are taken to define the corner [94]. This produces the change of intensity for the shift [u, v]

$$E(u,v) = \sum_{x,y} w(x,y)\,[I(x+u,y+v) - I(x,y)]^2 \tag{7.9}$$

where $w(x,y)$ is the window function, $I(x+u,y+v)$ is shift intensity and $I(x,y)$ is the intensity.

Due to the inherent shift and window function calculations of the HARRIS operator, it is known to be computationally intensive and, as such, has become less attractive particularly when the size of the image is large. Furthermore, its heavy production of false positives, a situation that occurs when an operator classifies non-existence feature as true feature, makes it less useful.

7.6.10 SUSAN Operator Revisited

In 1995, Smith and Brady [90] proposed a new approach for low level image processing, especially for corner and structure preserving noise reduction. Because it is based on brightness comparison within a circular mask, they called it SUSAN (Smallest Univalue Segment Assimilating Nucleus) because it assumes that within a relatively small circular region, pixels belonging to a given object will have relatively uniform brightness. The pixels computation is achieved by calculating the number of pixels with similar brightness together with those at the center of the mask, which is the nucleus of the mask. These pixels are called the USAN (Univalue Segment Assimilating Nucleus) of the mask. Corners are then detected by applying the mask to all pixels in the image and then finding the local minima in this new USAN map [90].

The algorithm proceeds as followed:

1. Determine a circular mask, typically of 37 pixels around a nucleus for each point within the image

2. Calculate the difference in brightness between each pixel of the mask and that of its nucleus and
$$C(\overrightarrow{r},\overrightarrow{r}_0) = \begin{cases} 1, & if \quad |I(\overrightarrow{r}) - I(\overrightarrow{r}_0)| \leq t \\ 0, & otherwise \end{cases} \tag{7.10}$$

3. Sum the number of pixels within the circular mask which have similar intensity levels to that of the nucleus
$$n(\overrightarrow{r}_0) = \sum_{\overrightarrow{r}_0} C(\overrightarrow{r},\overrightarrow{r}_0) \tag{7.11}$$
where r is a given pixel location within the mask and r_0 is the position of the mask nucleus in the image frame. $I(r)$ refers to the intensity of the pixel at location r, t is the brightness threshold, and c is the output of the comparison in intensities at locations r and r_0. The sum of the comparison outputs c is then taken and that represents the total number of pixels in the USAN region, in other words, the USAN area.

4. Compare n with g, the geometric threshold which is set to half of the maximum value that n can be $(n_{max}/2)$.

5. At a perfect corner (where two straight edges intersect) the USAN area will always

be less than half the size of the mask area, and will be a local minimum

$$R(\overrightarrow{r}_0) = \begin{cases} g - n(\overrightarrow{r}_0), & if \quad n(\overrightarrow{r}_0) < g \\ 0 \quad otherwise \end{cases} \tag{7.12}$$

The above rule is a simple formulation of the SUSAN principle since the response is inversely proportional to the USAN area.

From the algorithm, it is easy to see that, unlike most feature extraction operators; it is not based on the derivative computation and, for this reason, does not explicitly employ the local maximum/minimum concepts. This results in a poor repeatability rate and a high probability of false positives. Like the HARRIS operator, SUSAN is also computationally inefficient.

7.6.11 Generalized Radon Transform (GRT) Revisited

Curve detection, just like that of edges and corners, also play a role, though limited, in the general area of feature detection. And although there isn't as much literature on the subject as there is on edges and corners, it is still an important topic to examine. Of the few curve detection algorithms out there, one has shown greater potential warranting discussion, though brief, in this dissertation. The Radon Transform in general and the Generalized Radon Transform (GRT) in particular have been useful.

The Generalized Radon Transform (GRT) introduced by Rag in [96] is able to extract lines out of an image. In its basic form, a line can be represented by two parameters, such as the slope and the slope-intercept, and by varying these two parameters, it can obtain various lines. As the result, the Radon Transform of a continuous two-dimensional function $g(x,y)$ is found by stacking or integrating values $g(x,y)$ along slanted lines. In this process, the location of the line is determined from the line parameters (i.e., slope p and τ).

Similarly, GRT algorithm detects salient curves from the image by locating inner product maxima in the parameter space. In the case of first order Radon Transform, the inner product of each line with the image is calculated to give rise to a two-dimensional parameter representation in the (p,τ) rather than (x,y) domain of the image. Peaks in the parameter domain correspond to the salient lines present in the image [95].

The Generalized Radon Transform (GRT) can also be use effectively to extract faint curves from a noisy image. In given situations, GRT can handle higher order polynomial and other general functions as well as curve tracing. The GRT curve extraction described above, where curves are extracted from the image by locating inner product maxima in the parameter space, is given by

$$\hat{g}(p,\tau) = \int_{-\infty}^{\infty} g(x, px+\tau)dx \tag{7.13}$$

One major issue of concern for the Generalized Radon Transform (GRT) methodology is its operation outside of the x/y domain. This can be seen by observing its two-dimensional parameter representation in the (p,τ) rather than (x,y) domain of the image.

7.6.12 Supplementing BEMDEC with an Adaptive Corner and Curve Scheme

Corner and curve detections have a lot of similarities with edge detection except with few specifics regarding mathematical notations and definitions. In that sense, the three operations share quite a number of preprocessing steps. Like edges detection, one of the major drawbacks with the current corner/curve schemes is their inability to effectively detect true corners/curves and the tendency to detect noise as a true feature [93]. This oftentimes happens because of an over or under use of the detection scale.

$$\kappa(\mu,\sigma) = \frac{\hat{X}(\mu,\sigma)\acute{Y}(\mu,\sigma) - \acute{X}(\mu,\sigma)\hat{Y}(\mu,\sigma)}{\hat{(}X(\mu,\sigma)^2 - \hat{Y}(\mu,\sigma)^2)^{1.5}}, \tag{7.14}$$

where $\hat{X}(\mu,\sigma) = x(\mu) \otimes \hat{g}(\mu,\sigma), \acute{X}(\mu,\sigma) = x(\mu) \otimes \acute{g}(\mu,\sigma), \hat{(}Y(\mu,\sigma) = y(\mu) \otimes \hat{g}(\mu,\sigma), \acute{Y}(\mu,\sigma) = y(\mu) \otimes \acute{g}(\mu,\sigma)$ and $\otimes$ is the convolution operator while $g(\mu,\sigma)$ denotes a Gaussian function with deviation σ and $\hat{g}(\mu,\sigma)$, $\acute{g}(\mu,\sigma)$ are the first and second derivatives of $g(\mu,\sigma)$ respectively. Using this curve definition, we develop an algorithm utilizing specific scale and threshold.

The algorithm proposed herein is different from the existing schemes in that it employs a medium scale along with global and local maxima of the surface to strike a high level of balance so as to mitigate the previously mentioned concerns. The algorithm proceeds as followed:

- Extract edge contours from the edge map produced by *BEMDEC*.
- After contour extraction, compute the *curvature* at a *fixed medium* scale for each contour to retain the true corners/curves, and regard the local maxima of the surface as corner/curve candidates.
- Compute a *global detection threshold* based on the mean curvature within a region of support.
- Based on a *strongly recalculated region of support*, evaluate the angles of the remaining corner/curve candidates to eliminate any false corners/curves.
- Finally, consider the end points of open contours, and mark them as corners /curve unless they are very close to another corner.

Some of the key functions and parameters in the algorithm are summarized below:

- *Test_Image*: Is a parameter representing the test image whose features are being detected.
- *Print_Position*: Is a parameter which lists the positions of the detected features.
- *BI*: Is a parameter representing a binary image.
- *C_Gaps*: Is an input variable to fill the gaps of the contour.
- *Varargin*: Is a Matlab optional parameter.
- *Max_Angle*: Is a parameter denoting the maximum angle of a corner.
- *S_Deviation*: Is a parameter denoting standard deviation of a curve computation filter.
- *Ratio*: Is a parameter denoting the ration of a contour axis.
- *C_Flag*: Is a parameter representing control flag for adding curve endpoints.

7.7 Conclusion

This chapter gives the background of the proposed method including the various algorithms used to form the BEMDEC methodology. It further presented illustrative examples of the main algorithm to give the reader the necessary background to follow the discussion without confusion. Finally it pictorially depicted the scheme and how various parts work together or complement each other.

Chapter 8

EXPERIMENTAL RESULTS AND FUTURE WORK

8.1 Preliminary Results

This chapter gives some experimental results of the proposed BEMDEC method. The first image is 337 x 253 The University of Southern Mississippi Recreational (Payne) Center to test the BEMDEC edge detection capabilities. Fig. 8.1 shows the original image and the resulting edge maps.

The preliminary results were meant to test the effectiveness of the BEMDEC methodology, first using familiar images (USM stadium and the Payne Center). The idea was to see the initial performance using those images before it can be applied to some of the standard images. After that, several standard images (the House and Coins references) were used to test BEMDEC's ability to detect edges, corners and curves in reasonable time as well its ability to produce quality edge maps. The screenshots were shown.

The subsequent figures, therefore, show BEMDEC detection time in seconds as well as its comparison to other two closely related edge detection methods namely Canny and FABEMD. To demonstrate edge production quality, the Coin reference was used and a side-by-side comparison of the three edge detections methods was also given. Using the House reference, BEMDEC was also compared to two corner detection methods namely SUSAN and HARRIS in terms of time and edge map quality.

Fig. 8.2 shows a 620 x 219 University of Southern Mississippi Footbal Stadium to test the BEMDEC corner detection capabilities. Fig. 8.2 show the original image and the resulting edge maps.

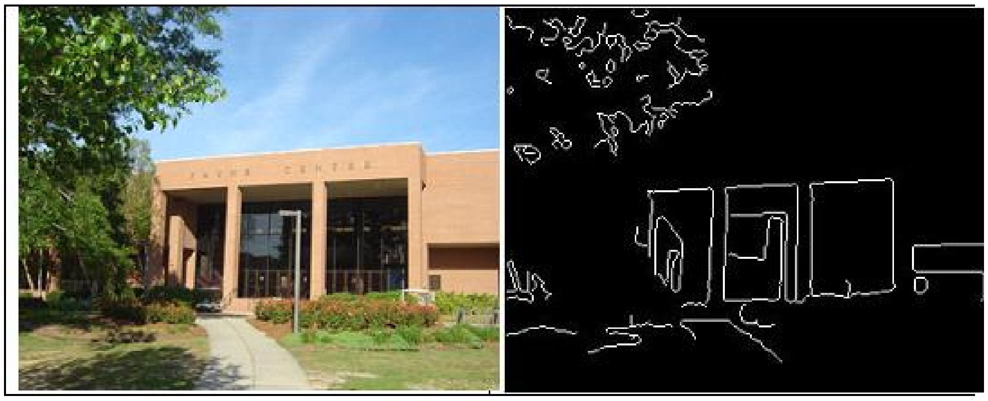


Figure 8.1: A 337 x 253 University of Southern Mississippi Recreational (Payne) Center original image and the resulting edge map.

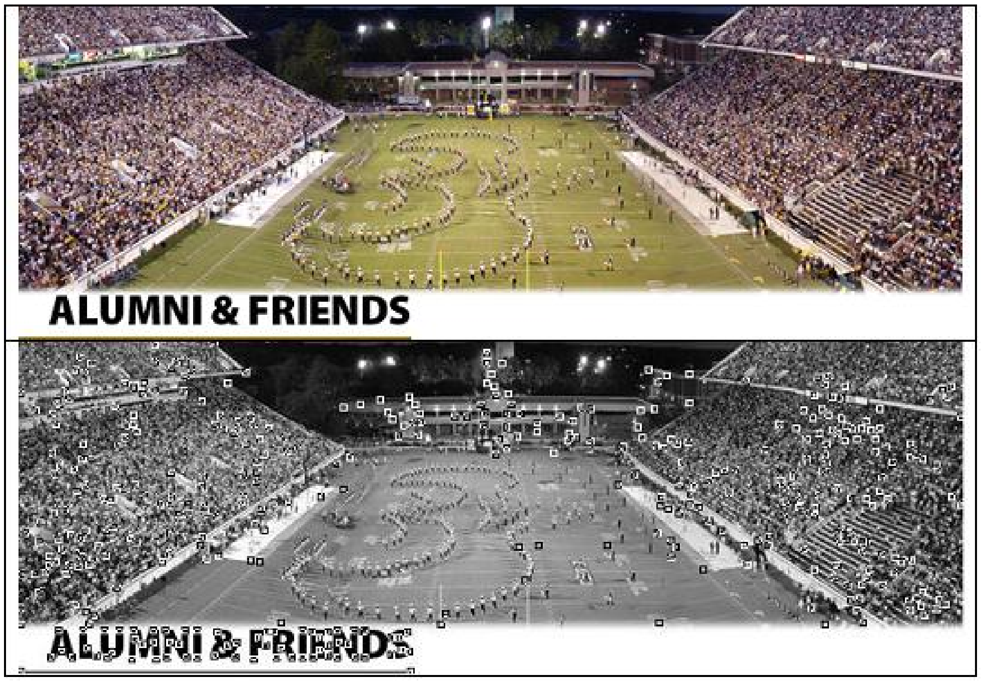


Figure 8.2: A 620 x 219 University of Southern Mississippi Recreational (Payne) Center original image and the resulting edge map.

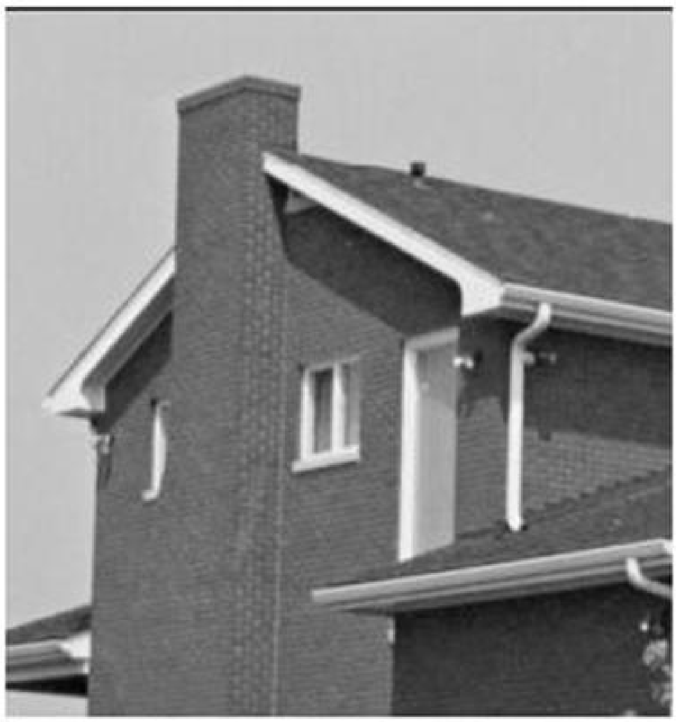
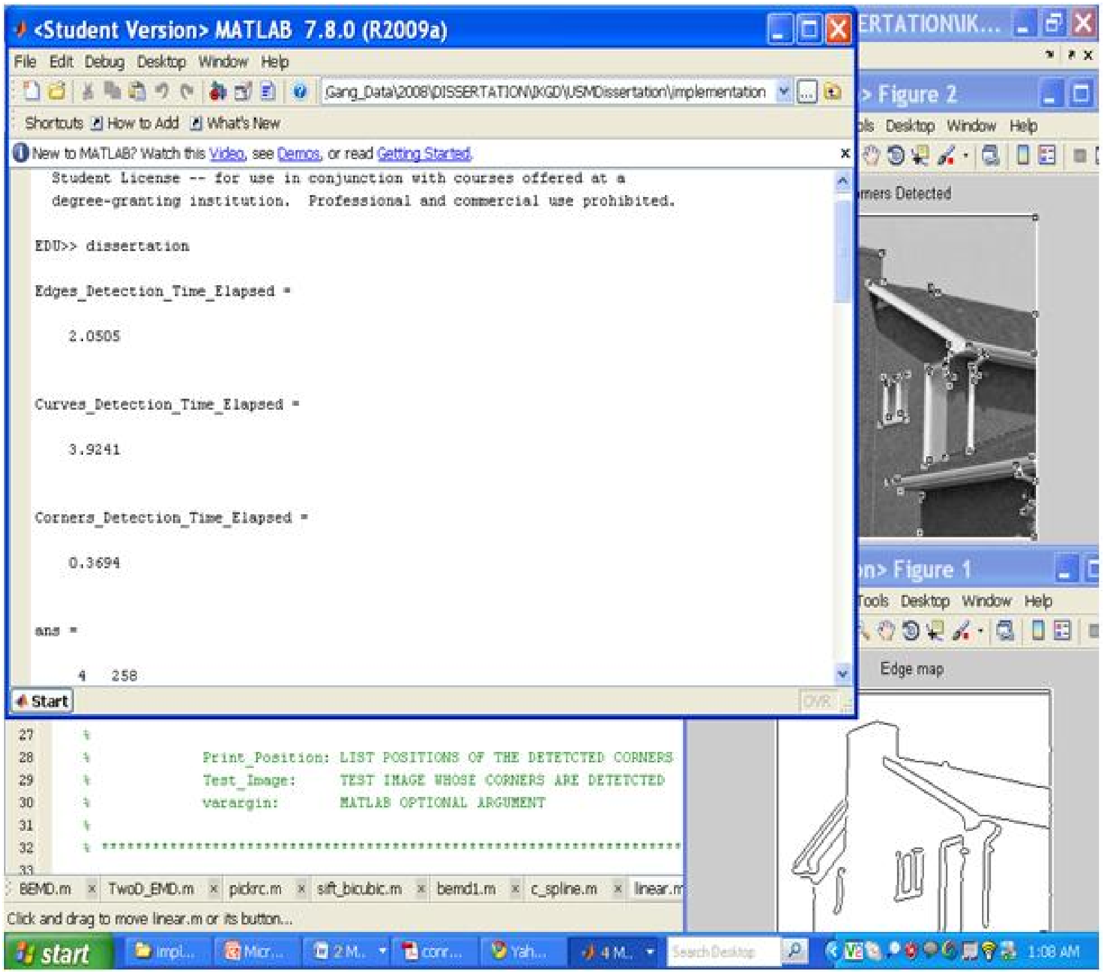


Figure 8.3: The House reference image.


<Student Version> MATLAB 7.8.0 (R2009a)
File Edit Debug Desktop Window Help
Gang_Data\2008\DISSERTATION\JKGD\USMDissertation\implementation
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Figure 2
Corners Detected
Figure 1
Edge map
Start
BEMD.m TwoD_EMD.m pickrc.m sift_bicubic.m bemd1.m c_spline.m linear.m
Click and drag to move linear.m or its button...
start
1:08 AM

```
  Student License -- for use in conjunction with courses offered at a
  degree-granting institution.  Professional and commercial use prohibited.

EDU>> dissertation

Edges_Detection_Time_Elapsed =

    2.0505


Curves_Detection_Time_Elapsed =

    3.9241


Corners_Detection_Time_Elapsed =

    0.3694


ans =

     4   258
```

```
    %
    %            Print_Position: LIST POSITIONS OF THE DETETCTED CORNERS
    %            Test_Image:     TEST IMAGE WHOSE CORNERS ARE DETETCTED
    %            varargin:       MATLAB OPTIONAL ARGUMENT
    %
    % *******************************************************************
```

Figure 8.4: Screen shots of the reference image.

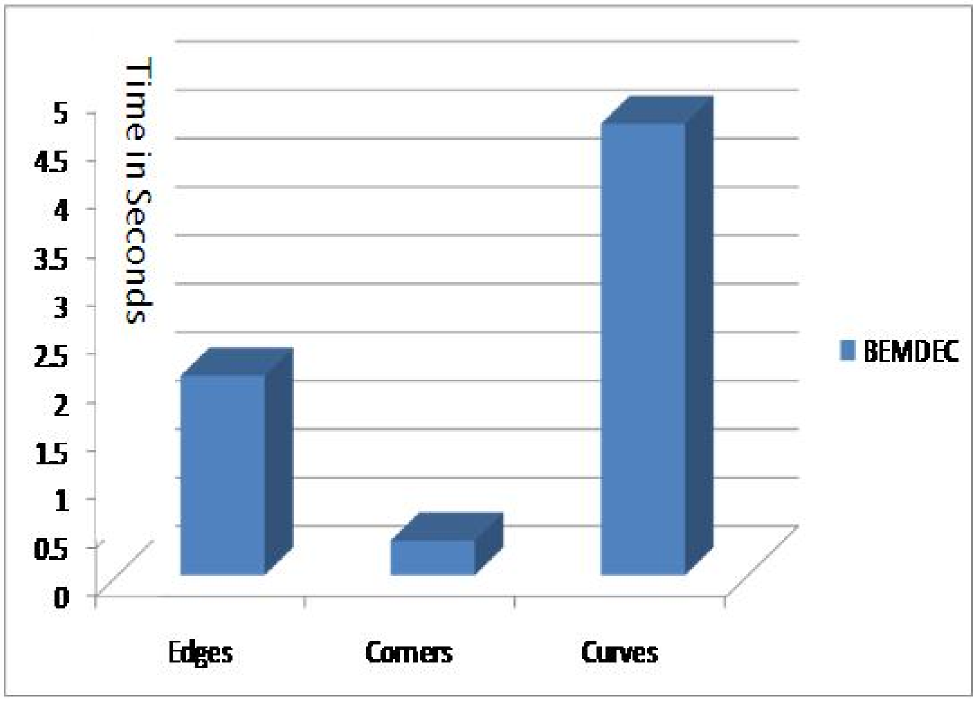


Figure 8.5: BEMDEC running time in seconds for the three features.

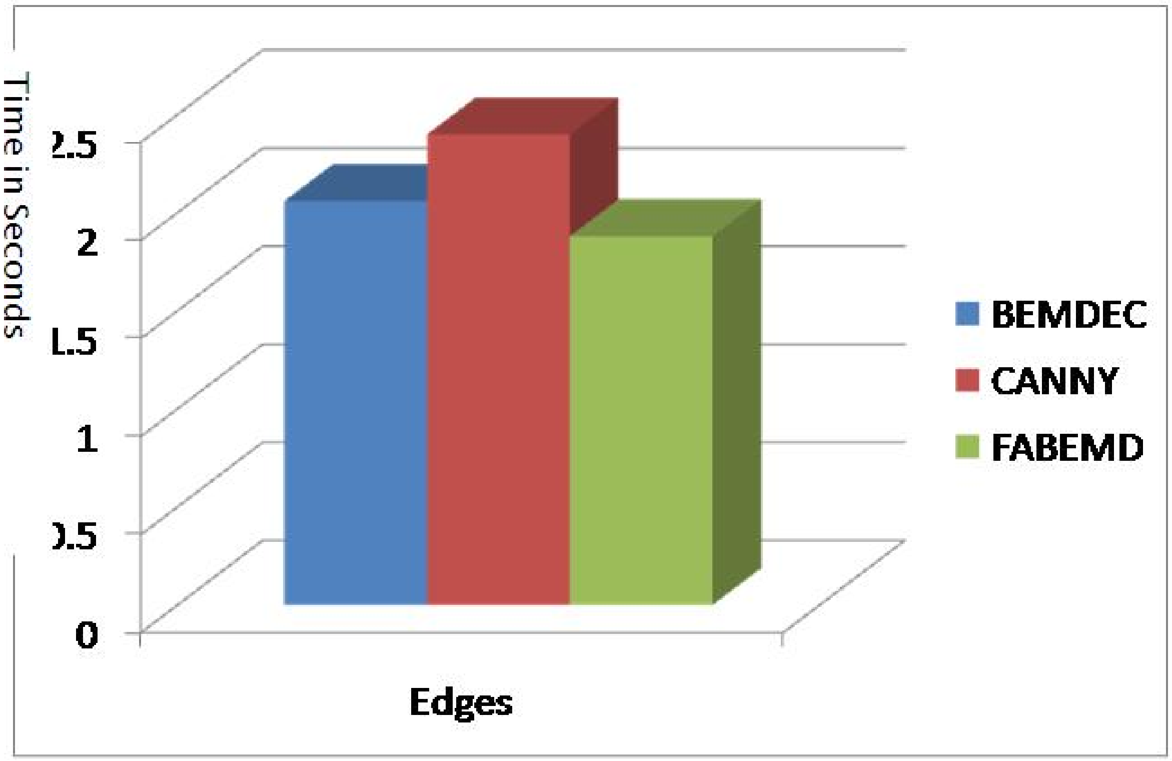


Figure 8.6: BEMDEC versus two other edge detectors of Canny FABEMD.

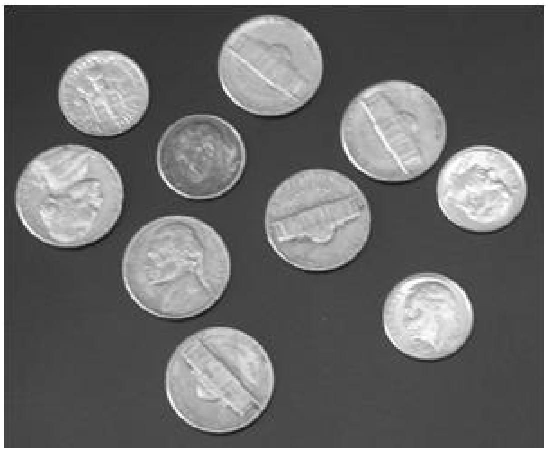


Figure 8.7: A 300 x 246 Coin Image used as reference.

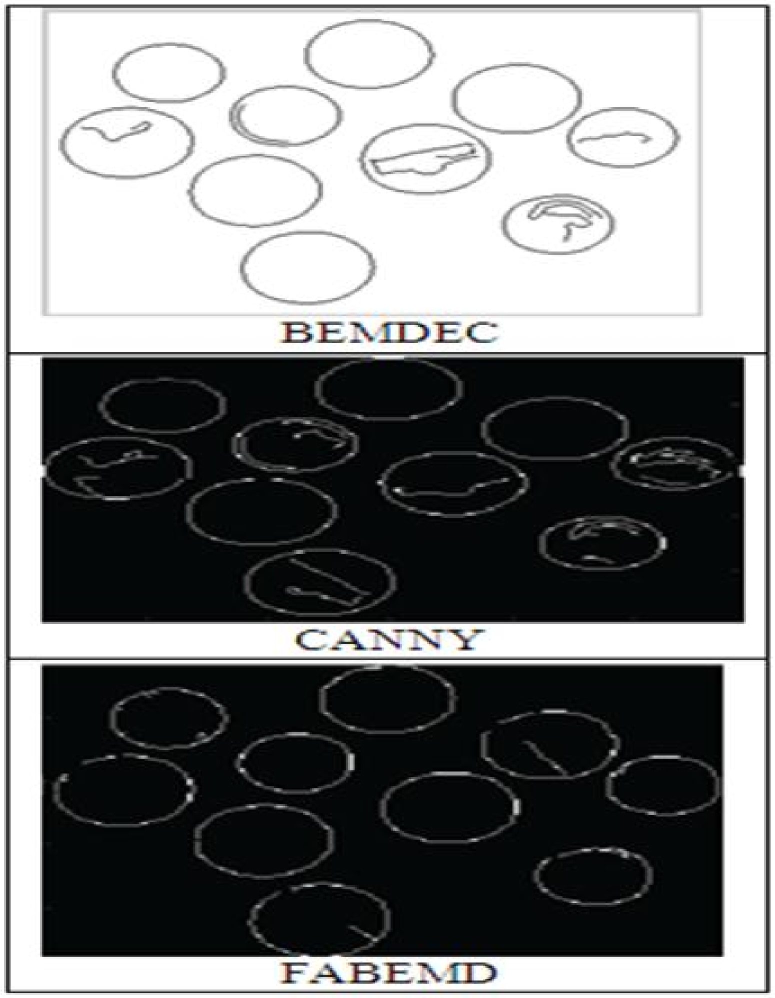


Figure 8.8: BEMDEC edge detection quality against the two competitors.

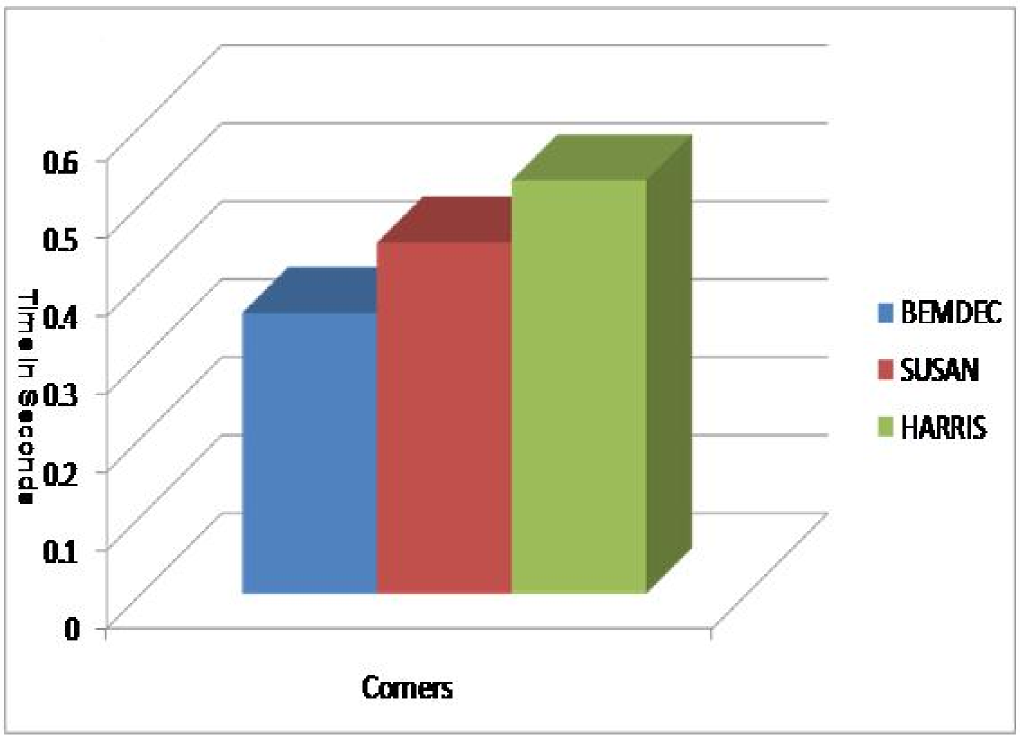


Figure 8.9: BEMDEC corner detection time in seconds versus two other corner operators of SUSAN and HARRIS.

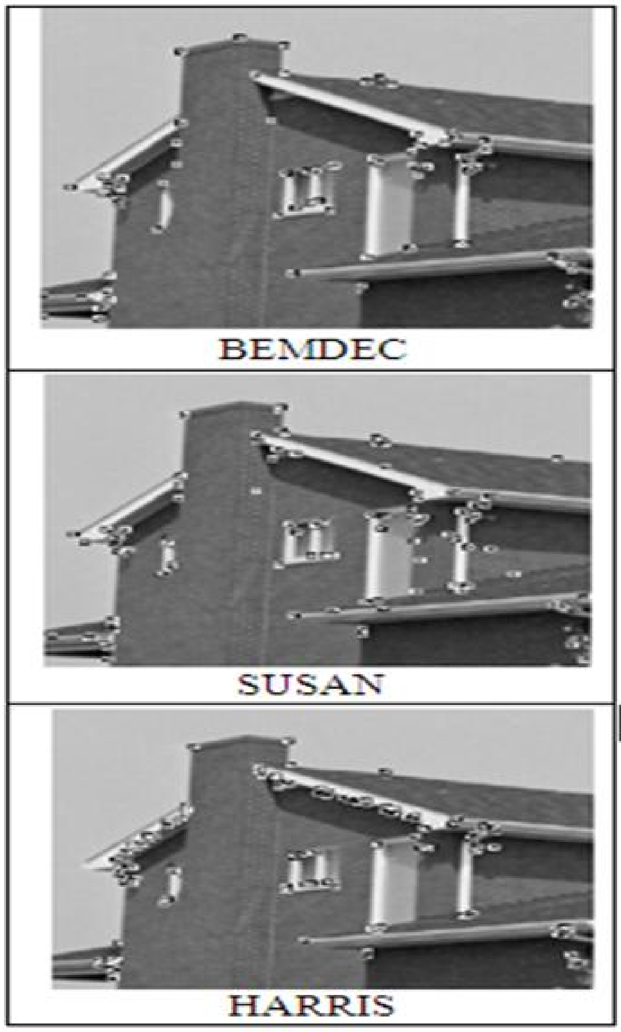


Figure 8.10: BEMDEC corner detection quality versus the two competitors.

8.2 Final Experimental Results

The second part of the result is the final experimental result for various images. The key idea is to use various variables to test the efficiency of the BEMDEC detector. The final results were meant to test the effectiveness of the BEMDEC methodology using two of the standard images for corner detection (House and Block references). The idea was to see the final performance using those standard images. After that, some screenshots for the Block reference were shown. Using the Block reference, a side-by-side comparison of BEMDEC and HARRIS was given, followed by another side-by-side comparison of BEMDEC and SUSAN.

Due to the inherent difficulties to obtain qualitative results in image process, an evaluation method to achieve that was devised in this dissertation and the result is a quantitative comparison between the three corner detection operators (BEMDEC, SUSAN and HARRIS).

The method proceeds as followed:

- F_{RI} = set of features from reference image (RI)
- F_{GM} = set of features found by a given method (GM)
- MD = maximum distance allowed between the detected and reference feature locations for which the detection is considered correct (default value = 3 pixels)
- FP_i, FP_j = (elements of RI and GM respectively) = set of feature points
- FP_i is considered correct detection of FP_j if the distance $d_{i,j}$ between the two feature points is minimum for all i and j and is less or equal to the maximum distance (MD).
- Otherwise FP_i is considered a missed feature
- All missed features in F_{RI} are considered undetected true features while the remaining features in F_{GM} are considered false detections
- The localization error is defined as the average of all the distances $d_{i,j}$ for correctly detected features

During this quantitative evaluation method, the performance of the three corner detection operators is evaluated using four parameters, namely; correct corners, false corners, missed corners and localization error using the Block reference with 64 manually identified corners. The same technique repeated for the three corner operators using the House reference with 74 manually identified corners.

Furthermore, a curve reference image is given to compare the curve detection methodology known as the Radon Transform against BEMDEC.

For the edge detection analysis in the final experimental results, two images (Lenna and Coins references) are provided. The first analysis there shows a sample run of four images (House, Block, Lenna and Admin) all different sizes. This was to see if there are any discrepancies in the BEMDEC methodology when dealing with images of different sizes. However, no major conclusions were drawn here and nothing significant happened. This led to running the operator with 50 images of the same sizes, and yet, there were no significant differences from running it with 1, 4, or 50 images. The same hold true when it is run with 50 images of the different sizes. That is, the running time did not change between the three edge detection operators (BEMDEC, Canny and FABEMD). The last two figures provided side-by-side comparisons of the BEMDEC against Canny and FABEMD respectively.

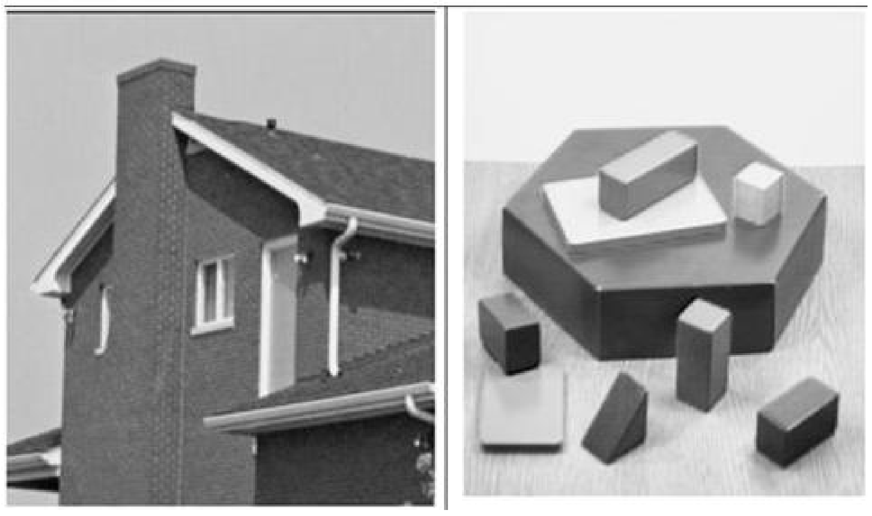
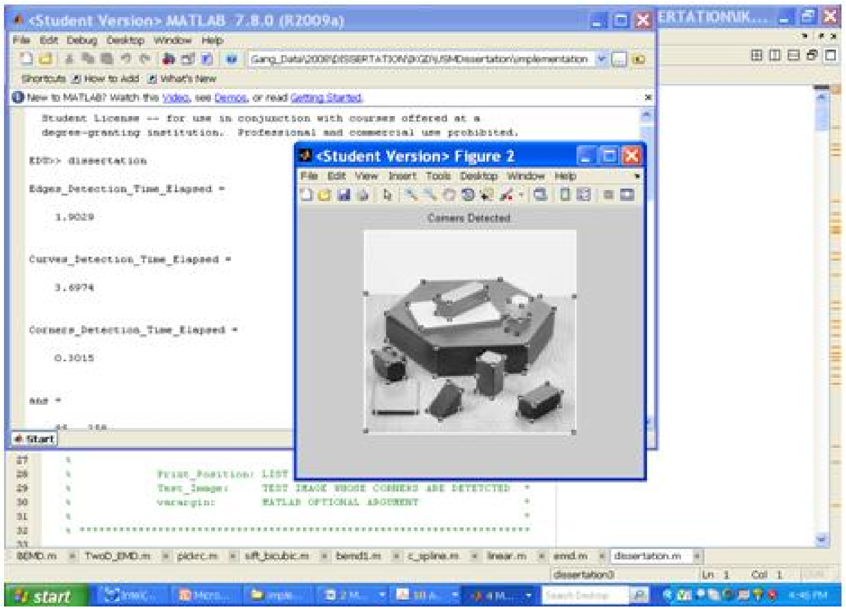


Figure 8.11: House and Block reference images for corners.


<Student Version> MATLAB 7.8.0 (R2009a)
File Edit Debug Desktop Window Help
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
<Student Version> Figure 2
File Edit View Insert Tools Desktop Window Help
Corners Detected
Start
BEMD.m TwoD_EMD.m pickrc.m sift_bicubic.m bemd1.m c_spline.m linear.m emd.m dissertation.m
start

```
  Student License -- for use in conjunction with courses offered at a
  degree-granting institution.  Professional and commercial use prohibited.

EDU>> dissertation

Edges_Detection_Time_Elapsed =

    1.9029


Curves_Detection_Time_Elapsed =

    3.6974


Corners_Detection_Time_Elapsed =

    0.3015


ans =
```

```
    %
    %            Print_Position: LIST
    %            Test_Image:     TEST IMAGE WHOSE CORNERS ARE DETETCTED  *
    %            varargin:       MATLAB OPTIONAL ARGUMENT                *
    %                                                                    *
    % ********************************************************************
```

Figure 8.12: Screen shots of the Block image with corners detected.

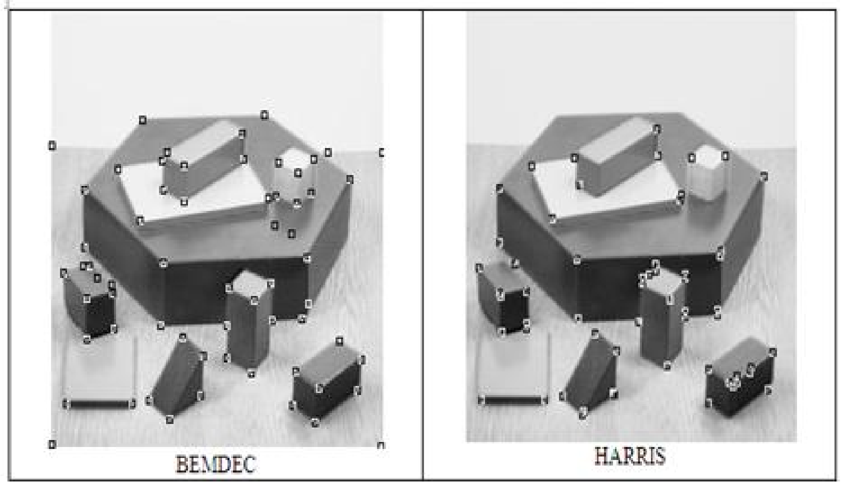


Figure 8.13: BEMDEC and HARRIS side-by-side comparison for corners.

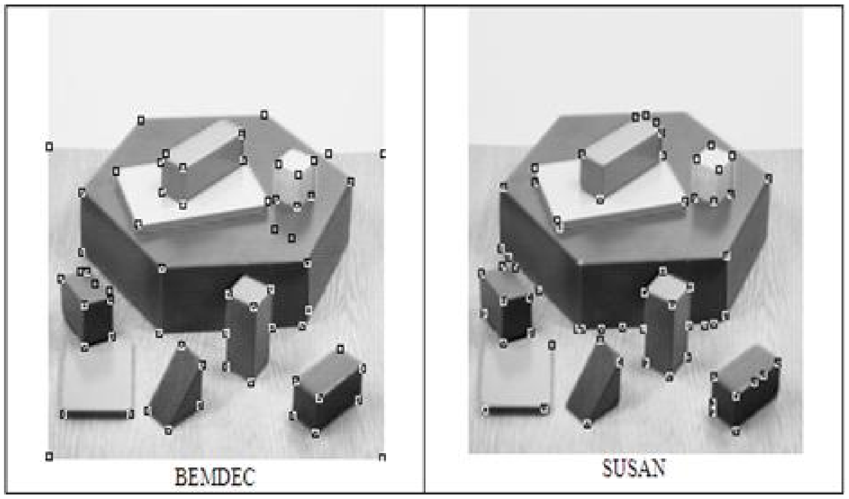
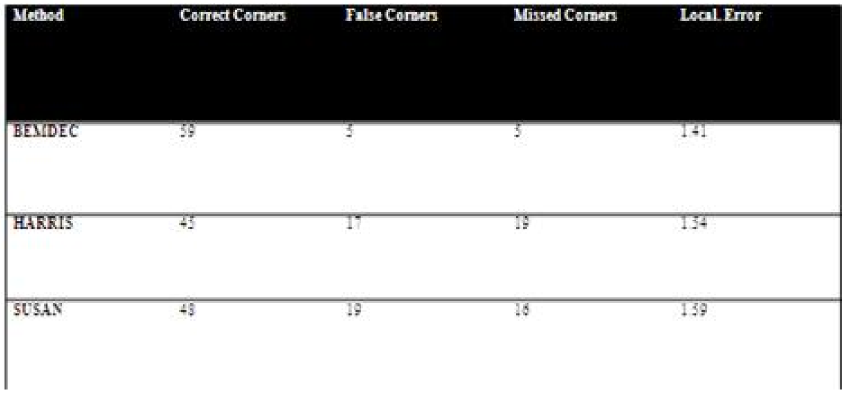
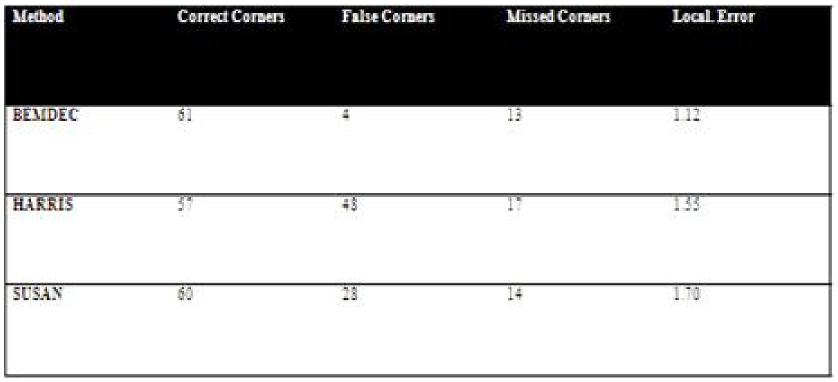


Figure 8.14: BEMDEC and SUSAN side-by-side comparison for corners.



Method	Correct Corners	False Corners	Missed Corners	Local. Error
BEMDEC	59	5	5	1.41
HARRIS	45	17	19	1.34
SUSAN	48	19	16	1.59

Figure 8.15: Reference (260 x 260 Block Image with 64 corners).



Method	Correct Corners	False Corners	Missed Corners	Local. Error
BEMDEC	61	4	13	1.12
HARRIS	57	48	17	1.55
SUSAN	60	28	14	1.70

Figure 8.16: Reference (424 x 346 House Image with 74 corners).

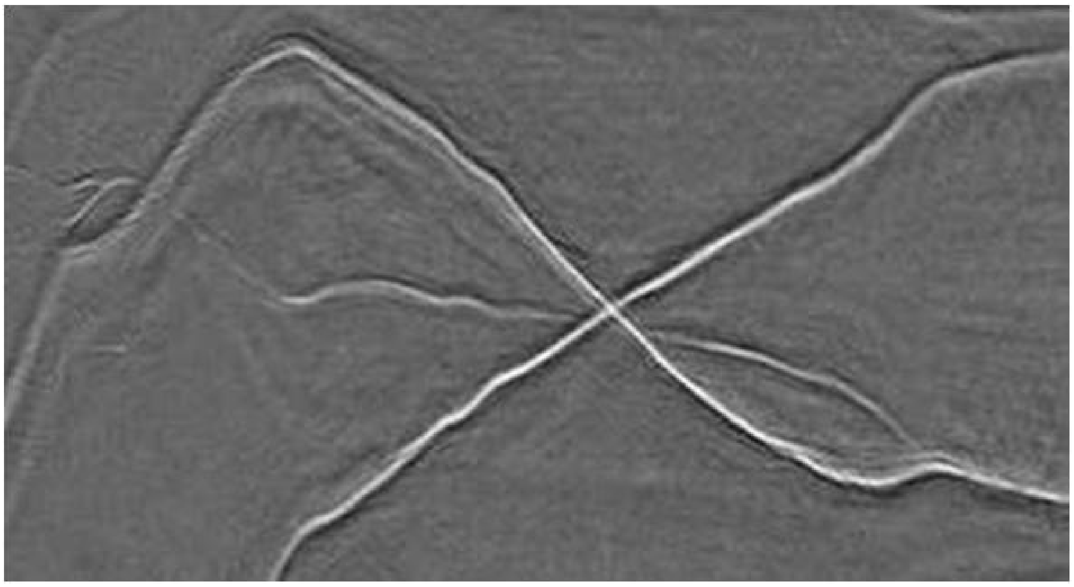


Figure 8.17: Reference image for curves.

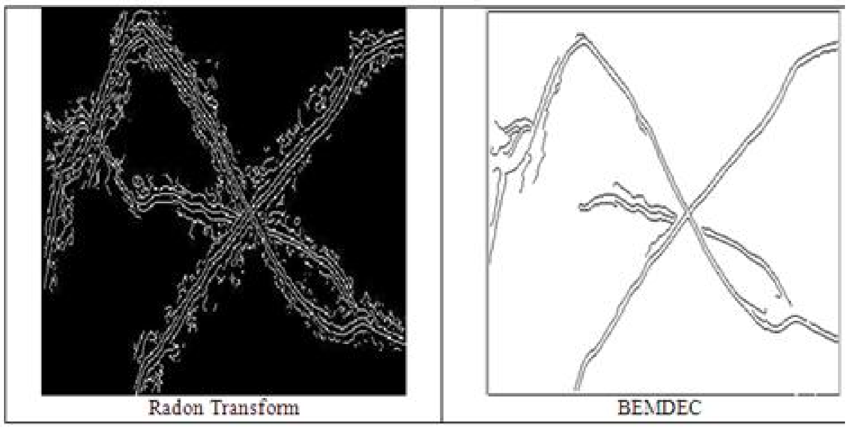


Figure 8.18: BEMDEC and Radon Transform side-by-side comparison for curves.

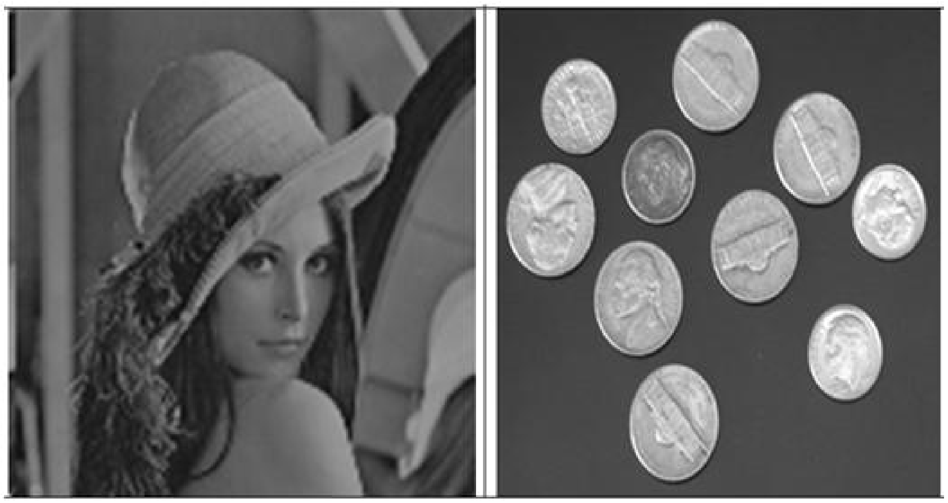
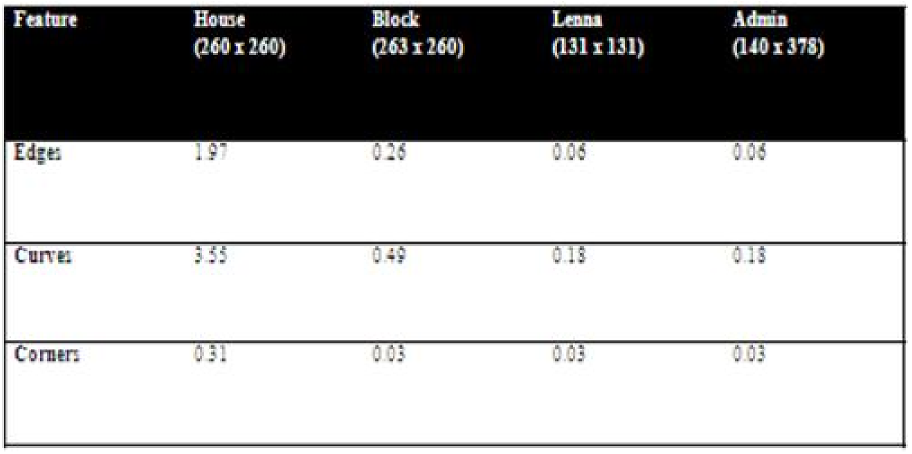


Figure 8.19: Lenna and coins reference images for edges.



Feature	House (260 x 260)	Block (263 x 260)	Lenna (131 x 131)	Admin (140 x 378)
Edges	1.97	0.26	0.06	0.06
Curves	3.55	0.49	0.18	0.18
Corners	0.31	0.03	0.03	0.03

Figure 8.20: Sample run of 4 images (with different sizes).

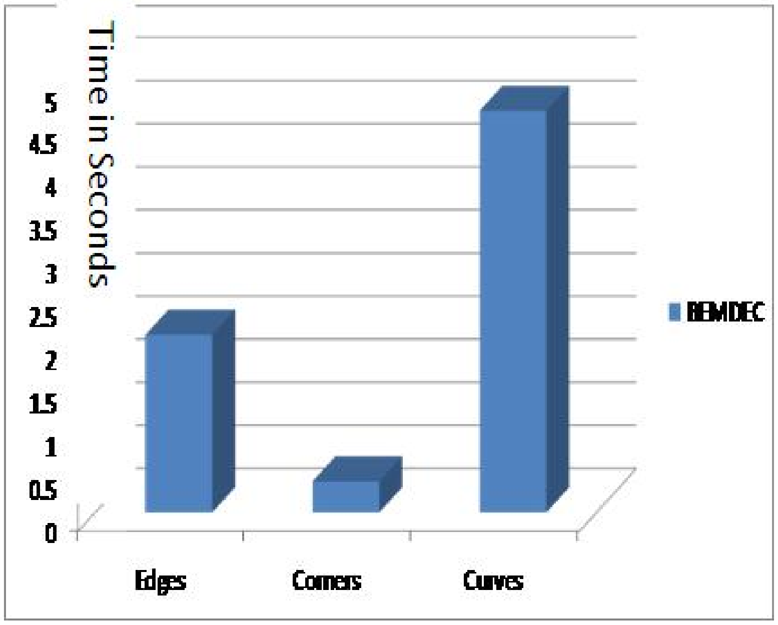


Figure 8.21: Sample run of 50 images (with same size).

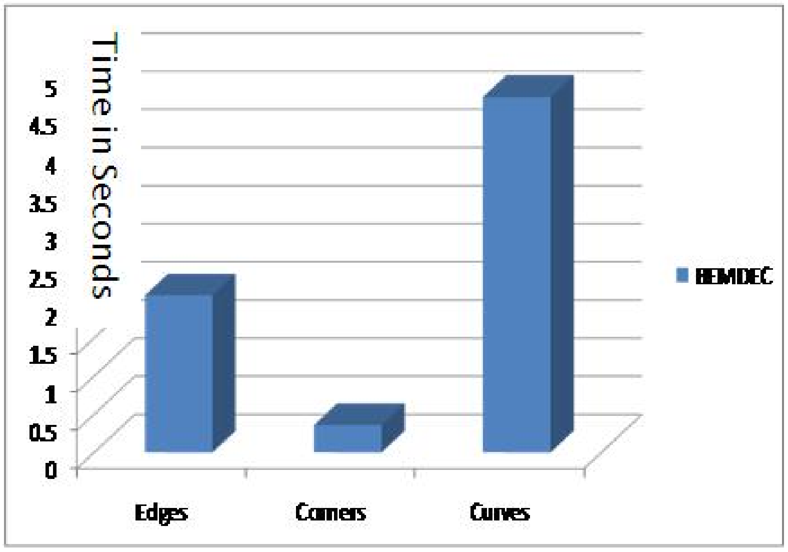


Figure 8.22: Sample run of 50 images (with different sizes).

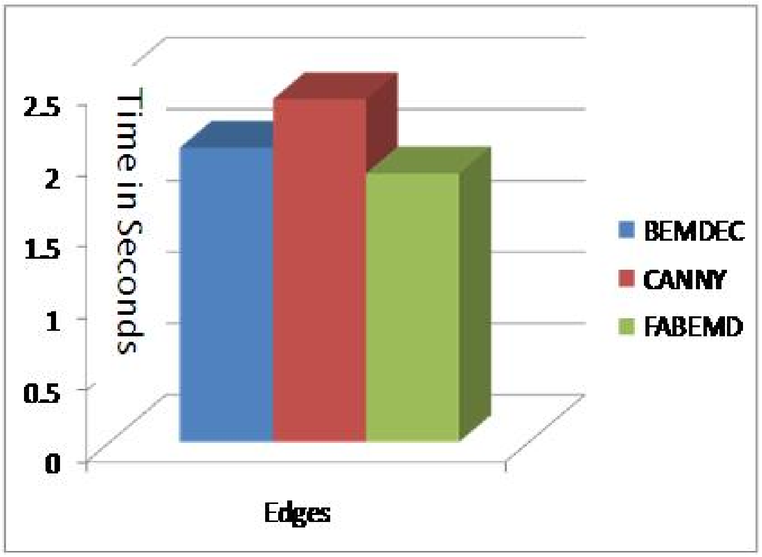


Figure 8.23: After 2 runs (No significant changes in time for the three operators).

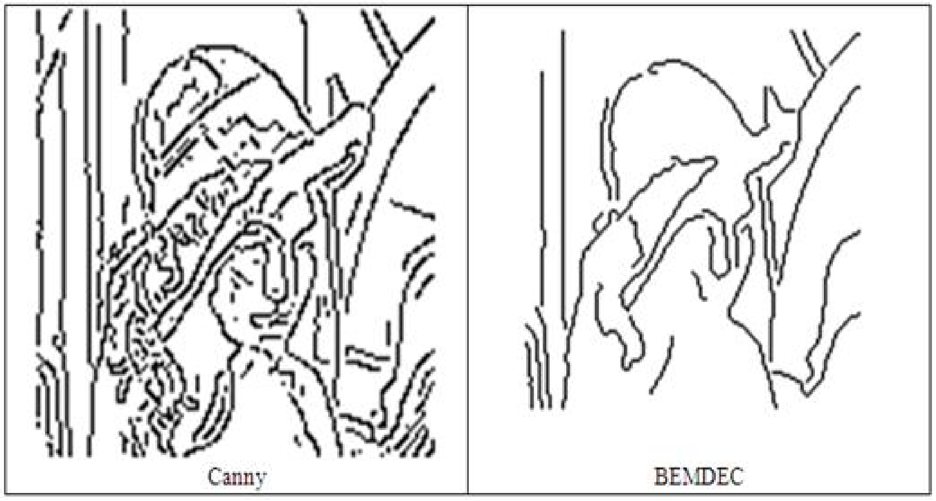


Figure 8.24: BEMDEC and Canny side-by-side Lenna image comparison.

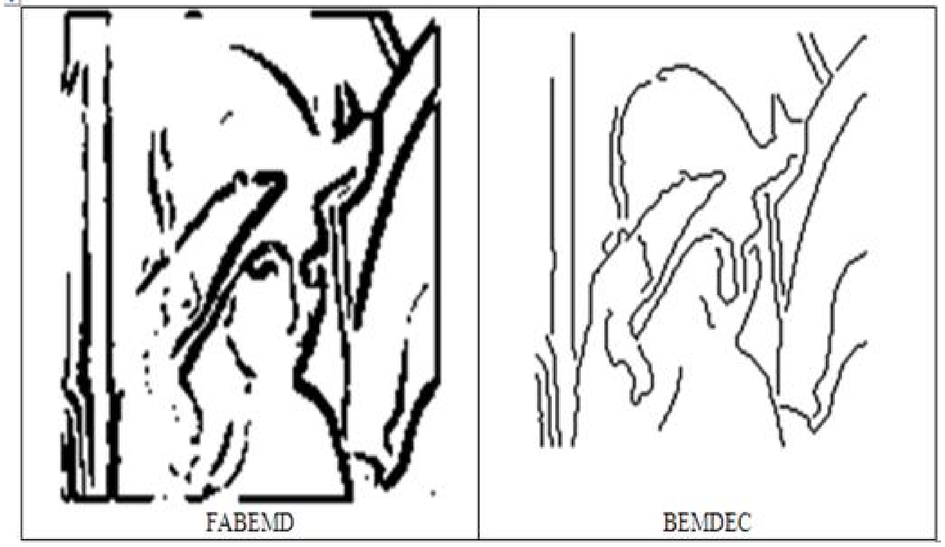


Figure 8.25: BEMDEC and Canny side-by-side Lenna image comparison.

8.3 Results Analysis

From the above results (preliminary and final), it can be concluded that BEMDEC is a versatile feature detector that is able to detect edges, corners and curves. Furthermore, a conclusion can also be drawn that it demonstrated reasonable consistency and stability in terms of running cost, and edge maps production.

8.4 Conclusion

BEMDEC, derived from bidimensional empirical mode decomposition (BEMD), is proposed in this dissertation to detect multi feature. Its performance shown promise in detecting feature of a one-band, sometimes referred to as grayscale, intensity, or monochromatic, images. As demonstrated by the experimental results, BEMDEC performs equally, and in some case, better than some of the closely related methods, both BEMD-based as well as traditional ones. To this end, we can reasonably concluded that this method can perform reasonably well when applied to real world applications. It should, however, be noted that this was not done during the testing phase.

Despite the potential of BEMDEC, several concerns and observation should be pointed out. First, BEMDEC seem to take longer time when decomposing an image whose curve profiles are not apparent. Also the edge detection time seem to be higher than the corner detection for many of the test images, especially when the pixels size increases. But because edge is the primary concern of this work, it is the desirable to have this situation reversed.

In addition, the proposed BEMDEC boundary adjustment, though effective, is arbitrary. One observation with this scheme is that it tended to show inconsistency in the image quality when the values of the evaluation function and the global threshold are higher. However, this was remedied by setting these parameters to fixed values.

8.5 Future Direction

BEMDEC, derived from bidimensional empirical mode decomposition (BEMD), is proposed in this dissertation to detect multi feature. Its performance shown promise in detecting feature of a one-band, sometimes referred to as grayscale, intensity, or monochromatic, images. As demonstrated by the experimental results, BEMDEC performs equally, and in some case, better than some of the closely related methods, both BEMD-based as well as traditional ones. To this end, we can reasonably concluded that this method can perform reasonably well when applied to real world applications. It should, however, be noted that this was not done during the testing phase.

Nevertheless, several concerns and observation should be pointed out. First, BEMDEC seem to take longer time when decomposing an image whose curve profiles are not apparent. The edge detection time seem to be higher than the corner detection for many of the test images, especially when the pixels size increases. But because edge is the primary concern of this work, it is the desirable to have this situation reversed. As such, further investigation into the algorithmic improvement to accomplish this situation could be pursued in the future.

In addition, the proposed BEMDEC boundary adjustment, though effective, is arbitrary. One observation with this scheme is that it tended to show inconsistency in the image quality when the values of the evaluation function and the global threshold are higher. Although this was remedied by setting these parameters to fixed values, future investigation may be necessary to make it adaptable so that it can handle higher values of these parameters effectively.

Furthermore, the edge detection methods discussed in this work are differential operators based and are widely used in the processing of a one-band, sometimes referred to as grayscale, intensity, or monochromatic, images [66, 67]. In contrast, multi-band, multispectral or color images have been seldom studied. Difficulties in dealing with the three components of the red-green-blue (RGB) have foiled the small effort in developing the appropriate color edge detection algorithm. Typically, one-band or the grayscale image processing is made possible by the application of either the zero-crossing, sometimes referred to as Laplacian or the gradient method [66, 1].

Few researchers who have tried to persist in their quest to develop multispectral edge detection operators are usually met with the sheer difficulties of extending the monochromatic versions of the one band image into a reasonable combination of the RGB components [67]. The first known attempt to extend a monochromatic edge detector to a multispectral one, to the author's knowledge, was introduced in 1977 by Professor Ramakant Nevatia of the University of Southern California (USC) in a journal paper where he proposed to extend Hueckel operator, developed four years earlier, to multispectral images [68, 67]. Though some improvements have been made toward this direction, BEMDEC could possibly be useful due to its flexibility and adaptability. However, some level of algorithmic augmentation would be necessary. This is an area that is of great interest and will be investigated in the future.

Appendix A

Fourier Transforms

A.1 Discrete Fourier Transform (DFT)

The discrete Fourier transform (DFT) is a specific kind of Fourier transform, used in Fourier analysis. It transforms one function into another, which is called the frequency domain representation, or simply the DFT, of the original function (which is often a function in the time domain).

A.2 Fast Fourier Transform (FFT)

A fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse. There are many distinct FFT algorithms involving a wide range of mathematics, from simple complex-number arithmetic to group theory and number theory.

A.3 Discrete-Time Fourier Transform (DTFT)

Discrete-time Fourier transform (DTFT) is one of the specific forms of Fourier analysis. As such, it transforms one function into another, which is called the frequency domain representation, or simply the ’DTFT’, of the original function (which is often a function in the time-domain). But the DTFT requires an input function that is discrete. Such inputs are often created by sampling a continuous function, like a person’s voice.

Appendix B

Wavelet Transforms

B.1 Discrete Wavelet Transform (DWT)

The fast Fourier transform (FFT) discribed in apendix A and the discrete wavelet transform (DWT) are both linear operations that generate a data structure that contains segments of various lengths, usually filling and transforming it into a different data vector of length. The mathematical properties of the matrices involved in the transforms are similar as well. The inverse transform matrix for both the FFT and the DWT is the transpose of the original. As a result, both transforms can be viewed as a rotation in function space to a different domain. For the FFT, this new domain contains basis functions that are sines and cosines. For the wavelet transform, this new domain contains more complicated basis functions called wavelets, mother wavelets, or analyzing wavelets.

Both transforms have another similarity. The basis functions are localized in frequency, making mathematical tools such as power spectra (how much power is contained in a frequency interval) and scalegrams useful at picking out frequencies and calculating power distributions. The most interesting dissimilarity between these two kinds of transforms is that individual wavelet functions are localized in space. Fourier sine and cosine functions are not.

B.2 Fast Wavelet Transform (FWT)

The Fast Wavelet Transform is a mathematical algorithm designed to turn a waveform or signal in the time domain into a sequence of coefficients based on an orthogonal basis of small finite waves, or wavelets. The transform can be easily extended to multidimensional signals, such as images, where the time domain is replaced with the space domain.

Appendix C

Gabor Transform

C.1 Gabor Transform Background

The Gabor transform is a special case of the short-time Fourier transform. It is used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time. The function to be transformed is first multiplied by a Gaussian function, which can be regarded as a window, and the resulting function is then transformed with a Fourier transform to derive the time-frequency analysis.

Appendix D

Gabor-Wigner Transform

D.1 Gabor-Wigner Transform Overview

The Gabor transform and the Wigner distribution function are both tools for time-frequency analysis. Since the Gabor transform described in Apendix C does not have high clarity, and the Wigner distribution function has a cross term problem, a new combination of the two transforms that has high clarity and no cross term problem was proposed by Pei and Ding in 2007. Since the cross term does not appear in the Gabor transform, the time frequency distribution of the Gabor transform can be used as a filter to filter out the cross term in the output of the Wigner distribution function.

BIBLIOGRAPHY

[1] J. F. Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, November 1986.

[2] S. M. A. Bhuiyan, *Bidimensional Empirical Mode Decomposition for Image Processing Using Order-statistic Filter Based Envelop Estimation*. Phd. dissertation, University of Alabama at Huntsville, Huntsville, March 2009.

[3] Z. Hou and G. Wei, “A new approach to edge detection,” *Pattern Recognition*, vol. 35, pp. 1559–1570, June 2001.

[4] Pietro and J. Malik, “Scale-space and edge detection using anisotropic diffusion,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 629–639, July 1990.

[5] J. F. Canny, “Finding edges and lines in images,” master’s thesis, Massachusetts Institute of Technology, Cambridge, Jun. 1983.

[6] L. Xi’ang, Y. Donghe, X. Jinsong, and L. Yingchun, “Cocoon edge detection based on self-adaptive canny operator,” in *Proc. International Conference on Computer Science and Software Engineering (CSSE’08)*, (Wuhan, China), pp. 7–10, December 2008.

[7] J. F. Canny, “A variational approach to edge detection,” in *Proc. National Conference on Artificial Intelligence (CAI’83)*, (Washington, DC, USA), pp. 22–26, August 1983.

[8] D. Marr and E. Hildreth, “Theory of edge detection,” *Royal Society of London*, vol. B 207, pp. 187–217, February 1980.

[9] M. D. Heath, “A robust visual method for assessing the relative performance of edge detection algorithms,” master’s thesis, University of South Florida, Dec. 1996.

[10] M. H. Yap, H. Ugail, R. Zwiggelaar, and B. Rajoub, “A short review of methods for face detection and multifractal analysis,” in *Proc. International Conference on CyberWorlds (CW’09)*, (Bradford, UK), pp. 231–236, September 2009.

[11] H. S. Neoh and A. Hazanchuk, “Adaptive edge detection for real-time video processing using fpgas,” in *Proc. Global Signal Processing Expo and Conference (GSPX’04)*, (Santa Clara, CA, USA), pp. 1–7, September 2004.

[12] X. Wang and J.-Q. Jin, “An edge detection algorithm based on improved canny operator,” in *Proc. 7th International Conference on Intelligent Systems Design and Applications (ISDA’07)*, (Rio de Janeiro, Brazil), pp. 623–628, October 2007.

[13] A. Rosenfeld and M. Thurston, “Edge and curve detection for visual scene analysis,” *IEEE Transactions on Computers*, vol. 20, pp. 562–569, May 1971.

[14] D. Marr, “Analyzing natural images: a computational theory of texture vision,” in *Proc. Cold Spring Harbor Symposia on Quantitative Biology*, (Long Island, New York, USA), pp. 647–662, June 1975.

[15] Y. Yao and H. Ju, “A sub - pixel edge detection method based on canny operator,” in *Proc. 6th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD’09)*, (Tianjin, China), pp. 97–100, August 2009.

[16] D. Marr and T. Poggio, “A computational theory of human stereo vision,” *Royal Society of London*, vol. 204, pp. 301–328, May 1979.

[17] D. Marr, “Early processing of visual information,” *Philosophical Transactions of the Royal Society*, vol. B 275, pp. 483–524, December 1975.

[18] R. M. Shapley and D. J. Tolhurst, “Edge detectors in human vision,” *Journal of Physiology*, vol. 229, pp. 165–183, February 1973.

[19] E. Nadernejad, S. Sharifzadeh, and H. Hassanpour, “Edge detection techniques: Evaluations and comparisons,” *Applied Mathematical Sciences*, vol. 2, pp. 1507–1520, January 2008.

[20] D. Marr, “A theory for cerebral neocortex,” *Royal Society of London*, vol. 76, pp. 161–264, November 1970.

[21] D. Marr, “Representing visual information,” Tech. Rep. AIM 415, Massachusetts Institute of Technology, Cambridge, May 1977.

[22] M. Heath, S. Sarkar, T. Sanocki, and K. Bowyer, "Comparison of edge detectors: A methodology and initial study," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, (San Francisco, CA, USA), pp. 143–183, June 1996.

[23] M. C. Motwani, M. C. Gadiya, R. C. Motwani, and J. Frederick C. Harris, "Survey of image denoising techniques," in *Proc. Global Signal Processing Expo and Conference (GSPx'04)*, (Santa Clara, California, USA), pp. 1–7, September 2004.

[24] A. Buades, B. Coll, and J. M. Morel, "On image denoising methods," Tech. Rep. 2004-15, Centre de Mathematiques et de Leurs Applications - CMLA, Cachan, France, March 2004.

[25] Q. Li, Y. Liang, and Y. Peng, "A comparative study of the anti-noise ability about cubic b-spline wavelet and canny algorithm," in *Proc. International Conference on Computer Science and Software Engineering (CSSE'08)*, (Wuhan, China), pp. 911–914, December 2008.

[26] S. Dangeti, "Denoising techniques - a comparison," master's thesis, Louisiana State University, May 2003.

[27] R. Oktem, K. Egiazarian, V. V. Lukin, N. N. Ponomarenko, and O. V. Tsymbal, "Locally adaptive dct filtering for signal-dependent noise removal," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, pp. 1–10, September 2007.

[28] Y. Li and F. Santosa, "A computational algorithm for minimizing total variation in image restoration," *IEEE Transactions on Image Processing*, vol. 5, pp. 987–995, June 1996.

[29] L. M. Lui, S. Thiruvenkadam, Y. Wang, P. Thompson, and T. Chan, "Optimized conformal parameterization of cortical surfaces using shape based landmark matching," in *Proc. 11th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'08, Part I)*, (New York, NY, USA), pp. 494–501, January 2008.

[30] M. S. Brown, M. Sun, R. Yang, L. Yun, and W. B. Seales, "Restoring 2d content from distorted documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 1904–1916, November 2007.

[31] M. Zhu, S. J. Wright, and T. F. Chan, "Duality-based algorithms for total-variation-regularized image restoration," *Computational Optimization and Applications*, vol. 41, pp. 1–24, January 2008.

[32] S. H. Bae, *Information Retrieval Via Universal Source Coding*. Phd. dissertation, Georgia Institute of Technology, Atlanta, November 2008.

[33] Y. Lou, X. Zhang, S. Osher, and A. Bertozzi, “Image recovery via nonlocal operators,” *Journal of Scientific Computing*, vol. 42, pp. 185–197, February 2010.

[34] P.-Y. Liu and H.-X. Li, “Fuzzy techniques in image restoration researchŮa survey,” *International Journal of Computational Cognition*, vol. 2, pp. 131–149, June 2004.

[35] Z. Wei, Y. Cao, and A. R. Newton, “Digital image restoration by exposure-splitting and registration,” in *Proc. 17th International Conference on Pattern Recognition (ICPR’04)*, (Cambridge, UK), pp. 657–660, August 2004.

[36] N. R. Harvey and S. Marshall, “Application of non-linear image processing: Digital video archive restoration,” in *Proc. International Conference on Image Processing (ICIP’97)*, (Washington, DC, USA), pp. 731–734, October 1997.

[37] M. Zhu and T. Chan, “An efficient primal-dual hybrid gradient algorithm for total variation image restoration,” Tech. Rep. UCLA CAM Report 08-34, University of California at Los Angles, Los Angeles, May 2008.

[38] K. Adi, T. L. Mengko, A. B. Suksmono, and D. Danudirdjo, “Digital image restoration using posterior distribution and updating pixel by self threshold,” in *Proc. SICE-ICASE International Joint Conference (’06)*, (Bexco, Busan, Korea), pp. 5707–5710, October 2006.

[39] B. Zitova and J. Flusser, “Image registration methods: a survey,” *Image and Vision Computing*, vol. 21, pp. 977–1000, June 2003.

[40] L. G. Brown, “A survey of image registration techniques,” *ACM Computing Surveys*, vol. 24, pp. 325–376, December 1992.

[41] H. Zhou, *Analysis of Vector-Directed and Best-parameter Search Algorithms for Transformation Function Construction in Automatic Image Registration*. Phd. dissertation, University of Southern Mississippi, Hattiesburg, August 2004.

[42] T. Aydin, Y. Yehfez, B. Sanliur, E. Anarim, and O. Alkin, “Use of m-band wavelet transform for multidirectional and multiscale edge detection,” in *Proc. IEEE International Confer-*

ence on Acoustics, Speech, and Signal Processing (ICASSP'94), (Adelaide, SA, Australia), pp. 17–20, April 1994.

[43] T. Surgailis, A. Valinevicius, and M. Zilys, "Traffic image processing systems," in *Proc. 2nd International Conference on Advances in Circuits, Electronics and Micro-Electronics (CENICS'09)*, (Sliema, Malta), pp. 61–66, October 2009.

[44] Z. Huan, L. Xiuhuan, and Y. Lilei, "Shot boundary detection based on mutual information and canny edge detector," in *Proc. International Conference on Computer Science and Software Engineering (CSSE'08)*, (Wuhan, China), pp. 1124–1128, December 2008.

[45] M. Das and J. Anand, "Robust edge detection in noisy images using an adaptive stochastic gradient technique," in *Proc. International Conference on Image Processing (ICIP'95)*, (Washington D.C, USA), pp. 2149–2152, October 1995.

[46] J. Dong, M. Jian, D. Gao, and S. Wang, "Reducing the dimensionality of feature vectors for texture image retrieval based on wavelet decomposition," in *Proc. 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'07)*, (Qingdao, China), pp. 758–763, July 2007.

[47] A. Chambolle and J. Darbon, "On total variation minimization and surface evolution using parametric maximum flows," *International Journal of Computer Vision*, vol. 84, pp. 288–307, September 2009.

[48] L. G. Roberts, *Machine perception of three-dimensional solids*. Phd. dissertation, Massachusetts Institute of Technology, Cambridge, June 1963.

[49] Z. Ajmal, J.-Y. Bouguet, and R. M. Mersereau, "Learning a face model for tracking and recognition," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP'02)*, (Orlando, Florida, USA), pp. 3612–3615, May 2002.

[50] Y.-P. Wong, V. C.-M. Soh, K.-W. Ban, and Y.-T. Bau, "Improved canny edges using ant colony optimization," in *Proc. 5th International Conference on Computer Graphics, Imaging and Visualization (CGIV'08)*, (Penang, Malaysia), pp. 197–202, August 2008.

[51] J. F. Khan, K. Barner, and R. Adhami, “Feature point detection utilizing the empirical mode decomposition,” *EURASIP Journal on Advances in Signal Processing*, vol. 2008, pp. 1–13, January 2008.

[52] T. Patil, “Evaluation of multi-core architectures for image processing algorithms,” master’s thesis, Clemson University, Aug. 2009.

[53] W.-Y. Ma and B. S. Manjunath, “Edgeflow: a technique for boundary detection and image segmentation,” *IEEE Transactions on Image Processing*, vol. 9, pp. 1375–1388, August 2000.

[54] K. Bowyer, C. Kranenburg, and S. Dougherty, “Edge detector evaluation using empirical roc curves,” in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’99)*, (Fort Collins, Colorado, USA), pp. 1354–1359, June 1999.

[55] F. Yan, X. Shao, G. Li, Z. Sun, and Z. Yang, “Edge detection of tank level ir imaging based on the auto-adaptive double-threshold canny operator,” in *Proc. 2nd International Symposium on Intelligent Information Technology Application (IITA’08)*, (Shanghai City, China), pp. 366–370, December 2008.

[56] C. Li, K. He, and J. Zhou, “Edge detection of image on the local feature,” in *Proc. 2nd International Symposium on Intelligent Information Technology Application (IITA’08)*, (Shanghai City, China), pp. 326–330, December 2008.

[57] L. Zhou, G. Hua, D. Xu, and H. Wu, “Edge detection algorithm for uneven lighting image based on vision theory,” in *Proc. International Conference on Computational Intelligence and Natural Computing (CINC’09)*, (Wuhan, China), pp. 182–185, June 2009.

[58] D. Marr and S. Ullman, “Directional selectivity and its use in early visual processing,” *Royal Society of London*, vol. 211, pp. 151–580, March 1981.

[59] G.S.Hollingworth, S.L.Smith, and A.M.Tyrrell, “Design of highly parallel edge detection nodes using evolutionary techniques,” in *Proc. 7th Euromicro Workshop on Parallel and Distributed Processing (PDP’99)*, (Funchal, Portugal), pp. 35–42, February 1999.

[60] D. Marr, T. Poggio, and S. Ullman, “Bandpass channels, zero-crossings, and early visual information processing,” *Journal of the Optical Society of America*, vol. 69, pp. 914–916, September 1979.

[61] F. Campbell and J. Robson, “Application of fourier analysis to the visibility of gratings,” *Journal of Physiology*, vol. 197, pp. 551–566, August 1968.

[62] B. Wang and S. Fan, “An improved canny edge detection algorithm,” in *Proc. 2nd International Workshop on Computer Science and Engineering (WCSE’09)*, (Qingdao, China), pp. 497–500, October 2009.

[63] M. H. Hueckel, “An operator which locates edges in digitized pictures,” *Journal of the ACM (JACM)*, vol. 18, pp. 113–125, January 1971.

[64] D. Marr, “Representing visual information,” *Royal Society of London*, vol. 76, pp. 161–264, November 1970.

[65] G. Li, Y. Tong, and X. Xiao, “New image edge detection approach by using grey entropy,” in *Proc. 2nd International Symposium on Knowledge Acquisition and Modeling (KAM’09)*, (Wuhan, China), pp. 209–212, November 2009.

[66] A. Cumani, “Edge detection in multispectral images,” *Graphical Models and Image Processing*, vol. 53, pp. 40–51, January 1991.

[67] H. Tao and T. S. Huang, “Color image edge detection using cluster analysis,” in *Proc. International Conference on Image Processing (ICIP’97)*, (Santa Barbara, CA, USA), pp. 834–836, October 1997.

[68] M. A. Ruzon and C. Tomasi, “Color edge detection with the compass operator,” in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’99)*, (Fort Collins, CO, USA), pp. 160–166, June 1999.

[69] M. Omachi and S. Omachi, “Traffic light detection with color and edge information,” in *Proc. 2nd IEEE International Conference on Computer Science and Information Technology (ICCSIT’09)*, (Beijing, China), pp. 284–287, August 2009.

[70] P. E. Trahanias and A. N. Venetsanopoulos, “Color edge detection using vector order statistics,” *IEEE Transactions on Image Processing*, vol. 2, pp. 259–264, April 1993.

[71] S. Dutta and B. B. Chaudhuri, “A color edge detection algorithm in rgb color space,” in *Proc. International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom’09)*, (Kottayam, Kerala, India), pp. 337–340, October 2009.

[72] J. Gao, “Human face detection in color images,” master’s thesis, Ryerson University, Dec. 2004.

[73] K. Deshmukh and G. N. Shinde, “An adaptive color image segmentation,” *Electronic Letters on Computer Vision and Image Analysis*, vol. 5, pp. 12–23, May 2005.

[74] A. N. Evans and X. U. Liu, “A morphological gradient approach to color edge detection,” *IEEE Transactions on Image Processing*, vol. 15, pp. 1454–1463, June 2006.

[75] Y. Liu and W. A. Pearlman, “Multistage lattice vector quantization for hyperspectral image compression,” Tech. Rep. 2008-1, Rensselaer Polytechnic Institute, Troy, NY, November 2007.

[76] J. Darbon, A. Cunha, T. F. Chan, S. Osher, and G. J. Jensen, “Fast nonlocal filtering applied to electron cryomicroscopy,” in *Proc. IEEE International Symposium on Biomedical Imaging (ISBI’08)*, (Paris, France), pp. 1331–1334, May 2008.

[77] E. Christophe and W. A. Pearlman, “Three-dimensional spiht coding of volume images with random access and resolution scalability,” *Journal on Image and Video Processing*, vol. 2008, pp. 1–13, February 2008.

[78] S. B. Wesolkowski, “Color image edge detection and segmentation: A comparison of the vector angle and the euclidean distance color similarity measures,” master’s thesis, University of Waterloo, Dec. 1999.

[79] T. Gevers and A. W. Smeulders, “Color constant ratio gradients for image segmentation and similarity of texture objects,” in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’01)*, (Kauai, Hawaii, USA), pp. 18–26, December 2008.

[80] Z. Chunjiang and D. Yong, “Color image edge detection using dempster-shafer theory,” in *Proc. International Conference on Artificial Intelligence and Computational Intelligence (AICI’09)*, (Shanghai, China), pp. 476–479, November 2009.

[81] H. Madasu, O. Verma, P. Gangwar, and S. Vasikarla, "Fuzzy edge and corner detector for color images," in *Proc. 6th International Conference on Information Technology: New Generations(ITNG'09)*, (Las Vegas, NV, USA), pp. 1301–1306, April 2009.

[82] Y. A. Syed and S. M, "Color edge enhancement based fuzzy segmentation of license plates," in *Proc. 9th International Conference on Information Visualisation (IV'05)*, (London, UK), pp. 227–232, July 2005.

[83] I. Ilovici, "Detecting color edges in the visible human dataset," in *Proc. 12th IEEE Symposium on Computer-Based Medical Systems (CBMS'99)*, (Stamford, CT, USA), pp. 254–258, June 1999.

[84] J. Fan, D. K. Y. Yau, A. K. Elmagarmid, and W. G. Aref, "Automatic image segmentation by integrating color-edge extraction and seeded region growing," *IEEE Transactions on Image Processing*, vol. 10, pp. 1454–1466, October 2001.

[85] L. Xue-wei and Z. Xin-rong, "A perceptual color edge detection algorithm," in *Proc. International Conference on Computer Science and Software Engineering (CSSE'08)*, (Wuhan, China), pp. 297–300, December 2008.

[86] A. Koschan, "A comparative study on color edge detection," in *Proc. 2nd Asian Conference on Computer Vision (ACCV'95)*, (Singapore), pp. 574–578, December 1995.

[87] C. Zhou and B. W. Mel, "Cue combination and color edge detection in natural scenes," *Journal of Vision*, vol. 8, pp. 1–25, August 2008.

[88] S. Dutta and B. B. Chaudhuri, "A statistics and local homogeneity based color edge detection algorithm," in *Proc. International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom'09)*, (Kottayam, Kerala, India), pp. 546–548, October 2009.

[89] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," in *Proc. The Royal Society A: Mathematical, Physical and Engineering Sciences ('98)*, (London, UK), pp. 903–995, March 1998.

[90] S. M. Smith and J. M. Brady, “SusanŮa new approach to low level image processing,” *International Journal of Computer Vision*, vol. 23, pp. 45–78, May 1997.

[91] A. Rattarangsi and R. T. Chin, “Scale-based detection of corners of planar curves,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 430–449, April 1992.

[92] C. Schmid, R. Mohr, and C. Bauckhage, “Comparing and evaluating interest points,” in *Proc. International Conference on Computer Vision (ICCV’98)*, (Bombay, India), pp. 230–235, January 1998.

[93] F. Mokhtarian and R. Suomela, “Robust image corner detection through curvature scale space,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1376–1381, December 1998.

[94] C. Harris and M. Stephens, “A combined corner and edge detection,” in *Proc. 4th Alvey Vision Conference(AVC’88)*, (Manchester, UK), pp. 147–151, August 1988.

[95] M. Chen, Z. Cheng, and Y. Liu, “A robust algorithm of principal curve detection,” in *Proc. 17th International Conference on Pattern Recognition (ICPR’04)*, (Cambridge, UK), pp. 429–432, August 2004.

[96] K. Raghupathy, “Curve tracing and curve detection in images,” master’s thesis, Cornell University, Ithaca, Aug. 2004.

[97] P. Toft, *The Radon Transform: Theory and Implementation.* Phd. dissertation, Technical University of Denmark, Lyngby, June 1996.

[98] L. Dai, S. Han, J. Wang, and C. Rizos, “Comparison of interpolation algorithms in network-based gps techniques,” *Journal of the Institute of Navigation*, vol. 50, pp. 277–293, January 2004.

[99] Z. Tianqing, “Suspicious financial transaction detection based on empirical mode decomposition method,” in *Proc. IEEE Asia-Pacific Conference on Services Computing (APSCC’06)*, (GuangZhou, China), pp. 300–304, December 2006.

[100] X. Li, W. Wang, and R. Zhang, “Speech-stream detection with low signal-to-noise ratios based on empirical mode decomposition and fourth-order statistics,” in *Proc. 2nd Interna-*

tional Multisymposium on Computer and Computational Sciences (IMSCCS'07), (Iowa City, Iowa, USA), pp. 187–191, August 2007.

[101] S. M. A. Bhuiyan, N. O. Attoh-Okine, K. E. Barner, A. Y. Ayenu-Prah, and R. R. Adhami, "Bidimensional empirical mode decomposition using various interpolation techniques," *Advances in Adaptive Data Analysis*, vol. 1, pp. 309–338, December 2009.

[102] T. Acharya and P.-S. Tsai, "Computational foundations of image interpolation algorithms," *ACM Ubiquity*, vol. 8, pp. 1–17, January 2007.

[103] E. Lyvers and O. Mitchell, "Precision edge contrast and orientation estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 927–937, November 1988.

[104] S. M. A. Bhuiyan, R. R. Adhami, and J. F. Khan, "Fast and adaptive bidimensional empirical mode decomposition using order-statistics filter based envelope estimation," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, pp. 1–18, January 2008.

[105] Y. Kopsinis and S. McLaughlin, "Enhanced empirical mode decomposition using a novel sifting-based interpolation points detection," in *Proc. IEEE/SP 14th Workshop on Statistical Signal Processing (SSP'07)*, (Madison, Wisconsin, USA), pp. 725–729, August 2007.

[106] X. Guanlei, W. Xiaotong, and X. Xiaogang, "Neighborhood limited empirical mode decomposition and application in image processing," in *Proc. 4th International Conference on Image and Graphics (ICIG'07)*, (Chengdu, Sichuan, China), pp. 149–154, August 2007.

[107] N. ur Rehman and D. P. Mandic, "Qualitative analysis of rotational modes within three dimensional empirical mode decomposition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'09)*, (Taipei, Taiwan), pp. 3449–3452, April 2009.

[108] Y. Lu, J. Yan, and Y. Yam, "Model-based ecg denoising using empirical mode decomposition," in *Proc. IEEE International Conference on Bioinformatics and Biomedicine (BIBM'09)*, (Washington D.C., USA), pp. 191–196, November 2009.

[109] X. Gan, W. Huang, J. Yang, and B. Fu, "Internal wave packet characterization from sar images using empirical mode decomposition (emd)," in *Proc. Congress on Image and Signal Processing (CISP'08)*, (Sanya, Hainan, China), pp. 750–753, May 2008.

[110] L. Liang and Z. Ping, “An edge detection algorithm of image based on empirical mode decomposition,” in *Proc. 2nd International Symposium on Intelligent Information Technology Application (IITA’08)*, (Shanghai, China), pp. 128–132, December 2009.

[111] H. Li and Y. Zheng, “Image fusion algorithm using pyramidal empirical mode decomposition,” in *Proc. 9th International Conference on Hybrid Intelligent Systems (HIS’09)*, (Shenyang, China), pp. 152–157, May 2009.

[112] T. Chan, S. Esedoglu, and K. Ni, “Histogram based segmentation using wasserstein distances,” in *Proc. International Conference on Scale Space and Variational Methods in Computer Vision (SSVM’07*, (Ischia, Italy), pp. 697–708, June 2007.

[113] F. Shi, Q. Chen, and X. Niu, “Functional similarity analyzing of protein sequences with empirical mode decomposition,” in *Proc. 4th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD’07)*, (Haikou, Hainan, China), pp. 766–770, August 2007.

[114] Q. Li, “Flow pattern identification of two-phase flow using neural network and empirical mode decomposition,” in *Proc. 4th International Conference on Natural Computation (ICNC’08)*, (Jinan, China), pp. 375–378, October 2008.

[115] G. Tang and A. Qin, “Ecg de-noising based on empirical mode decomposition,” in *Proc. 9th International Conference for Young Computer Scientists (ICYCS’09)*, (Zhang Jia Jie, Hunan, China), pp. 903–906, November 2008.

[116] F. Wang and D. Zhao, “Chaotic oscillator detection based on empirical mode decomposition and its application,” in *Proc. International Workshop on Chaos-Fractals Theories and Applications (’09)*, (Shen Yang, Liao Ning, China), pp. 318–322, November 2009.

[117] S. Zhi-xi, H. Xi-yue, and M. Xiao-xiao, “An intelligent fault diagnosis method based on empirical mode decomposition and support vector machine,” in *Proc. 3rd International Conference on Convergence and Hybrid Information Technology (ICCIT’08)*, (Busan, Korea), pp. 865–869, November 2008.

[118] G. Rilling, P. Flandrin, and P. Goncalves, “On empirical mode decomposition and its algorithms,” in *Proc. 6th IEEE/EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP’03)*, (Grado, Italy), pp. 1–5, June 2003.

[119] W. Jian, R. Longtao, and Z. Chunhui, "Image feature extraction based on the two-dimensional empirical mode decomposition," in *Proc. Congress on Image and Signal Processing (CISP'08)*, (Sanya, Hainan, China), pp. 627–631, May 2008.

[120] S. Ai and H. Li, "Gear fault detection based on ensemble empirical mode decomposition and hilbert-huang transform," in *Proc. 5th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD'08)*, (Shandong, China), pp. 173–177, October 2008.

[121] D. Rouvre, D. Kouamé, F. Tranquart, and L. Pourcelot, "Empirical mode decomposition (emd) for multi-gate, multi-transducer ultrasound doppler fetal heart monitoring," in *Proc. 5th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT'05)*, (Athens, Greece), pp. 208–212, December 2005.

[122] A. Abhyankar and S. Schuckers, "Empirical mode decomposition liveness check in fingerprint time series captures," in *Proc. Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, (New York, New York, USA), pp. 28–35, June 2006.

[123] L. Liang and Z. Ping, "Information hiding based on empirical mode decomposition," in *Proc. IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA'08)*, (Wuhan, China), pp. 561–565, December 2008.

[124] N. E. Huang, Z. Shen, and S. R. Long, "A new view of nonlinear water waves: The hilbert spectrum," *Annual Review of Fluid Mechanics*, vol. 31, pp. 417–457, January 1999.

[125] C. Damerval, S. Meignen, and V. Perrier, "A fast algorithm for bidimensional emd," *IEEE Signal Processing Letters*, vol. 12, pp. 701–704, June 2005.

[126] J. Taghia, M. A. Doostari, and J. Taghia, "An image watermarking method based on bidimensional empirical mode decomposition," in *Proc. Congress on Image and Signal Processing (CISP'08)*, (Sanya, Hainan, China), pp. 674–678, May 2008.

[127] L. He and H. Wang, "Spatial-variant image filtering based on bidimensional empirical mode decomposition," in *Proc. 18th International Conference on Pattern Recognition (ICPR'06)*, (Hong Kong, China), pp. 1196–1199, August 2006.

[128] S. M. A. Bhuiyan, J. F. Khan, N. O. Attoh-Okine, and R. R. Adhami, "Study of bidimensional empirical mode decomposition method for various radial basis function surface in-

terpolators," in *Proc. 8th International Conference on Machine Learning and Applications (ICMLA'09)*, (Miami Beach, Florida, USA), pp. 1–7, December 2009.

[129] L. Ling, L. Ming, and L. YuMing, "Texture classification and segmentation based on bidimensional empirical mode decomposition and fractal dimension," in *Proc. 1st International Workshop on Education Technology and Computer Science (ETCS'09)*, (Wuhan, China), pp. 574–577, March 2009.

[130] Z. Liu, H. Wang, and S. Peng, "Texture classification through directional empirical mode decomposition," in *Proc. 17th International Conference on Pattern Recognition (ICPR'04)*, (Cambridge, UK), pp. 803–806, August 2004.

[131] F. L. Bookstein, "Principal warps: Thin-plate splines and the decompositions of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 567–585, June 1989.

[132] Y. Zheng and Z. Qin, "Region-based image fusion method using bidimensional empirical mode decomposition," *Journal of Electronic Imaging*, vol. 18, pp. 01300801–300810, January 2009.

[133] A. Ayenu-Prah and N. Attoh-Okine, "Evaluating pavement cracks with bidimensional empiricalmode decomposition," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, pp. 1–7, March 2008.

[134] R. B. Pachori, D. Hewson, H. Snoussi, and J. Duchene, "Postural time-series analysis using empirical mode decomposition and second-order difference plots," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'09)*, (Taipei, Taiwan), pp. 537–540, April 2009.

[135] M. A. Arafat and M. K. Hasan, "Automatic detection of ecg wave boundaries using empirical mode decomposition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'09)*, (Taipei, Taiwan), pp. 461–464, April 2009.

[136] J. tian Tang, X. li Yang, J. chao Xu, Y. Tang, Q. Zou, and X. kai Zhang, "The algorithm of r peak detection in ecg based on empirical mode decomposition," in *Proc. 4th International Conference on Natural Computation (ICNC'08)*, (Jinan, China), pp. 624–627, October 2008.

[137] D. Kim and H.-S. Oh, “Emd: A package for empirical mode decomposition and hilbert spectrum,” *The R Journal*, vol. 1, pp. 40–46, May 2009.

[138] P. Flandrin, G. Rilling, and P. Goncalves, “Empirical mode decomposition as a filter bank,” *IEEE Signal Processing Letters*, vol. 11, pp. 112–114, February 2004.

[139] S. Fang, *Image Data Compression Using Spectral Shifting*. Phd. dissertation, University of Southern Mississippi, Hattiesburg, August 2001.

[140] J. Qiao, *Fast Bernstein Function Interpolation Algorithm for Image Manipulation*. Phd. dissertation, University of Southern Mississippi, Hattiesburg, August 2004.

[141] Y. Mu, *3D Embedded Color Image Coding Using Zerotrees of Wavelet Coefficients Applied to Vissible Human Dataset*. Phd. dissertation, University of Southern Mississippi, Hattiesburg, December 2005.

[142] H. Liu, Z. Ni, and J. Li, “Time series similar pattern matching based on empirical mode decomposition,” in *Proc. 6th International Conference on Intelligent Systems Design and Applications (ISDA’06)*, (Jinan, China), pp. 644–648, October 2006.

[143] Y. Liu, *Low-Complexity Scalable Multidimensional Image Coding With Random Accessibility*. Phd. dissertation, Rensselaer Polytechnic Institute, Troy, July 2008.

[144] W. Xiong and H. Pan, “Transformer winding vibration enveloping for empirical mode decomposition based on non-uniform b-spline fitting,” in *Proc. 2nd International Symposium on Knowledge Acquisition and Modeling (KAM’09)*, (Wuhan, China), pp. 220–223, December 2009.

[145] J. Nunes, Y. Bouaoune, E. Delechelle, O. Niang, and P. Bunel, “Image analysis by bidimensional empirical mode decomposition,” *Image and Vision Computing*, vol. 21, pp. 1019–1026, May 2003.

INDEX