Image Processing Based on a Fuzzy Logic Edge Detection Algorithm

Jamshid Khazaai

Recommended Citation
https://scholarworks.wmich.edu/masters_theses/4790
IMAGE PROCESSING BASED ON A FUZZY LOGIC
EDGE DETECTION ALGORITHM

by

Jamshid Khazaai

A Thesis
Submitted to the
Faculty of The Graduate College
in partial fulfillment of the
requirements for the
Degree of Master of Science
Department of Electrical and Computer Engineering

Western Michigan University
Kalamazoo, Michigan
June 1998
ACKNOWLEDGEMENTS

A special note of gratitude is extended to my major thesis advisor, Dr. Frank Severance, for his continuous guidance and support throughout this project. I could not finish this work without his significant helps. I also wish to acknowledge my committee members Dr. Hossein Mousavinezhad and Dr. John Kapenga for their encouragement and taking the time to review my work.

Also, this task would not have been completed without the helps of my friends, Ann Schwindner and Ramin Monajemi for editing and providing me with a large number of articles and references.

Jamshid Khazaai
IMAGE PROCESSING BASED ON A FUZZY LOGIC
EDGE DETECTION ALGORITHM

Jamshid Khazaai, M.S.E.
Western Michigan University, 1998

One of the major problems in image processing is that of efficient edge detection. This is because the automated determination of an edge requires a subjective interpretation of just what the image author intended to be an edge. By training an edge determination algorithm to be sensitive to edges in a consistent manner, efficiency can be improved. This is the central idea of this thesis in which I define edge occurrences for the intensity (monochrome) images using fuzzy sets.

Edge characteristic functions are proposed for detecting edge pixels within a desired block of an (intensity and binary) image based on quadruple child windowing and binary edge patterns. I called these functions QCW-ECF. Fuzzy theory has been also applied to extend the QCW-ECF algorithm to the FQCW-ECF algorithm for edge detection within a block of a fuzzy intensity image. The (F)QCW-ECF results in a degree of edginess concerning the middle pixel of the processing block.

Cyclic coordinate algorithm is adopted to minimize the desired performance index of the Edge Detector System (EDS) by tuning the input/output membership functions of the Fuzzy Logic Controller (FLC). This is an optimization applied to the FQCW-ECF algorithm of edge detection. Finally, I have all the methods simulated against classical methods. The result are very promising.
TABLE OF CONTENTS

ACKNOWLEDGEMENTS......................................................................................................................... ii

LIST OF TABLES........................................................................................................................................ viii

LIST OF FIGURES......................................................................................................................................... ix

CHAPTER

I. INTRODUCTION........................................................................................................................................ 1

1.1 Introduction............................................................................................................................................. 1

1.2 Image Processing...................................................................................................................................... 2

1.2.1 Image Definition and Modeling.......................................................................................................... 3

1.2.2 Digital Image Processing..................................................................................................................... 3

1.2.3 Intensity Images................................................................................................................................. 4

1.2.4 Binary Images..................................................................................................................................... 6

1.2.5 Enhancing Images.............................................................................................................................. 6

1.2.6 Block Processing............................................................................................................................... 7

1.2.7 Median Filtering............................................................................................................................... 8

1.2.8 Noisy Image...................................................................................................................................... 8

1.2.9 Edge Definition................................................................................................................................. 9

1.2.10 Edge Detectors............................................................................................................................. 10

1.3 Fuzzy Theory.......................................................................................................................................... 12

1.3.1 Crisp Sets and Fuzzy Sets................................................................................................................. 12

1.3.2 Fuzzy Logic..................................................................................................................................... 13
Table of Contents-Continued

CHAPTER

1.3.3 Fuzzy Membership Function, $FMF$ ............................................... 15
1.3.4 Fuzzification ........................................................................ 18
1.3.5 Fuzzy Inference Rules ............................................................... 19
1.3.6 Fuzzy Reasoning ...................................................................... 20
1.3.7 Defuzzification ........................................................................ 21
1.3.8 Membership Function Tuning Technique................................. 23

1.4 Fuzzy Image Processing............................................................... 25
1.4.1 Fuzzy Intensity Image Definition .............................................. 25
1.4.2 Fuzzy Image Enhancement........................................................ 27
1.4.3 Fuzzy Image Smoothing Algorithms .......................................... 29
1.4.4 Edge Detection ........................................................................ 30

1.5 System Optimization................................................................... 32
1.5.1 Discrete Nonlinear System Optimization.................................... 32
1.5.2 Non-Derivative Methods ............................................................ 33
1.5.3 Sequential Line Search Algorithm............................................. 33
1.5.4 Golden Ratio Line Search Algorithm.......................................... 35
1.5.5 Multidimensional Optimization................................................ 36
1.5.6 Cyclic Coordinate Algorithm ..................................................... 38

1.6 Some Mathematical Operations .................................................. 41
1.6.1 Inner Product ........................................................................ 41
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.6.2</td>
<td>Norms</td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>1.6.3</td>
<td>Distance and Angle Definitions in Inner Product Space</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>1.6.4</td>
<td>Statistical Operation</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>1.6.5</td>
<td>Mean and Square Errors</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>1.6.6</td>
<td>Two Dimensional Correlation</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>1.6.7</td>
<td>The $t$-norm, $t$-conorm, and Aggregation Operators</td>
<td>46</td>
<td></td>
</tr>
<tr>
<td>II.</td>
<td>QCW EDGE DETECTION</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>Intensity or Gray Tone Distance</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>2.1.1</td>
<td>Analyzing Structure (Morphological Operation)</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>2.1.2</td>
<td>Presentation of Binary Edge Patterns</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>2.1.3</td>
<td>Quadruple Child Windowing</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>2.1.4</td>
<td>Child Window Processing</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>2.1.5</td>
<td>Presentation of Binary QCW-Edge Patterns</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>2.1.6</td>
<td>QCW-Edge Characteristic Function for Binary Images</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>2.1.7</td>
<td>QCW-Edge Characteristic Function for Intensity Images</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td>Correlation Between QCW-Pattern and Binary Edge Patterns</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>2.2.1</td>
<td>Distance Correlation</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>2.2.2</td>
<td>Angular Correlation</td>
<td>87</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER

2.3 Edge Detectors Based on QCW-ECF ............................... 88

III. FUZZY QCW EDGE DETECTION ........................................... 92

3.1 Fuzzy Intensity Distances .................................................. 92

3.1.1 Fuzzy Quadruple Child Windowing, FQCW ........................ 93

3.1.2 Fuzzy Child Window Processing ...................................... 94

3.1.3 FQCW Intensity Pattern ............................................... 97

3.1.4 FQCW-Edge Characteristic Function ............................ 98

3.2 Correlation of FQCW Pattern With Binary Edge Patterns .......... 105

3.2.1 Distance Correlation of FQCW Pattern.......................... 106

3.2.2 Angular Correlation of FQCW Pattern ............................ 107

3.3 Tunable Edge Detector Based on FQCW-ECF ......................... 108

IV. SIMULATION AND MODELING ............................................. 111

4.1 Optimal Edge Detector System Using Fuzzy Logic System .... 111

4.1.1 Introducing EDS Modules ......................................... 113

4.1.2 EDS Task Descriptions ............................................ 116

4.1.3 Experimental Results .............................................. 121

4.1.4 Summary of Experimental Results .............................. 151

V. SUMMARY AND CONCLUSION ............................................. 153
Table of Contents-Continued

CHAPTER

5.1 Future Research ........................................................ 154

REFERENCES ............................................................................... 156
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Operator Comparison</td>
<td>14</td>
</tr>
<tr>
<td>2.</td>
<td>Illustration of Some $t$-norms and $t$-conorm</td>
<td>47</td>
</tr>
<tr>
<td>3.</td>
<td>Illustration of Constructing the Quasi Operators Using Some $t$-(co)norms</td>
<td>48</td>
</tr>
<tr>
<td>4.</td>
<td>Illustration of Some Averaging Operators</td>
<td>49</td>
</tr>
<tr>
<td>5.</td>
<td>Illustration of Some Generalized Compensatory Operators</td>
<td>53</td>
</tr>
<tr>
<td>6.</td>
<td>Illustration of Some Symmetric Operators</td>
<td>54</td>
</tr>
<tr>
<td>7.</td>
<td>Binary Edge Pattern Codes in Block $3\times3$</td>
<td>59</td>
</tr>
<tr>
<td>8.</td>
<td>Binary $QCW$-Edge Pattern Codes</td>
<td>69</td>
</tr>
<tr>
<td>9.</td>
<td>Truth Table for the Binary $QCW$ Edge Detector</td>
<td>71</td>
</tr>
<tr>
<td>10.</td>
<td>Illustration of Some $t$-(co)norm Operators Applied for $QCW$-$ECF$</td>
<td>78</td>
</tr>
<tr>
<td>11.</td>
<td>Comparison Table for Fuzzy and Non-Fuzzy Algorithms in Example 1</td>
<td>125</td>
</tr>
<tr>
<td>12.</td>
<td>Comparison Table for Fuzzy and Non-Fuzzy Algorithms Applied for 130$\times$130 Pixels Image $II2$ in Example 2, Illustrated in Figure 48</td>
<td>133</td>
</tr>
<tr>
<td>13.</td>
<td>Comparison Table for Fuzzy and Non-Fuzzy Algorithms Applied for 130$\times$130 Noisy Image (Gussian Noise $\sigma^2=0.005$) $NI3a$</td>
<td>141</td>
</tr>
<tr>
<td>14.</td>
<td>Comparison Table for Fuzzy and Non-Fuzzy Algorithms Applied for 130$\times$130 Noisy Image (Salt &amp; Pepper Noise $d=0.02$) $NI3b$</td>
<td>145</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

1. Illustration of 8x8 Physical Image, 8x8 Image Matrix, and Intensity Levels ............................................................... 5
2. Edge Model Based on Intensity Transition ............................................................... 9
3. A Block of 3x3 Pixels ........................................................................ 11
4. Illustration of a Fuzzy Processing System .................................................... 15
5. Illustration of Triangular $MF$ ............................................................. 18
6. Illustration of the Rules for Fuzzy Logic Controller ...................................... 20
7. Fuzzy Intensity Plane ........................................................................ 27
8. Image Enhancement Using Fuzzy Theory .................................................... 28
9. Fuzzy Image Enhancement Sequence ..................................................... 29
10. Smoothing and Edge Detection Sequence ............................................. 31
11. Sequential Line Search on Direction $x$ .................................................. 34
12. Third Sequential Line Search Algorithm .................................................. 35
13. Golden Ratio Sequential Line Search Algorithm ........................................ 36
14. Sequential Line Search Illustration ......................................................... 37
15. Cyclic Coordinate Algorithm .................................................................. 38
16. Golden Ratio as a Sub-Algorithm of Cyclic Coordinate Algorithm .......... 39
17. Inter-Module Communication for the Cyclic Coordinate Algorithm ........ 41
18. Illustration of Median Operation .......................................................... 44
19. Presentation of Nine Binary Edge Patterns Within a Block of 3x3 .......... 58
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.</td>
<td>Presentation of Intensity Distances Within the Block of 3x3</td>
<td>58</td>
</tr>
<tr>
<td>21.</td>
<td>A Binary Edge Detector for 3x3 Block</td>
<td>59</td>
</tr>
<tr>
<td>22.</td>
<td>QCW Configurations Within a Block of 3x3</td>
<td>62</td>
</tr>
<tr>
<td>23.</td>
<td>QCW Configurations Within a Block of 5x5</td>
<td>64</td>
</tr>
<tr>
<td>24.</td>
<td>A Possible QCW Configuration (non-standard) Within a Block of 5x5</td>
<td>64</td>
</tr>
<tr>
<td>25.</td>
<td>Illustration of QCW Configurations</td>
<td>66</td>
</tr>
<tr>
<td>26.</td>
<td>Presentation of Binary Diagonal QCW-Edge Pattern Configurations</td>
<td>67</td>
</tr>
<tr>
<td>27.</td>
<td>Presentation of Binary Cross QCW-Edge Pattern Configurations</td>
<td>68</td>
</tr>
<tr>
<td>28.</td>
<td>Intensity Distances in QCW- Patterns</td>
<td>69</td>
</tr>
<tr>
<td>29.</td>
<td>Edge Detector Functioning Based on QCW-ECF</td>
<td>70</td>
</tr>
<tr>
<td>30.</td>
<td>Karnough Map</td>
<td>72</td>
</tr>
<tr>
<td>31.</td>
<td>Illustration of Correlation Between Vectors PBP and BEP&lt;sub&gt;p&lt;/sub&gt;</td>
<td>86</td>
</tr>
<tr>
<td>32.</td>
<td>Illustration of Correlation Between Vectors PBP and BEP&lt;sub&gt;p&lt;/sub&gt;</td>
<td>90</td>
</tr>
<tr>
<td>33.</td>
<td>Fuzzy Child Window Processing</td>
<td>95</td>
</tr>
<tr>
<td>34.</td>
<td>Illustration of FQCW Configurations</td>
<td>96</td>
</tr>
<tr>
<td>35.</td>
<td>Illustration of Edge Detection Using Fuzzy Logic Controller</td>
<td>108</td>
</tr>
<tr>
<td>36.</td>
<td>Optimal Edge Detector System (EDS) Using FLC</td>
<td>112</td>
</tr>
<tr>
<td>37.</td>
<td>Inter-Communication of Fuzzy Part of EDS</td>
<td>116</td>
</tr>
<tr>
<td>38.</td>
<td>Inter-Communication of Non-Fuzzy Part of EDS</td>
<td>118</td>
</tr>
<tr>
<td>39.</td>
<td>One Possible (F)QCW Configuration Within a 5x5 Block B</td>
<td>123</td>
</tr>
</tbody>
</table>
List of Figures-Continued

40. Fuzzy Rules Applied in *FLC* for Fuzzy Algorithms in Examples 1,2,3 ........................................... 123

41. Optimized Input/Output *MF* of *FLC* for Example 1 Obtained by *MOM* .................................................. 126

42. Illustration of Edge Trace Matrices Generated by *EDS* in Example 1 .................................................. 127

43. Illustration of Edge Trace Matrices Generated by *EDS* in Example 1 .................................................. 128

44. Illustration of Edge Intensity Surfaces for Comparison of Intensity Levels at Edge Points Between the Ideal Edge Trace and the Edge Traces Resulting From Some Algorithms in Example 1 .................................................. 129

45. Illustration of *SQE* Trajectory (*SQE* Minimization) During the Optimization of Edge Trace *IE18* Using Cyclic Coordinate Algorithm to Tune the Input/Output Membership Functions of *FLC* Illustrated in Figure 41 (a, b) .............. 130

46. Illustration of Mean Error at Each Pixel for Fuzzy and Some Non- Fuzzy Algorithms Obtained by *EDS* in Example 1 .................................................. 130

47. Optimized Input/Output Membership Functions of *FLC* for Example 2 ........................................... 134

48. Illustration of Edge of Image *II2* Generated by *EDS* in Example 2 .................................................. 135

49. Illustration of Edge of Image *II2* Generated by *EDS* for Example 2 .................................................. 136

50. Illustration of *SQE* Trajectory (*SQE* Minimization) During the Optimization of Edge of Sample Image Using Cyclic Coordinate Algorithm to Tune the Input/Output Membership Functions of *FLC* Illustrated in Figure 47 (a), (b), for Example 2 .................................................. 137

51. Illustration of Mean Error at Each Pixel for Some (Fuzzy and Non-Fuzzy) Algorithms for Edge Detection of Image *II2* in Example 2 ........................................... 137
List of Figures-Continued

52. Illustration of the Edge of a Noisy Image, $NI3a$, in Example 3a.............. 142

53. Illustration of Mean Error at Each Pixel for (Fuzzy and Non-Fuzzy) Algorithms Concerning Edge Detection of Noisy Image $NI3a$ in Example 3a, the Guassian Noise with Variance $\sigma^2 = 0.005$................. 143

54. Illustration of $SQE$ Versus $\sigma^2$ of a Noisy Image $NI3a$ for Example 3a.......................................................... 144

55. Illustration of $TDC$ Versus $\sigma^2$ of a Noisy Image $NI3a$ for Example 3a.......................................................... 144

56. Illustration of Edge of Image $NI3b$ Generated by $EDS$ in Example 3b.......................................................... 146

57. Illustration of Mean Error at Each Pixel for (Fuzzy and Non-Fuzzy) Algorithms Concerning Edge Detection of Noisy Image $NI3b$ in Example 3b, Salt & Pepper Noise with Density $d=0.02$................. 147

58. Illustration of $SQE$ Vs. Density $d$ of a Noisy Image $NI3b$ in Example 3b.................................................................. 148

59. Illustration of $TDC$ Vs. Density $d$ of a Noisy Image $NI3b$ in Example 3b.................................................................. 148
CHAPTER I

INTRODUCTION

1.1 Introduction

Edge extraction is the most significant image enhancing technique. It is a fundamental step in segmentation, object identification and motion estimation [2]. Most edge detection algorithms are implemented using block processing technique in the spatial domain by manipulation of the intensity value of each pixel and its neighborhood within a certain block of size $W \times W$ pixels. Edge detectors find edges in an intensity image by detecting pixels where the intensity values change abruptly. Using algorithms based on derivative methods, edge detection is accomplished by comparing the first derivative of the intensity to a threshold, or by checking the second derivative for a zero crossing.

Most derivative edge extraction techniques such as Sobel, Roberts, Prewitt, and Wallis methods are limited to making some non-linear manipulation of pixels over a $2 \times 2$ or $3 \times 3$ block as a means of edge enhancement before thresholding. All these mentioned methods possess the disadvantage of sensitivity to noise and dependency on the size of the block. The Roberts algorithm is sensitive to diagonal edges. The Sobel algorithm is sensitive to horizontal and vertical edges. But the edge characteristic functions based on quadruple child windowing, $QCW-ECF$, which I
introduce here, are sensitive to diagonal, vertical and horizontal edges. This follows since the generalized $QCW-ECF$ is defined according to possible edge patterns illustrated in Figures 26 and 27 on pages 67 and 68. One major problem in edge detection is finding the edge in intensity images at those pixels which have the same intensity with the background [9].

Also another problem in edge extraction is that, for many cases, the intensity edge techniques depend on the appropriate choice of thresholds. I have found that Edge detection algorithms based on fuzzy logic with tunable membership functions are an effective approach to address the above mentioned problems. The $FQCW-ECF$ which will be introduced here is an effective tool in fuzzy image processing for edge detection. Further, $QCW-ECF$ and its fuzzy extension $FQCW-ECF$ are flexible and can be applied for any desired size of block. Their sensitivities to noises are very low.

1.2 Image Processing

The term image processing generally refers to improving the visual appearance of the images or preparing the images for measurement of the features and structures present. Edge detection, which is the main part of this paper, is generally considered as a process of image feature extraction. Image processing can be generally categorized into two domains, one in frequency domain involved with the Fourier transform of images and another in the spatial domain which deals with manipulation of pixels and their neighborhoods.
1.2.1 Image Definition and Modeling

In an image processing system, the image is mathematically characterized as a deterministic or statistical model [2]. In a deterministic model, the image is functionally defined by pixel properties, such as the sum of two dimensional orthogonal functions called basis images. In a statistical model, the image is defined by statistical properties of pixels, such as mean and covariance functions.

Generally any two or higher dimensional function which carries information can be considered an image. Images can be either continuous or discrete. The continuous one is presented as a \( n \)-dimensional function \( f(X_1, X_2, \ldots, X_n) \) where \( X_n \) is an independent variable and \( n \geq 2 \). Also a discrete image which is the result of sampling of a continuous image is represented by \( f(K_1, K_2, \ldots, K_m) \) where \( (K_1, K_2, \ldots, K_m) \) refers to the coordinates of a physical image pixel.

1.2.2 Digital Image Processing

Following the given definition of an image as a discrete \( n \)-dimensional function, digital image processing refers to processing of any 2-dimensional data. A digital image \( I \) of \( M \times N \) dimension can be considered as an array of real or complex numbers represented by a finite number of bits [2].

In digital image processing systems, we usually deal with arrays of numbers obtained by spatially sampling points of a physical image. After processing, another array of numbers is produced and these numbers are then used to reconstruct a
continuous image for viewing. Thus, digital image processing is the sequence of digitizing (sampling and quantizing) and processing of digital data.

1.2.3 Intensity Images

A digital gray tone (monochrome) image $II$ can be generally represented by a 2-dimentional intensity function $f(m,n)$ where $(m,n)$ are the coordinates of an image pixel and $f(m,n)$ is the brightness value. A digital 2-dimensional image in spatial domain can be shown as a single $M \times N$ dimension matrix with $L$ gray tone levels as

$$II = f(m,n) = [X_{mn}] \quad \text{where} \quad \begin{cases} X_{mn} \in [0,1] \\ l \in \{1,2,...,(L-1)\} \end{cases} \quad \text{for} \quad \begin{cases} m = 1,2,...,M \\ n = 1,2,...,N \end{cases} \quad (1.2.3-1)$$

Intensity image is represented by a single matrix of size $M \times N$ containing double precision values within continuum interval $[0,1]$ and a crisp gray level $l$ ranging from 0 to $L-1$, with $L$ discrete gray levels. Each element of this matrix contains the information of one image pixel. The gray level of intensity 0 refers to black and $L-1$ refers to white, the highest intensity. A black and white photograph is the best example of intensity images.

The following $8 \times 8$ matrix $II$ generated by the MATLAB image processing toolbox represents an intensity image $II = f(m,n)$ with $L=16$ gray levels. The normalized gray scale is considered as an intensity scale which is over the interval $[0,1]$. The intensity value 0 indicates the black (the gray level $0^{th}$) and 1 corresponds to white (the gray level $(L-1)^{th}$). Consequently any intensity value within $[0,1]$
corresponds to the gray level \( l \in \{0,1,...,L-1\}\). Figure 1 represents an physical intensity image, a ring. The relevant \( 8 \times 8 \) matrix is processed in Chapter IV. This simple matrix which represents an intensity image of a ring is used as example of simulation further.

\[
\begin{array}{cccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0.28 & 0.28 & 0.28 & 1 & 1 \\
1 & 1 & 0.28 & 0.7 & 0.7 & 0.28 & 1 & 1 \\
1 & 1 & 0.28 & 0.7 & 0.7 & 0.28 & 1 & 1 \\
1 & 1 & 0.28 & 0.28 & 0.28 & 0.28 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}
\]

\[
\begin{align*}
I_l &= f(m,n) = [X_{mn}] \\
M &= 8, \quad m = 1,...,8 \\
N &= 8, \quad n = 1,...,8 \\
L &= 16 \\
X_{mn} &\in [0,1] \\
l &\in \{0,1,...,15\}
\end{align*}
\]

Figure 1. Illustrations of \(8\times8\) Physical Image, \(8\times8\) Image Matrix, and Intensity Levels.
1.2.4 Binary Images

A binary image can be considered as a special case of an intensity image. A binary image is a matrix $M \times N$ dimension with two levels, 0 (black) and 1 (white). Logical or Boolean matrices can be considered as binary images. We will see later that edge extraction from intensity image using threshold results in a binary image.

1.2.5 Enhancing Images

Image enhancement is the technique of processing an image to prepare it for further analysis or for monitoring [1]. Therefore, based on a specific application, some certain image features are accentuated for subsequent processing. Contrast stretching, smoothing and sharpening, edge extraction, noise filtering and magnifying are important image enhancing techniques.

**Contrast stretching** changes the intensity level of an image by mapping the gray level into another level using a single-valued transformation in order to increase the total dynamic range of gray levels. This, in turn, increases contrast.

**Image smoothing** is generally applied to reconstruct a noisy image. In the spatial domain, this is achieved by intensity modification of each pixel. In the frequency domain it is accomplished by applying a low pass filter, LPF. In this process each pixel and its neighborhood posses almost equal intensity.

Defocusing, Averaging (which is a LPF) and Max–Min rule algorithms are some techniques applied for the smoothing operation [1]. For example, the averaging method applied to remove the noise is based on averaging the neighboring intensities
of each pixel. If \( X_{mn} \) is the processing pixel, then the smoothed pixel \( X_{mn}' \) is computed based on \( N \) number of intensity values of neighborhoods within block \( B \) as

\[
X_{mn}' = \frac{1}{N} \sum_{B} X_{ij} \quad \text{where} \quad (i,j) \in B - \{(m,n)\}
\]

(1.2.5-1)

1.2.6 Block Processing

Most algorithms of digital image processing in the spatial domain deal with modifying of an image matrix based on local algorithm. In other words, most algorithms in the spatial domain manipulate the intensity of each pixel by considering the intensities of the pixels only in immediate neighborhood. Since processing each of the \( M \times N \) pixels serially is time consuming, block processing is desirable. Block processing is a suitable technique by which the image matrix is divided into some smaller matrix dimensions for processing, which can be done in parallel.

Block processing can be achieved by rectangular partitioning of whole matrix into a group of smaller matrices of \( W \times W \) dimension with no component overlap or by sliding a rectangular window \( W \times W \) dimension from left to right and top to bottom of an image pixel by pixel. In sliding block processing, the center pixel of the block is processed considering neighboring pixels based on an algorithm and finally replaced by the result of processing. The process is repeated pixel by pixel with its own proper \( W \times W \) block until the whole image is covered.
1.2.7 Median Filtering

Median filter is a linear operation on neighboring pixels of the middle pixel in block \( B \). Input of the filter is the intensity of the middle pixel and output is the computed intensity value from neighborhood \( B \).

\[
X_{mn} = \text{median}_{i,j} \left\{ X_{ij} \right\} \quad \text{where} \quad (i, j) \in B - \{(m, n)\} \quad (1.2.7-1)
\]

Median filtering orders the pixel values in a neighborhood and chooses the median value as the result. In this process the center pixel in the block is replaced by the median of the intensity values of neighborhoods within the block \( B \) with \( W \times W \) dimension.

1.2.8 Noisy Image

Due to physical realities involved in image processing systems, there is always an additional unwanted signal, noise. For example [1], a general mathematical model for an Electro-Optical system can be expressed as

\[
V(x,y) = g[w(x,y)] + n(x,y) \quad \text{where} \quad w(x,y)=h(u(x,y)) \quad (1.2.8-1)
\]

\( V(x,y) \) represents the observed image, \( h \) is a linear operation on the original image \( u(x,y) \) and \( g \) is a non-linear operation on \( w \). The general model of additive noise denoted by \( n(x,y) \) has application in many situations and defined as

\[
n(x,y) = f [g[w(x,y)]] \cdot n_1 \ (x,y) + n_2 \ (x,y) \quad (1.2.8-2)
\]
Functions $f$ and $g$ are non-linear operation of decoder/recording mechanisms. First term of $n(x,y)$ is the image-dependent random noise and second term is independent random noise. Noise pixels distributed randomly over the image are the result of errors in image transmission and noisy components of the electronic systems. Two important types of noise are Gaussian noise with zero mean and outliers, extreme pixel intensity values. Median filtering is useful for removing outliers and Wiener filtering works best when the noise is a constant power additive noise such as Gaussian white noise.

### 1.2.9 Edge Definition

A pixel at which the intensity or gray level value changes abruptly is considered an edge pixel. Figure 2 illustrates one-dimensional edge model.

![1-dimensional edge model](image)

**Figure 2.** Edge Model Based on Intensity Transition.

Considering direction $i$ of one dimensional edges, edge at point $m$ on this
direction can be modeled based on one pixel wide-transition in intensity from $X_{i,m-1}$ to $X_{i,m+1}$. Above figure shows edge model as a ramp increase in intensity level. Edge is characterized by its height $h$, the slope angle $\alpha$ and $m$ coordinate of the slope midpoint $X_{i,m}$.

According to this model we can say an edge exists if both the slope angle $\alpha$ and the height $h$ are greater than specified values (thresholds). Most derivative algorithms for edge detection are based on checking the approximation of first for exceeding the threshold or the second derivative for zero crossing.

1.2.10 Edge Detectors

Edge detectors find edges in an intensity image by detecting pixels where the intensity values change abruptly. Using certain algorithms such as derivative methods, edge detection is done by comparing the first derivative of the intensity to certain thresholds, or by checking the second derivative for a zero crossing. Roberts, Sobel, Wallis, Kirsch, Marr-Hildreth and Prewitts are some possible derivative methods of non-linear edge enhancement. Most techniques are limited to making some non-linear manipulation of pixels over a $2 \times 2$ or $3 \times 3$ blocks illustrated in Figure 3 as a means of edge enhancement before thresholding [13].

The **Roberts method** is a simple nonlinear cross operation on a window $2 \times 2$ as a differencing method for edge sharpening and edge isolation.

\[
E_R(m,n) = ( |f(m,n) - A_d|^2 - |A_3 - A_5|^2 )^{0.5} \quad (1.2.10-1)
\]
The Roberts approximation to derivative, $E_R(m,n)$, detects edges at those pixels where the gradient of $f$ is maximum and is also sensitive to diagonal edges.

The Sobel method is a non-linear operation on a $3 \times 3$ block described by pixel intensities in neighborhoods.

$$E_S(m,n) = (X + Y)^2$$ (1.2.10-2)

Where

$$X = (A_2 + 2. A_3 + A_4) - (A_0 + 2. A_7 + A_6)$$ (1.2.10-3)

$$Y = (A_0 + 2. A_1 + A_2) - (A_6 + 2. A_5 + A_4)$$ (1.2.10-4)

The Sobel approximation to derivative, $E_S(m,n)$, also detects edges at those pixels where the gradient of $f$ is maximum and is sensitive to horizontal and vertical edges.

The Wallis method is based on the logarithm of image luminance at the center pixel and neighborhoods over a $3 \times 3$ block. According to this non-linear method an edge exists if the magnitude of the logarithm of intensity value at the center pixel exceeds the magnitude of average logarithmic intensity of its four nearest neighbors by a fixed threshold value.
1.3 Fuzzy Theory

In 1965 Zadeh suggested a fuzzy set theory in which each element of a set has a degree of membership (degree of belonging to the set) which continuously ranged over \([0,1]\), rather than being either 0 or 1. He developed all set operations such as union, intersection and complement over a fuzzy set. Today fuzzy sets and their associated fuzzy logic are widely used in many scientific and commercial problem domains.

1.3.1 Crisp Sets and Fuzzy Sets

Considering a universal set \(M\), an ordinary crisp set \(A\) is defined by identifying those elements of \(M\) which set \(A\) contains. The characteristic function \(P_A(x)\) of set \(A\) is defined so that each element of the set \(M\) has a corresponding value 1 or 0. The element \(x_i\) of \(M\) belongs to set \(A\) if and only if \(P_A(x_i)=1\). Said more formally, \(A=\{x \mid P_A(x)=1\}\). Fuzzy sets are based on the idea of extending the range of the characteristic function from the binary values 0 or 1 to the continuous real interval values \([0,1]\). Therefore, a fuzzy set \(A\) is defined by identifying its components \(x\) and their associated membership degrees \(\mu_A(x)\). The membership function of fuzzy set \(A\) is defined to give each element of set \(A\) a corresponding real value within interval \([0,1]\). The mathematical notation of fuzzy set \(A\) is represented as

\[
E_w(m,n) = \frac{1}{4} \cdot \log \left[ \frac{(f(m,n))^4}{A_1 \cdot A_3 \cdot A_5 \cdot A_7} \right] \tag{1.2.10 - 5}
\]
\[ A = \{(x, \mu_A(x)) \mid \mu_A \in [0,1], \quad x \in M\} \quad (1.3.1-1) \]

When the universal set \( M \) is discrete fuzzy set \( A \) is represented by Equation (1.3.1-1) where the \( \sum \) referring to union operator in fuzzy set.

\[ A = \sum_{i=1}^{N} \left( \frac{\mu_{x_i}}{x_i} \right) \quad (1.3.1-1) \]

### 1.3.2 Fuzzy Logic

Logic is the study of the methods and principles of reasoning in all its possible forms [13]. Classical logic deals with propositions that yield values over set \( \{0,1\} \), false or true. Boolean logic or two-valued (binary) logic has been used in traditional logic and set theory to model the world as black and white, false and true, over set \( T_2 = \{0,1\} \). Two-valued logic is suitable for systems which are modeled crisply. It has the advantage of being crisp in inferences but the disadvantage of not accurately describing the analog real world.

Multi-valued logic was developed by Lukasiewicz in the 1930s. The set \( T_N \) of truth values for integer \( N \) is assumed to be evenly divided over the closed interval \( [0,1] \). That is, \( N \)-valued logic can be extended to the case where \( N \) is countable, and the truth values are allowed correspond to the points on the interval \( [0,1] \). This case is known as Lukaiswics \( L_1 \) logic, and provides the base logic needed for working with fuzzy sets [4].
The term fuzzy logic usually involves the manipulation of truth values such as 'nearly true' defined as fuzzy sets over the interval [0,1] of $L_1$ truth values. In fuzzy logic, usually the knowledge is linguistically expressed as rules in the form of 'If $x$ is $A$, then $y$ is $B$' where $x$ and $y$ are fuzzy variables and $A$ and $B$ are fuzzy sets or fuzzy values. Fuzzy logic deals with propositions that possess values over interval [0,1] and also combinations of variables that stand for arbitrary propositions. These variables called fuzzy linguistic variables. The combination of truth values of these fuzzy variables is done by multi-valued logic operators such as complement, max, min operators. Table 1 illustrates the main operators used in crisp set and $N$-valued logic.

Table 1

<table>
<thead>
<tr>
<th>Crisp set</th>
<th>Two-valued logic</th>
<th>N-valued logic</th>
<th>Fuzzy logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A'$</td>
<td>$1 - A$</td>
<td>$1 - A$</td>
<td>$1 - A$</td>
</tr>
<tr>
<td>$\cup$</td>
<td>$\lor, \vee$</td>
<td>Max</td>
<td>Max, t-conorm</td>
</tr>
<tr>
<td>$\cap$</td>
<td>$\land, \wedge$</td>
<td>Min</td>
<td>Min, t-norm</td>
</tr>
</tbody>
</table>

Fuzzy logic processing includes fuzzification of crisp sets, decision making logic (fuzzy inferencing), knowledge base (IF–THEN rules) and defuzzification. A simple architecture of a fuzzy logic system (FLC) is illustrated in Figure 4. This system, which is applied in image processing, will be described in detail later.
1.3.3 Fuzzy Membership Function, FMF

A MF is a curve that defines how each member $x$ in fuzzy set $A$ is mapped to a membership value $\mu_A(x)$ (or degree of membership) over an interval $[0,1]$. We mathematically represent this as

$$\mu_A(x) : x \xrightarrow{\mu_A} [0,1] \quad (1.3.3 - 1)$$

Membership function (MF) can be generated heuristically or generated by using clustering techniques, such as $c$-mean, adaptive vector quantization AVQ, and the self organizing map SOM. In this paper we simply apply a heuristic approach to generate membership function. Choosing the appropriate MF is application dependent. The most commonly used MFs are the triangular, trapezoidal, S-shaped, Z-shaped, bell-shaped, Sigmoid and Gaussian. Following are the examples of MFs.

**Triangular shape MF** can be generated functionally using Equation (1.3.3-2). Parameters $a, b, c$ are the parameters by which to tune the MF as desired to make a symmetric or anti-symmetric MF.
\[
\Lambda(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) = \begin{cases} 
0 & \text{if } x \leq a \\
\frac{x-a}{b-a} & \text{if } a \leq x \leq b \\
\frac{c-x}{c-b} & \text{if } b \leq x \leq c \\
0 & \text{if } x \geq c 
\end{cases} \tag{1.3.3-2}
\]

**Trapezoid shape MF** can be generated functionally using Equation (1.3.3-3).

Parameters \(a, b, c, d\) are also the tuning parameters by which to make a symmetric or anti-symmetric MF.

\[
\Pi(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) = \begin{cases} 
0 & \text{if } x \leq a \\
\frac{x-a}{b-a} & \text{if } a \leq x \leq b \\
1 & \text{if } b \leq x \leq c \\
\frac{d-x}{d-c} & \text{if } c \leq x \leq d \\
0 & \text{if } d \leq x 
\end{cases} \tag{1.3.3-3}
\]

**S-shaped MF**, the Zadeh's MF, is defined as Equation (1.3.3-4). Parameters \(a, b, c\) are also the tuning parameters for this MF.

\[
S(x; a, b, c) = \begin{cases} 
0 & \text{if } x \leq a \\
2\left(\frac{x-a}{b-a}\right)^2 & \text{if } a < x \leq b \\
1 - 2\left(\frac{x-c}{x-a}\right)^2 & \text{if } b < x \leq c \\
0 & \text{if } c < x 
\end{cases} \tag{1.3.3-4}
\]
Bell-shaped $MF$ is defined by Equation (1.3.3-5). Parameters $a$, $b$, $c$ can be used to tune this $MF$ as desired to make a symmetric or anti-symmetric $MF$.

$$
\Pi(x; a, b, c) = \begin{cases} 
S\left(x; c - b, c - \frac{b}{2}, c\right) & \text{if } x \leq c \\
1 - S\left(x; c + \frac{b}{2}, c + b, c\right) & \text{if } x > c 
\end{cases}
$$

(1.3.3-5)

The generalized bell $MF$ is represented in Equation (1.3.3-6). The parameters $a$, $b$, $c$ are used to tune $MF$ centered at point $c$ over the $x$-axis.

$$
\Pi(x; a, b, c) = \frac{1}{1 + \left|\frac{x - c}{2a}\right|^{2b}} 
$$

(1.3.3-6)

Consider the case of triangular $MF$. The most significant points in $MF$ are the left and right points, peak point and the cross-point level. The peak point is the point $X_{peak}$ for which the value of the $MF$ is equal to 1. Left and right points are respectively the least and the greatest value of fuzzy set $A$ in which the value of $MF$ is equal to 0. The peak value is an interval for membership function. Now we can define the left and right width based on the above mentioned points. The left width, $W_{left}$ is the length of the interval $[X_{left}, X_{peak}]$ that is $W_{left} = X_{peak} - X_{left}$. Similarly the right width, $W_{right}$ is the length of the interval $[X_{peak}, X_{right}]$ that is $W_{right} = X_{right} - X_{peak}$.

Figure 5 illustrates the triangular $MF$ and its corresponding parameters.

In case of overlap between two $MF$s $A$ and $B$, the cross-point $X_{cross}$ and the cross point level are parameters which are useful in defining the percentage of
overlap. A cross-point $X_{cross}$ is the point at which both membership functions have the same value (level). Cross-point level, which is the same on both $MF$, is the membership degree of $X_{cross}$. Two $MF$s $A$ and $B$ may have more than one cross-point. Now we can define percentage of overlap of $A$ in $B$ as a ratio of the length of interval $[X_{cross}, X_{right}]$ and $W_{right}$ in membership function $A$ that is

$$P_{ovl,A} = \frac{X_{right,A} - X_{cross}}{X_{right,A} - X_{peak}} = \frac{X_{right,A} - X_{cross}}{W_{right,A}}$$

(1.3.3-7)

And consequently the percentage of overlap of $B$ in $A$ can be defined as

$$P_{ovl,B} = \frac{X_{cross} - X_{left,B}}{X_{peak} - X_{left,B}} = \frac{X_{cross} - X_{left,B}}{W_{left,B}}$$

(1.3.3-8)

Figure 5 represents the triangular membership functions $A$ and $B$ and their corresponding parameters.

Figure 5. Illustration of Triangular $MF$.

### 1.3.4 Fuzzification

Fuzzification is the process of producing a fuzzy set from a crisp set as illustrated by Figure 4. Using the fuzzy logic system to process any data, we have to
first convert the crisp data to the fuzzy data in order to apply fuzzy operators for any computation in rule-based inferencing stage. Fuzzification is done by associating the membership degree for inputs to the fuzzy logic system using certain membership functions. Fuzzification requires (a) identifying relevant input and output variables of the fuzzy logic system and ranges of their values, (b) selecting meaningful linguistic states for each variables and expressing them by appropriate fuzzy sets and (c) defining membership functions for each input variable to express the associated membership degrees.

In this paper (Chapter IV) inputs to the fuzzy system range over gray levels scaled from 0 (Black) to 255 (White). We have applied fuzzification methods by first partitioning the intensity scale [0,1] (or the discrete gray scale 0-255) into 5 individual ranges, and second by defining linguistic variables Dark, Dark Gray, Bright Gray and Bright for these 5 ranges. Finally we have applied the fuzzy quantization over the range [0,1] using 5 membership functions. We have used two trapezoid-shaped MFs for left and right ranges, and three triangular-shaped MFs for the other three ranges. You will see further, Figure 7 on page 27 represents the fuzzy plane applied for fuzzification over intensity scale [0,1] or discrete gray scale 0-255.

1.3.5 Fuzzy Inference Rules

Fuzzy rules, which are often expressed in the form of IF-THEN statement, are essentially fuzzy relations or fuzzy implications [4]. In this step, the knowledge pertaining to the given problem is expressed in terms of fuzzy inference rules and
stored in fuzzy rule base. Figure 6 represents a rule base containing the 25 fuzzy rules applied in a fuzzy control system. Then the rule $25^{th}$ is expressed as follows.

Rule $25^{th}$: IF ($X_{mn}$ is $G$) and ($Y$ is $G$), THEN ($d$ is $Z$)

Figure 6. Illustration of the Rules for Fuzzy Logic Controller.

The module consisting a number of fuzzy rules called fuzzy knowledge-base or fuzzy rule-base. In fuzzy knowledge-base all the possible rules can conveniently be represented in a matrix form.

1.3.6 Fuzzy Reasoning

Fuzzified input variables of fuzzy logic system are taken to fuzzy inference machine to be processed by relevant fuzzy information rules to make inferences regarding the output variables. This process is called fuzzy reasoning. There are various techniques to combine inputs with relevant rule in order to infer an appropriate fuzzy output. The most commonly applied are Max-Min and Max-Dot fuzzy inference methods. Assume we have $n$ rules in knowledge base as

Rule 1: IF ($x$ is $A_1$ and $y$ is $B_1$), THEN $z$ is $Z_1$  
.................................................................
Rule \( n \): IF \((x \text{ is } A_n \text{ and } y \text{ is } B_n)\), THEN \( z \text{ is } Z_n \)

For two fuzzy input variables \(x_0\) and \(y_0\), assume only two rules 2 and 4 are active as

Rule 2: IF \((x_0 \text{ is } A_2 \text{ and } y_0 \text{ is } B_2)\), THEN \(z\) is \(C_2\)

Rule 4: IF \((x_0 \text{ is } A_4 \text{ and } y_0 \text{ is } B_4)\), THEN \(z\) is \(C_4\)

Then, The corresponding fire strength parameters for rules 2 and 4 are respectively defined as

\[
\alpha_2 = \min \left[ \mu_{A_2}(x_0), \mu_{B_2}(y_0) \right] \quad \text{and} \quad \alpha_4 = \min \left[ \mu_{A_4}(x_0), \mu_{B_4}(y_0) \right] \quad (1.3.6-1)
\]

Then for Max-Min reasoning, Mamadani’s minimum operation is used for fuzzy implication and the output membership function for consequent \(C\) can be inferred as

\[
\mu_C(z) = \max \left[ \min (\alpha_2, \mu_{C_2}(z)), \min (\alpha_4, \mu_{C_4}(z)) \right] \quad (1.3.6-2)
\]

In Max-Dot reasoning, Larsen’s product operation is used as fuzzy implication function [4]. The membership degree of the inferred consequence is obtained as

\[
\mu_C(z) = \max \left[ (\alpha_2 \cdot \mu_{C_2}(z)), (\alpha_4 \cdot \mu_{C_4}(z)) \right] \quad (1.3.6-3)
\]

Note that in both methods, if \(x_0\) and \(y_0\) are crisp values then the above mentioned operations result in fuzzy singleton.

1.3.7 Defuzzification

The conclusion resulting from a certain fuzzy reasoning technique in fuzzy
inference machine is expressed in terms of fuzzy set. This must be converted to a real crisp value. This process of converting a fuzzy set to a single real number is called defuzzification. There are some defuzzification methods each of which leads to a different result. Assume the fuzzy set resulting from a fuzzy inference machine is a discrete fuzzy set \( C \) as

\[
C = \sum_{k=1}^{n} \frac{\mu_k}{z_k}, \quad \text{where} \quad \mu_k \in [0,1] \tag{1.3.7 - 1}
\]

The most commonly used techniques are center of area \( COA \), center of maxima \( COM \), and mean of maxima \( MOM \). These are defined as follows

**Center Of Maxima, \( COM \)**, Consider fuzzy set \( M \) given by Equation (1.3.7-2), where \( h(c) \) is the height of \( C \) (maximum membership degree in \( C \)).

\[
M = \left\{ z_i \mid \mu_i = h(c) \right\} \tag{1.3.7 - 2}
\]

\( COM \) is defined as the average of the smallest value and the largest value of \( C \) with the same membership degree equal to \( h(c) \). This is represented as

\[
d_{com}(c) = \frac{\min\{z_i \mid z_i \in M\} + \max\{z_i \mid z_i \in M\}}{2} \tag{1.3.7 - 3}
\]

**Mean Of Maxima, \( MOM \)**, method which is the average of all members of above defined set \( M \), is defined as

\[
d_{mom}(c) = \frac{\sum_{z_i \in M} z_i}{|M|} \tag{1.3.7 - 4}
\]
Center Of Area, COA, method indicates the value within the range of $z_i$ in which the area under membership function $C$ is partitioned into two equal sub-areas. This is represented as

$$d_{com}(c) = \frac{\sum_{i=1}^{n} (\mu_i \cdot z_i)}{\sum_{i=1}^{n} \mu_i} \quad (1.3.7-5)$$

The results obtained from the above methods are application dependent. For example in our approach to edge detection which is explained further in Chapter IV, the concluded fuzzy set from the inference machine is

$$D = \left\{ \mu_{/d_1}^/, \mu_{/d_2}^/, \mu_{/d_3}^/, \mu_{/d_4}^/ \right\} \quad (1.3.7-6)$$

While the COA technique gives a better result than the MOM and COM techniques, using one of the edge characteristic functions (ECF) given in section (3.1.4) results in a better value as a defuzzification technique.

$$d_{ecf}(D) = E_{qcw}(D) = \frac{\left\{ (\mu_1 \cdot d_1 + \mu_4 \cdot d_4) \cdot (\mu_2 \cdot d_2 + \mu_3 \cdot d_3) \right\}}{4} \quad (1.3.7-7)$$

1.3.8 Membership Function Tuning Technique

As we will see, tuning of membership functions is important to optimize parameters. By reforming the membership function both the conclusion obtained from the fuzzy inference machine and, consequently the value resulting from the
defuzzification stage, change considerably.

The shape of each MF depends on parameters $a$, $b$, $c$, $d$ involved in constructing the MF. Thus, the variation of parameters $a$, $b$, $c$, $d$ in membership functions has an effects on the resulting output of fuzzy logic system. We take an advantage of this effect to tune MF in order to obtain the desired result in output. The gradient decent and neural network algorithms are two of the most commonly used approaches to tuning membership functions. In this paper we use the multidimensional vector optimization technique called the cyclic coordinate algorithm. This technique is used to tune membership functions and consequently to optimize the defuzzification parameters simultaneously.

Assume we have triangular-shaped MF defined by Equation (1.3.3-2) with $W_{left}$ and $W_{right}$. Then the variable vector used for tuning MF is a 3-dimensional vector as $V= [W_{left}, b, W_{right}]$. In our approach to optimal edge detection using fuzzy logic system, we have applied the percentage of overlap $P_{ovl}$ (the same one for all membership functions), as well as peak points of membership functions, as the variables to tune the input/output MFs of fuzzy logic controller (FLC). This FLC has been designed to lead the edge detector system (EDS) toward an optimal output using cyclic coordinate algorithm. You will see further in Figure 7 on page 27, there are five MFs assigned for five input linguistic variables to the fuzzy logic system. The variable vector applied for tuning input MF has been chosen as $V_{in}= [P_{ovl}, b_1, b_2, b_3, b_4, b_5]$. If $|X_{cross} - X_{right}| = |X_{left} - X_{cross}|$, then $(P_{ovl})_A = (P_{ovl})_B$. The cyclic coordinate algorithm is explained in section 1.5 in detail.
In Chapter IV, the variable vector pertaining to the optimal fuzzy edge
detection problem consists of the overlap percentages and peak points belonging to
input/output membership functions. This variable vector is used to tune membership
functions to obtain the optimal edge based on minimizing the error $e$ between the
desired edge trace and the actual resulting edge trace from EDS as the objective
function.

1.4 Fuzzy Image Processing

Fuzzy logic has been applied to gray tone image processing such as gray level
thresholding, edge detection, image enhancement, segmentation and so on. Several
fuzzy operators have been used in fuzzy image processing which include $max$, $min$,
$INT$ [1]. Further, as we see, the use of fuzzy logic system (controller) is a significant
choice in optimization of image processing. A gray tone image possesses some
ambiguity within the pixels due to the pixel multi-valued levels of brightness.
Therefore it is justifiable to apply the logic of fuzzy set rather than ordinary set to
image processing problems.

1.4.1 Fuzzy Intensity Image Definition

In the spatial domain, a fuzzy intensity image $FII$ of dimension $M\times N$ with $L$
gray tone levels is an array of fuzzy singletons that can mathematically be
represented as Equation (1.4.1-1), for $m=1,2,...,M$ and $n=1,2,...,N$. $X_{mn}$ denotes the
intensity value of pixel $(m,n)^{th}$ and $\mu_{mn}$ is its associated membership degree.
Similarly, the definition of an intensity image given by Equation (1.2.3-1), the normalized gray scale is also considered here as an intensity scale over the interval [0,1]. That is, \( X_{mn} \in [0,1] \). Depending on the membership function(s) used for fuzzification of digital image we define the fuzzy plane as

\[
\mu_{mn} = P(X_{mn}) \quad (1.4.1-2)
\]

\( P \) refers to membership function(s) defined over the intensity value range [0,1] (or gray level scale 0 to \( L-1 \)). \( FX_{mn} \) is a fuzzy singleton denoting the fuzzified intensity image pixel \( (m,n) \). \( X_{mn} \) is the intensity value (or gray level \( l \)) at this pixel and \( P(X_{mn}) \) defined by Equation (1.4.1-2) is the corresponding membership degree of having grayness relating to some gray level \( l \). Membership function(s) applied in the fuzzy plane \( P \) can be selected as one of the functions explained in section 1.3.3. The membership degree \( \mu_{mn} \) takes a value between interval [0,1].

In Chapter IV for fuzzy approach to edge detection, we apply triangular and trapezoidal membership functions, defined by Equations (1.3.3-2) and (1.3.3-3) in the fuzzy plane illustrated in Figure 7 to fuzzify the digital image array. As this Figure shows, the fuzzy plane consists of three triangular membership functions concerning linguistic variables \( DG, G, BG \) and two trapezoidal membership functions relevant to...
linguistic variables \( D, B \). These five membership functions quantize the gray level 0-255 within interval \([0,1]\) as fuzzification. The positive constants \( a, b, c, d \) which are the tools for tuning membership functions, consequently have the effects of altering the ambiguity in the fuzzy plane by changing the cross-over point and slope of transformation function.

![Fuzzy intensity plane](image)

Figure 7. Fuzzy Intensity Plane.

1.4.2 Fuzzy Image Enhancement

As discussed in section 1.2.5, the purpose of enhancing an image is the modification of that image in such a way that the result is more suitable than original for a specific application. This modification is implemented by some manipulations
on pixel intensities in the fuzzy property domain. Figure 8 illustrates a block of image enhancement. The fuzzy operator \( INT \), which is actually a contrast intensification function, is taken as an image enhancing tool. This function is defined as

\[
T_1 = INT(\mu_{mn}) = \begin{cases} 
2 \cdot \mu_{mn}^2 & \text{for } 0 \leq \mu_{mn} \leq 0.5 \\
1 - 2 \cdot (1 - \mu_{mn})^2 & \text{for } 0.5 \leq \mu_{mn} \leq 1
\end{cases} \tag{1.4.2-1}
\]

This function enhances the membership value of those elements whose membership is above 0.5 and diminishes that of those elements with membership below 0.5. This function is recursively applied for better contrast enhancement as

\[
T_s = INT(T_{s-1}) \tag{1.4.2-2}
\]

The subscript \( s \) represents the number of composites of \( INT \) function [1]. Assume \( P(X_{mn}) \) defines a fuzzy plane such as Figure 7. The block diagram illustrated in Figure 8 is considered as a block of image enhancement using intensification function in the fuzzy property domain. The parameters \( a, b, c, d \) are fuzzifiers and defuzzifiers and \( s \) in function \( T_s \) is used as an intensity tuner.

![Figure 8. Image Enhancement Using Fuzzy Theory.](image-url)
1.4.3 Fuzzy Image Smoothing Algorithms

Image smoothing, which is classified as an image enhancing algorithm, is achieved by intensity modification of pixels in the spatial domain. Each pixel and its neighborhood possess equal brightness or gray level. The technique used for fuzzy image enhancement [1] is presented in Figure 9.

Figure 9. Fuzzy Image Enhancement Sequence.

The sequence consists of pre-enhancement of an image using the method described in section 1.4.2, block $E$ followed by the smoothing process through block $S$ and a post-enhancement of the smoothed image through block $E$ again. All smoothing processes blur the images by removing the abrupt changes in intensity or gray level and consequently smooth the edges. Therefore, the post-enhancement is used to improve the intensity contrast in order to obtain sharp images. There are some methods to smooth an image which can be applied as tools in block $S$.

Defocusing technique is used as a smoothing algorithm using a linear non-recursive filter. It is defined as

$$ Y_{mn} = a_0 \cdot Y_{mn} + a_1 \cdot \sum_{Q_1} Y_{ij} + a_2 \cdot \sum_{Q_2} Y_{ij} + \cdots + a_s \cdot \sum_{Q_s} Y_{ij} $$  \hspace{1cm} (1.4.3-1)

where

$$ \begin{cases} a_0 + N_1 \cdot a_1 + N_2 \cdot a_2 + \cdots + N_s \cdot a_s = 1 \\ 0 \leq a_1 \leq a_2 \leq \cdots \leq a_s \leq 1 \\ (i,j) \neq (m,n) , \quad m=1,2,\cdots M , \quad n=1,2,\cdots N \end{cases} $$

$$ Y_{mn} ' = X_{mn} \quad Y_{mn} \quad Y_{mn} ' \quad X_{mn} ' $$

Figure 9. Fuzzy Image Enhancement Sequence.
$Y_{mn}$ represents the $(m,n)^{th}$ pixel intensity of the pre-enhanced image to smoother $S$. It is also in the center of $s$ circles $Q_1, \ldots, Q_s$ with radii $R_1, \ldots, R_s$. Each circle $Q_s$ contains $N_s$ number of pixels (ignoring the middle pixel) on or within the circle but out of $Q_{s-1}$. While the radius $R_s$ of circle $Q_s$ decreases, the coefficient $a_s$ also decreases and consequently the portion of intensity added to the middle pixel $Y_{mn}$ decreases.

**Averaging technique** is considered as the special case of a single circle in the defocusing technique. Since $a_i = a_j$ for $i=j=0,1,\ldots,s$, then

$$Y'_{mn} = \frac{1}{N_1} \sum_{Q_1} Y_{ij} \quad for \quad (i, j) \in Q_1 \setminus \{(m, n)\} \quad (1.4.3 - 2)$$

The other method called the max-min rule [1] is not explained here. As the name "defocusing algorithm" implies, smoothing algorithms also have the disadvantage of blurring the images.

### 1.4.4 Edge Detection

Edge detection can be applied as a final enhancement in the spatial domain. This is represented in Figure 10. As you observe block 'ED' is added to sequence processing illustrated in Figure 9 for edge enhancement as a final process. The edge enhancing algorithm can be achieved by the algorithms introduced in section 1.2.10 or by our approach to edge detection discussed in Chapter II. Edge detection using $\min$ and $\max$ operators [1] is given in following equations. These equations can be applied for the block $ED$ in sequence illustrated by Figure 10.
Figure 10. Smoothing and Edge Detection Sequence.

\[ X'_{mn} = |X'_{mn} - \min_{Q}(X'_{ij}) | \quad (1.4.4 - 1) \]

or

\[ X''_{mn} = |X'_{mn} - \max_{Q}(X'_{ij}) | \quad (1.4.4 - 2) \]

or

\[ X''_{mn} = |\max_{Q}(X'_{ij}) - \min_{Q}(X'_{ij}) | \quad (1.4.4 - 3) \]

for \( (i, j) \neq (m, n) \)

\( (i, j) \in Q \)

\( Q \) is a circle of radius \( R \) centered at the pixel \((m,n)\) and contains \( N \) pixels \((i,j)\) located on/within circle. According to this set of equations, the better edge enhancement with high intensity trace will be achieved when the intensity contrast increases. These three equations have been examined on some different gray tone images and the results indicate that equation (1.4.4.2) generates better edge trace compared to two others. Generally the edge obtained by any desired methods can be mathematically represented as

\[ E_{II}(m,n) = [X''_{mn}] \quad \text{where} \begin{cases} m = 1,2, \ldots, M \\ n = 1,2, \ldots, N \end{cases} \quad (1.4.4 - 4) \]
$X_{mn}'$ is the detected or computed intensity value of an edge pixel. The subscript II denotes original (or smoothed and contrast intensified) image to edge detector.

Briefly, the edge enhancing algorithm can be implemented in four stages: (1) pre-enhancing which consists of contrast intensification in the fuzzy domain; (2) Smoothing; (3) post-enhancing which consists of redoing the contrast intensification in the fuzzy domain; and (4) Edge detection by applying the appropriate algorithm.

1.5 System Optimization

The object of system optimization is to specify an input vector $V(t)$, which drives a system to a specified target state in such a way that during the process a predefined performance index is minimized or maximized.

Some different optimizations for linear and non-linear systems exist such as linear optimal quadratic control, switching curve strategy and dynamic programming which are most commonly used in control problems. They are all categorized as derivative methods involved with analytical solution (direct solution for differential equations) or numerical solution (approximation of differential equations, like Euler’s method). Also there are some non-derivative methods such as sequential line searches, golden ratio sequential line search and cyclic coordinate applied in multi-dimensional optimization.

1.5.1 Discrete Nonlinear System Optimization

A discrete non-linear system can be mathematically expressed as Equation
(1.5.1-1), where $S_k, S_{k+1}, U_k$ represent respectively the present state, next state and input variables. The term $g$ denotes the system function at time $k$. This function as well as the aforementioned variables can be scalar or vector.

$$S_{k+1} = g(k, S_k, U_k) \quad (1.5.1 - 1)$$

To optimize the above system, the performance index $PI$ is defined by

$$PI(U_k) = f_k(k, S_k, U_k) \quad (1.5.1 - 2)$$

The term $f$ denotes the cost function at time $k$. Usually cost function during the optimization time is constant and can be denoted by $f$. At the optimal point the system function is optimum, $g(k^*, S_k^*, U_k^*)$. And consequently the performance index is max or min, $PI^* = f(k^*, S_k^*, U_k^*)$. In Chapter IV, we consider image enhancement system as a discrete nonlinear system and will optimize the enhanced image using a multidimensional optimization, cyclic coordinate algorithm.

1.5.2 Non-Derivative Methods

The function $f$ aimed to be optimized is called objective function. To find an explicit formula for an objective function can sometimes be so complicated or impossible, then a non-derivative method is adopted to optimization problem, such as sequential line search algorithm.

1.5.3 Sequential Line Search Algorithm

Assume objective function $f$ is convex meaning the concavity does not
change, or is at least locally convex in the neighborhood of the optimal value. Without loss of the generality assume that one-dimensional function $f$ is concave up within interval $[a_i, b_i]$ during iteration $i$ of the algorithm. Then Equations (1.5.3-1) and (1.5.3-2) define two evaluation points $lep_i$ and $rep_i$ for thirds algorithm of sequential line search as

\begin{align*}
lep_i &= \frac{2 \cdot a_i + b_i}{3} \quad (1.5.3 - 1) \\
rep_i &= \frac{a_i + 2 \cdot b_i}{3} \quad (1.5.3 - 2)
\end{align*}

Terms $lep_i$ and $rep_i$ are evaluations representing left-side and right-side evaluation points respectively for objective function $f$ at iteration $i$, that is, $f_{left} = f(lep_i)$ and $f_{right} = f(rep_i)$. The Figure 11 illustrates evaluation points within interval $[a_i, b_i]$.

![Figure 11. Sequential Line Search on Direction $x$.](image)

The length of interval $[a_i, b_i]$ is called length of uncertainty and denoted by $l_i = b_i - a_i$. According to convexity theorem [16], the uncertainty interval, $[a_{i+1}, b_{i+1}]$, in iteration $i+1$ can be computed as

\begin{align*}
\text{If } f(lep_i) < f(rep_i) & \text{ then } a_{i+1} = a_i, \quad b_{i+1} = rep_i \\
\text{If } f(lep_i) > f(rep_i) & \text{ then } a_{i+1} = lep_i, \quad b_{i+1} = b_i \\
\text{If } f(lep_i) = f(rep_i) & \text{ then } a_{i+1} = lep_i, \quad b_{i+1} = rep_i
\end{align*}

For a given initial interval $[a, b]$ and the length of uncertainty $l$, the thirds
sequential search is given by the algorithm shown in Figure 12.

![Algorithm Figure]

Figure 12. Third Sequential Line Search Algorithm.

1.5.4 Golden Ratio Line Search Algorithm

Let the evaluation points in line search algorithms be defined in terms of uncertainty boundary points $a_i, b_i$ as

$$lep_i = \alpha \cdot a_i + (1 - \alpha) \cdot b_i \quad (1.5.4-1)$$

$$rep_i = (1 - \alpha) \cdot b_i + \alpha \cdot b_i \quad (1.5.4-2)$$

If constant $\alpha$ is equal to 1, then the evaluation points fall on end points. If $\alpha$ is equal to 0.5, then the evaluation points are identical and finally for $0.5 < \alpha < 1$, these points locate between boundary points $a_i, b_i$. The famous golden ratio from Greek antiquity is obtained approximately as $\alpha = 0.618$, [16]. The golden ratio line search algorithm is given for one-dimensional objective function as shown in Figure 13.
INPUT  \( L, a, b \)
\( lep = (0.618).a + (0.382).b \)
\( rep = (0.382).a + (0.618).b \)
CALCULATE  \( f_{left} = f(lep) \)
CALCULATE  \( f_{right} = f(rep) \)
DO WHILE  \( b-a > l \)
  IF  \( f_{left} < f_{right} \) THEN
    \( b = rep \)
    \( rep = lep \)
    \( lep = (0.618).a + (0.382).b \)
    CALCULATE  \( f_{lep} \)
    \( f_{left} = f_{lep} \)
  IF  \( f_{left} >= f_{right} \) THEN
    \( a = lep \)
    \( lep = rep \)
    \( rep = (0.382).a + (0.618).b \)
    CALCULATE  \( f(rep) \)
    \( f_{left} = f_{right} \)
    \( f_{right} = f(rep) \)
END DO
PRINT  \( (a+b)/2, f((a+b)/2) \)

Figure 13. Golden Ratio Sequential Line Search Algorithm.

1.5.5 Multidimensional Optimization

Line search algorithms are techniques for optimization of a one-dimensional objective function. In system optimization, for many problems higher dimension is necessary to model system function as objective to optimize. In cases where there are no analytic means available, numerical methods are adopted for optimization. These methods may involve derivatives, but more often that not, the problems will require non-derivatives.

Assume that the two-dimensional system function \( g(x,y) \) is considered to be optimized subject to the constraint \( f(x,y) \). Sequential search methods adopted for one-
dimensional line search in sections 1.5.3 and 1.5.4, are also applicable in higher dimensions. This is done in multidimensional optimization by choosing a direction, then looking for the min/max along this line. In general this will require a line search. Once the minimum is found along this line a new direction is chosen and the procedure repeats. This continues until a reasonable approximation to min/max is achieved. This approach to optimum point from origin is generally illustrated on contour lines of the surface $f(x,y)$ in Figure 14.

As can be seen, the line search algorithm starts from origin and by sequentially choosing line $d_1$, $d_2$, $d_3$ and finally $d_4$ ends at an optimal point. The best directions can be found by gradient of $f$ (that is, the partial derivations of $f$). This is called steepest decent algorithm, [16].
1.5.6 Cyclic Coordinate Algorithm

As we told before the directions in the line search algorithm can be obtained by the gradient of function $f$. In case there is no explicit formula for gradient, non-derivative methods are helpful by cycling through a protocol of directions, independent of function $f$. In this technique of optimization directions are adopted parallel to the coordinate axis. For example, in the case of a two dimension $x$-$y$, first try the $x$-axis and then the $y$-axis, again repeating this process. Extension to higher degree systems is obvious.

To show this algorithm, consider a two-dimensional optimization problem for a function $f(c_1, c_2)$. The main structure for the cyclic coordinate algorithm is given in Figure 15.

![Figure 15. Cyclic Coordinate Algorithm.](image)

This main program, called cyclic, requires the initial point $c = (c_1, c_2)$, the
tolerance $\varepsilon$ (by which we can tune the amount of closeness to optimal point) and $l$ (the length of uncertainty during a line search). The sub-algorithm GOLDEN (golden ratio sequential line search) is shown in Figure 16.

In order for a line search be initiated, it is necessary to know just where to place the interval of uncertainty, $[a,b]$. This is done using a fixed distance $d$ as

---

**Figure 16. Golden Ratio as a Sub-Algorithm of Cyclic Coordinate Algorithm.**

```
GOLDEN(c₁, c₂, dir, d, l; c₁, c₂)

IF dir = 1 THEN a = c₁ - d, b = c₁ + d
IF dir = 2 THEN a = c₂ - d, b = c₂ + d

lep = (0.618).a + (0.382).b
rep = (0.382).a + (0.618).b

IF dir = 1 THEN CALCULATE fᵢᵢₜᵢ = f(lep, c₂), CALCULATE fᵢᵢᵣᵢᵦᵦᵢ = f(rep, c₂)
IF dir = 2 THEN CALCULATE fᵢᵢₜᵢᵦᵦᵢ = f(c₁, lep), CALCULATE fᵢᵢᵣᵢᵦᵦᵢ = f(c₁, rep)

DO WHILE b-a > l
  IF fᵢᵢₜᵢᵦᵦᵢ < fᵢᵢᵣᵢᵦᵦᵢ THEN
    b = rep
    rep = lep
    lep = (0.618).a + (0.382).b
    fᵢᵢᵣᵢᵦᵦᵢ = fᵢᵢₜᵢᵦᵦᵢ
    IF dir = 1 THEN CALCULATE fᵢᵢₜᵢᵦᵦᵢ = f(lep, c₂)
    IF dir = 2 THEN CALCULATE fᵢᵢₜᵢᵦᵦᵢ = f(c₁, lep)
  ELSE
    a = lep
    lep = rep
    rep = (0.382).a + (0.618).b
    fᵢᵢₜᵢᵦᵦᵢ = fᵢᵢᵣᵢᵦᵦᵢ
    IF dir = 1 THEN CALCULATE fᵢᵢᵣᵢᵦᵦᵢ = f(rep, c₂)
    IF dir = 2 THEN CALCULATE fᵢᵢᵣᵢᵦᵦᵢ = f(c₁, rep)
  END IF

END DO

IF dir = 1 THEN RETURN c₁ = (a+b)/2, c₂ = c₂
IF dir = 2 THEN RETURN c₁ = c₁, c₂ = (a+b)/2
```
\( a = c_1 - d, \ b = c_1 + d \) on direction \( V = (1,0) \) or flag \( dir = 1 \) and \( a = c_2 - d, \ b = c_2 + d \) on direction \( V = (0,1) \) or flag \( dir = 2 \). The output from the cyclic is simply the final, min, point found by the algorithm.

The cyclic must call a line search algorithm such as golden ratio line search, considered as the best sequential line search approach, to perform the actual line search. Parameters \( c_1, c_2, l, d \) are passed from cyclic to golden and also flag \( dir \) is specified in cyclic to indicate which coordinate in golden is to be subjected to the line search algorithm. At the end, golden will pass the best point \( c = (c_1, c_2) \) back to cyclic as well. The golden ratio line search algorithm is given as sequentially follows.

The following chart, Figure 17, illustrates the inter-module communication requires for the cyclic coordinate algorithm. \( c^* \) and \( f^* \) represent respectively the optimal point (or vector) and optimal value as the output of algorithm. The block \textit{FUNCTION} represents the calculation of \( f_{left} \) \textit{and} \( f_{right} \) in an individual module. The given algorithm which is for a two-dimensional system can be simply extended to a higher dimension.

This algorithm is just for directions parallel to \( c_1 \) and \( c_2 \) axis. Choosing a diagonal direction may help to reach the minimal point faster. However for a \( n \)-dimensional we can consider \( n \) directions parallel to the main axis and \( 2^n - n - 1 \) diagonal directions. In the case of a two-dimensional function there are two directions \( (1,0) \) and \( (0,1) \) parallel to main axis, and one diagonal direction \( (1,1) \). This algorithm is applied in Chapter IV to minimize the performance index of edge detector system by tuning a vector \( V \) (with 12 components) as an input to this algorithm.
1.6 Some Mathematical Operations

In this section some basic operations on matrices and vectors, applied further as image processing tools, are explained.

1.6.1 Inner Product

Let $U$ and $V$ be vectors in $\mathbb{R}^n$: $U = (u_1, u_2, \ldots, u_n)$ and $V = (v_1, v_2, \ldots, v_n)$. The inner or dot product of $U$ and $V$, denoted by $U \cdot V$ or $\langle U, V \rangle$, is the scalar obtained by multiplying corresponding components and adding the resulting products.

$$ U \cdot V = \langle U, V \rangle = u_1 v_1 + u_2 v_2 + \ldots + u_n v_n \quad (1.6.1-1) $$

$U$ and $V$ are said to be orthogonal or perpendicular if their dot product is zero: $U \cdot V = 0$. 

Figure 17. Inter-Module Communication for the Cyclic Coordinate Algorithm.
1.6.2 Norms

Let $U$ be a vector in $\mathbb{R}^n$: $U = (u_1, u_2, \ldots, u_n)$ and $A$ be a matrix in $\mathbb{R}^{m \times n}$: $A = [a_{ij}]$ where $i = 1, \ldots, m$ and $j = 1, \ldots, n$, then the norms of the vector $U$ and the matrix $A$ are defined as follows:

1. **Vector norm** or the length of the vector $U$, written $\|U\|$, is the nonnegative square root of $U \cdot U$ as

$$\|U\| = \sqrt{U \cdot U} = \sqrt{\langle U, U \rangle} = \sqrt{u_1^2 + u_2^2 + \cdots + u_n^2} \quad (1.6.2-1)$$

The norm of vector $U$ which is actually the distance between vector $U$ and vector zero, is obtained by root square of inner product $\langle U, U \rangle$.

2. **Matrix norm**, written $\|A\|$, is the minimum value of $k$ such that

$$\|A \cdot X\| \leq k\|X\| \quad (1.6.2-2)$$

Geometrically, multiplication by a matrix $A$ changes the length of a vector. Choose the vector $X_0$ whose length is increased the most, then $\|A\|$ is the ratio of the length of $A \cdot X_0$ to length of $X_0$. Two major properties of any matrix norm are

$$\begin{align*}
1) \quad & \|A \cdot X_0\| \leq \|A\| \cdot \|X_0\| \\
2) \quad & \|A\| = 0 \quad \text{iff} \quad A = 0
\end{align*} \quad (1.6.2-3)$$
1.6.3 Distance and Angle Definitions in Inner Product Space

If \( U \) and \( V \) are two \( n \)-dimensional vectors defined in vector space \( \mathbb{R}^n \), then the distance between two vectors \( U \) and \( V \) is defined by Equation (1.6.3-1). Where \( d(U, V) \) is the distance between vectors \( U = (u_1, u_2, ..., u_n) \) and \( V = (v_1, v_2, ..., v_n) \), which can be also considered as the norm of vector \( U-V \), written \( \|U-V\| \).

\[
d(U, V) = \|U - V\| = \sqrt{(u_1 - v_1)^2 + (u_2 - v_2)^2 + \cdots + (u_n - v_n)^2} \tag{1.6.3-1}
\]

The angle between these two vectors \( U \) and \( V \) is defined by Equation (1.6.3-2). If \( U \) and \( V \) are orthogonal, then the inner product \( \langle U, V \rangle \) is zero and consequently the angle between the two vectors is 90° degree.

\[
\theta = \cos^{-1}\left( \frac{\langle U, V \rangle}{\|U\| \cdot \|V\|} \right) \tag{1.6.3-2}
\]

1.6.4 Statistical Operation

Let \( U \) be a vector in \( \mathbb{R}^n \): \( U = (u_1, u_2, ..., u_n) \), then operators \( \text{max}, \text{min}, \text{mean}, \text{median} \) are defined over components of vector as follows:

1. **Operator 'max'**. Maximum operation is defined on vector \( U \) by specifying maximum component \( u_i \) of \( U \).

\[
u_i = \max(U) \quad \text{where} \quad u_i \geq u_j \quad \text{for} \quad j = 1, 2, ..., n \tag{1.6.4-1}
\]
2. **Operator 'min'**. Minimum operation is defined on vector $U$ by specifying minimum component $u_i$ of $U$.

$$u_i = \min(U) \quad \text{where} \quad u_i \leq u_j \quad \text{for} \quad j = 1, 2, \ldots, n \quad (1.6.4-2)$$

3. **Operator 'mean'**. Mean operation is defined by averaging of components of vector $U$.

$$u_{\text{avg}} = \text{mean}(U) \quad \text{where} \quad u_{\text{avg}} = \frac{1}{n} \sum_{i=1}^{n} u_j \quad \text{for} \quad j = 1, 2, \ldots, n \quad (1.6.4-3)$$

4. **Operator 'median'**. Median operation is defined on vector $U$ by, (a) increasingly sorting the components, ordering components of $U$ and (b) finding the center value. This operation is expressed as

$$u_{\text{med}} = \text{median}(U) \quad (1.6.4-4)$$

This operation can be easily understood by Figure 18.

![Figure 18. Illustration of Median Operation.](image-url)
1.6.5 Mean and Square Errors

If $e$ is a matrix $M \times N$, considered as a error between two matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ for $i=1,...,M$ and $j=1,...,N$, then $e$ is defined as

$$e = |A - B| = [e_{ij}] \quad \text{such that} \quad e_{ij} = |a_{ij} - b_{ij}| \geq 0 \quad (1.6.5-1)$$

$E(m,n)$, mean error at pixel $(m,n)$ can be obtained by

$$ME(m,n) = \text{mean}_{i,j} \{e_{ij}\} \quad \text{for } \begin{cases} i = 1,...,m \\ j = 1,...,n \end{cases} \quad (1.6.5-2)$$

Consequently, total mean error $TME$ is a mean error at point $(M,N)$ as

$$TME = ME(M,N) = \text{mean}_{i,j} \{e_{ij}\} \quad \text{for } \begin{cases} i = 1,...,M \\ j = 1,...,N \end{cases} \quad (1.6.5-3)$$

Let $e$ be an array of $M \times N$ considered as a error matrix, then the square error can be expressed as

$$SQerror = \|e\|^2 \quad (1.6.5-4)$$

The term $\|e\|$ is the norm of matrix $e$ defined by Equation (1.6.5-1). In the case of $e$ is a vector, then the above defined square error represents the length of $e$.

1.6.6 Two Dimensional Correlation

Consider two matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ for $i=1,...,M$ and $j=1,...,N$, then two dimensional correlation between $A$ and $B$ matrices is defined as
1.6.7 The t-norm, t-conorm, and Aggregation Operators

The t-norm $T$ and the t-conorm $S$ are two algebraic operators defined over the continuous interval $[0,1]$ by taking the combination of algebraic product $\cdot$, sum $+$, \textit{min} ($\wedge$) and \textit{max} ($\vee$). These operators map the 2-dimentional point $(x,y) \in [0,1] \times [0,1]$ to the point $z \in [0,1]$. Some t-(co)norm operators are given in Table 2.

\[
TDC = \frac{\sum_i \sum_j a_{ij} \cdot b_{ij}}{\sqrt{\sum_i \sum_j a_{ij}^2 \cdot \sum_i \sum_j b_{ij}^2}} \quad \text{for} \quad \begin{cases} i = 1, 2, \ldots, M \\ j = 1, 2, \ldots, N \end{cases} \\
\]

The \textbf{t-norm operator $T$} must satisfy at least the properties called axiomatic skeleton for all $x,y,z \in [0,1]$. These properties are (a) boundary conditions: $T(x,1) = x$ and $T(x,0) = 0$, (b) monotonicity: if $y \leq z$, then $T(x,y) \leq T(x,z)$, or strict monotonicity: if $x_1 \leq x_2$ and $y_1 \leq y_2$, then $T(x_1,y_1) \leq T(x_2,y_2)$, (c) commutativity: $T(x,y) = T(y,x)$, and (d) associativity: $T(x,T(y,z)) = T(T(x,y),z)$.

The \textbf{t-conorm operator $S$} must also satisfy at least the properties of axiomatic skeleton for all $x,y,z \in [0,1]$. These properties are (a) boundary conditions: $S(x,1) = 1$ and $S(x,0) = x$; (b) monotonicity: if $y \leq z$, then $S(x,y) \leq S(x,z)$, or strict monotonicity: if $x_1 \leq x_2$ and $y_1 \leq y_2$, then $S(x_1,y_1) \leq S(x_2,y_2)$; (c) commutativity: $S(x,y)$
\[ S(y, x) \]; and (d) associativity: \( S(x, S(y, z)) = S(S(x, y), z) \).

### Table 2

<table>
<thead>
<tr>
<th>Operators</th>
<th>( t)-norm ( T(x, y) )</th>
<th>( t)-conorm ( S(x, y) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard operators</td>
<td>Min: ( \min(x, y) ), ( x \land y )</td>
<td>Max: ( \max(x, y) ), ( x \lor y )</td>
</tr>
<tr>
<td>Algebraic operators</td>
<td>Product: ( x \cdot y )</td>
<td>Sum: ( x + y - x \cdot y )</td>
</tr>
<tr>
<td>Bounded operators</td>
<td>Difference: ( \max(0, x + y - 1) ), ( 0 \lor (x + y - 1) )</td>
<td>Sum: ( \min(1, x + y) ), ( 1 \land (x + y) )</td>
</tr>
</tbody>
</table>

The **Quasi \( t\)-norm** \( Q_t \) and the **Quasi \( t\)-conorm** \( Q_s \) [17] are the special kinds of \( t\)-norm and \( t\)-conorm operators which meet all above mentioned axiomatic skeleton properties except the associativity. These operators are represented as

\[
\begin{align*}
Q_t(x, y) &= T \left( T(x, y), S(x, y) \right) \\
Q_{st}(x, y) &= S \left( S''(x, y), T(x, y) \right)
\end{align*}
\]  \( (1.6.7 - 2) \)

**where** \( x, y \in [0, 1] \), \( Q_t \xleftarrow{dual} Q_s \), \( T \xleftarrow{dual} S \), \( T \xleftarrow{dual} S'' \), \( S \xleftarrow{dual} T'' \)

\( T, T', T'' \) are different \( t\)-norms and \( S, S', S'' \) are also different \( t\)-conorms. If the operator \( T \) and \( S \) are Demorgan's like dual, then \( S = S'' \) and \( T = T'' \). Quasi \( t\)-norms and Quasi \( t\)-conorms do not necessarily meet associativity property, therefore they are not necessarily \( t\)-norms and \( t\)-conorms. \( Q_t \) and \( Q_s \) are Demorgan's like dual, therefore we
have \( Q_s(x,y) = 1 - Q_t(1-x,1-y) = 1 - T'(T(1-x,1-y), S(1-x,1-y)) \). The Quasi t-norm \( Q_t \) satisfies at least boundary conditions, strict monotonicity, and commutativity properties of other t-norms as well as the property of inequality \( Q_t(x,y) \leq T(x,y) \). Also \( Q_s \) satisfies at least boundary conditions, strict monotonicity, and commutativity properties of other t-norms as well as the property of inequality \( Q_s(x,y) \geq S(x,y) \). Table 3 illustrates how to make the desired t-(co)norms.

Table 3

<table>
<thead>
<tr>
<th>( T(x,y) = a )</th>
<th>( S(x,y) = b )</th>
<th>( T'(x,y) )</th>
<th>( Q_s(x,y) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x \cdot y )</td>
<td>( x + y - x \cdot y )</td>
<td>( a \cdot b )</td>
<td>( x \cdot y (x+y-x \cdot y) )</td>
</tr>
<tr>
<td>Hamacher product: ( \frac{x + y}{x + y - x \cdot y} )</td>
<td>Hamacher sum: ( \frac{x + y - 2x \cdot y}{1 - x \cdot y} )</td>
<td>( a \cdot b )</td>
<td>( \frac{(x \cdot y)(x+y-2x \cdot y)}{(x+y-x \cdot y)(1-x \cdot y)} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( T''(x,y) = c )</th>
<th>( S''(x,y) = d )</th>
<th>( S'(x,y) )</th>
<th>( Q_s(x,y) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x \cdot y )</td>
<td>( x + y - x \cdot y )</td>
<td>( c + d + c \cdot d )</td>
<td>( (x+y \cdot x \cdot y) (x+y-x \cdot y) )</td>
</tr>
<tr>
<td>Hamacher product: ( \frac{x + y}{x + y - x \cdot y} )</td>
<td>Hamacher sum: ( \frac{x + y - 2x \cdot y}{1 - x \cdot y} )</td>
<td>( c + d + c \cdot d )</td>
<td>( \frac{(x+y)^2 + (x \cdot y)(1-2x-2y+x \cdot y)}{(x+y-x \cdot y)(1-x \cdot y)} )</td>
</tr>
</tbody>
</table>

Aggregation operator \( g \) is an averaging operator defined over the continuous interval \([0,1]\) by taking the combination of algebraic product (\( \cdot \)), sum (+), min (\( \land \)) and max (\( \lor \)). These operators map the \( n \)-dimensional point \( (x_1, \ldots, x_n) \in [0,1]^n \) to the point \( z \in [0,1]^n \).
\[(x_1, \ldots, x_n) \xrightarrow{g} z \quad \text{where} \quad x_1, \ldots, x_n, z \in [0,1] \quad (1.6.7 - 3)\]

All aggregation operations must satisfy at least the, (a) boundary conditions: \(g(0,0,\ldots,0) = 0\) and \(g(1,1,\ldots,1) = 1\), (b) monotonicity: if \(x_n \leq y_n\), then \(g(x_1,\ldots,x_n) \leq g(y_1,\ldots,y_n)\), and (c) continuity. Table 4 shows some two-dimensional averaging operators denoted by \(M\).

**Table 4**

Illustration of Some Averaging Operators

<table>
<thead>
<tr>
<th>Averaging Operator</th>
<th>Arithmetic Mean</th>
<th>Root Square Mean</th>
<th>Harmonic Mean</th>
<th>Quasi Linear Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>(M(x,y))</td>
<td>(\frac{x+y}{2})</td>
<td>(\sqrt{x \cdot y})</td>
<td>(\frac{2x \cdot y}{x+y})</td>
<td>(p \left[ \frac{1}{\left( \frac{x^p + y^p}{2} \right)^{\frac{1}{p}}} \right]) for (0 \leq p \leq 1)</td>
</tr>
</tbody>
</table>

The \(t\)-norms and \(t\)-conorms are the tools to define respectively the intersection and union between two fuzzy set \(A\) and \(B\). The aggregation operators are also used to combine the fuzzy sets such a way to produce the desired fuzzy set. In Chapter II we apply these operators to modify an intensity image to obtain the desired intensity image. Since intensity image processing is defined as an operation over the intensity interval \([0,1]\) (with \(L\) gray level), then \(t\)-norm, \(t\)-conorm, and the aggregation operators can be used as the tools of image modification. We will also discuss about how these operators can be applied for extension of the two-valued algebraic operators (Boolean algebra) to \(n\)-valued algebraic operators.
The standard $t$-norm, $\text{min}$ operator, results in the biggest value among the other $t$-norms. Also the standard $t$-conorm, $\text{max}$ operator, generates the least value among the other $t$-conorms. If the aggregation operator $g$ satisfies the additional property of idempotency which is $g(x_1, x_2, ..., x_l) = x_1$, then the values of all aggregation operations fall between two values obtained by the standard $t$-norm and $t$-conorm operators, $\text{max}$ and $\text{min}$. Also As said, Since Quasi $t$-norms $Q_t$ and Quasi $t$-conorms $Q_s$ does not necessarily satisfy the associativity property, They are not necessarily $t$-norms and $t$-conorms. However, the following inequality is true between $Q_t, Q_s, T, S$ operators and the standard operators, $\text{min} (\wedge)$ and $\text{max} (\vee)$.

$$Q_t(x, y) \leq T(x, y) \leq \text{min}(x, y) \leq g(x, y) \leq \text{max}(x, y) \leq S(x, y) \leq Q_s(x, y) \quad (1.6.7-4)$$

As seen, the Quasi $t$-norm $Q_t$ and its dual obtained by combinations of $t$-norms and $t$-conorms. Consequently, by combinations of either $t$-norms $T_1, T_2$ or $t$-conorms $S_1, S_2$, we can also define different operators such as the $t$-norm $A_t$ and its dual $t$-conorm $A_s$ or the $t$-norm $B_t$ and its dual $t$-conorm $B_s$.

$$ \begin{align*}
A_t(x, y) &= T(T_1(x, y), T_2(x, y)) \quad (1.6.7-5) \\
A_s(x, y) &= S(S_1(x, y), S_2(x, y)) \quad (1.6.7-6) \\
B_t(x, y) &= S(T_1(x, y), T_2(x, y)) \quad (1.6.7-7) \\
B_s(x, y) &= T(S_1(x, y), S_2(x, y)) \quad (1.6.7-8)
\end{align*} $$

where \( x, y \in [0,1] \), \( A_t \xrightarrow{\text{dual}} A_s \), \( T \xrightarrow{\text{dual}} S \), \( T_1 \xrightarrow{\text{dual}} S_1 \), \( T_2 \xrightarrow{\text{dual}} S_2 \)
The operators $A_1$ and $B_1$ must meet at least the boundary conditions, strict monotonicity and commutativity properties of $t$-norms plus the inequality property

$$A_1(x,y) \leq \min(T_1(x,y), T_2(x,y)) \leq \min(x,y)$$

for operator $A_1$ and $B_1(x,y) \geq \max(T_1(x,y), T_2(x,y))$ for operator $B_1$. Also the operators $A_s$ and $B_s$ must satisfy the boundary conditions, strict monotonicity and commutativity properties of $t$-conorms plus the inequality property $A_s(x,y) \geq \max(S_1(x,y), S_2(x,y)) \geq \max(x,y)$ for operator $A_s$ and $B_s(x,y) \leq \min(T_1(x,y), T_2(x,y))$ for operator $B_s$. $A_1, A_s, B_1, B_s$ are not necessarily associative.

**Compensatory operator $C$** was first defined by Zimmerman to model the human use of the “and” more precisely. Since the use of the operator $\text{min} (\land)$ and algebraic product $(.)$ is not very appropriate for this purpose, he defined the compensatory operators as: (a) by taking the combination of algebraic product and sum for example, $(x \cdot y)^{1-p} (x + y - x \cdot y)^p$. (b) by taking the convex combination of $\text{min} (\land)$ and $\text{max} (\lor)$ operators such as, $(x \land y)(1-p) + (x \lor y)p$. In general, we can define the compensatory operator as

$$C(x,y) = F(x,y)^{1-p} \cdot G(x,y)^p$$

(1.6.7 - 9)

$$C(x,y) = F(x,y) \cdot (1-p) + G(x,y) \cdot p$$

(1.6.7 - 10)

where $x, y, p \in [0,1]$ and $F, G \in \{T, S, Q_t, Q_s, A_t, A_s, B_t, B_s, M, S_s\}$

The operators $F$ and $G$ can be the above defined operators, $T, S, Q_t, Q_s, A_t, A_s, B_t, B_s$, the averaging operators $M$ (aggregation operators $g(x,y)$) and symmetric sums
operators can be defined by using different combinations of above mentioned operators. The compensatory operators $C$ and its dual $C^*=1-C(1-x,1-y)$ must satisfy at least the boundary conditions $C(0,0)=C^*(0,0)=0$ and $C(1,1)=C^*(1,1)=1$, strict monotonicity and commutativity properties. Besides, they must also meet continuity and inequality as: if $F(x,y) \leq G(x,y)$ then $F(x,y) \leq C(x,y) \leq G(x,y)$ for operator $C$, and if $F^*(x,y) \leq G^*(x,y)$ then $F^*(x,y) \leq C^*(x,y) \leq G^*(x,y)$ for operator $C^*$.

**Generalized compensatory operator** $C_m$ is defined by averaging operation $M$ between previously defined $F$ and $G$ as given in Equation (1.6.7-11). The dual of $C_m$ denoted by $C_m^*$ can be derived by converting $M$, $F$, $G$ to their dual $M^*$, $F^*$, $G^*$ respectively.

$$
\begin{align*}
C_m(x,y) &= M(F(x,y),G(x,y)) \\
C_m^*(x,y) &= M^*(F^*(x,y),G^*(x,y))
\end{align*}
$$

(1.6.7-11) (1.6.7-12)

where $C_m \leftrightarrow_{dual} C_m^*$, $M \leftrightarrow_{dual} M^*$, $F \leftrightarrow_{dual} F^*$, $G \leftrightarrow_{dual} G^*$

for $x,y,p \in [0,1]$ and $F,G \in \{T,S,Q_t,Q_s,A_t,A_s,B_t,B_s,M,S_s\}$

The compensatory operator $C_m$ and its dual $C_m^*$ must satisfy at least the (a) boundary condition: $C(0,0)=C^*(0,0)=0$; (b) strict monotonicity; (c) commutativity; and (d) increasing, properties. Besides, they must also meet inequality such that if $F(x,y) \leq G(x,y)$ then $F(x,y) \leq C_m(x,y) \leq G(x,y)$ for operator $C$, and if $F^*(x,y) \leq G^*(x,y)$ then $F^*(x,y) \leq C_m^*(x,y) \leq G^*(x,y)$ for operator $C^*$. Table 5 gives the idea of how to construct compensatory operators from formerly defined operators $T, S, Q, \ldots$
$Q_s, A_b, A_s, B_b, B_s, M$ (aggregation operators $h(x,y)$), and symmetric sums $S_s$ which will be discussed later.

**Table 5**

<table>
<thead>
<tr>
<th>$F(x,y)$</th>
<th>$G(x,y)$</th>
<th>$M(x,y)$</th>
<th>$C_m(x,y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x,y$</td>
<td>$x \lor y$, $\max(x,y)$</td>
<td>Arithmetic Mean: $\frac{F + G}{2}$</td>
<td>$\frac{x \cdot y + \max(x,y)}{2}$</td>
</tr>
<tr>
<td>$\sqrt{x \cdot y}$</td>
<td>$\frac{x + y}{2}$</td>
<td>Harmonic Mean: $\frac{2F \cdot G}{F + G}$</td>
<td>$\frac{2 \cdot \sqrt{x \cdot y + x + y}}{4}$</td>
</tr>
<tr>
<td>Hamacher product: $\frac{x + y}{x + y - x \cdot y}$</td>
<td>Hamacher sum: $\frac{x + y - 2x \cdot y}{1 - x \cdot y}$</td>
<td>Root Square Mean: $\sqrt{F \cdot G}$</td>
<td>$\sqrt{(x \cdot y) \cdot (x + y - 2x \cdot y)}$ $\sqrt{(x + y - x \cdot y) \cdot (1 - x \cdot y)}$</td>
</tr>
<tr>
<td>$x, y$</td>
<td>$x + y - x \cdot y$</td>
<td>Quasi Averaging: $\frac{p \sqrt{F^p + G^p}}{2}$</td>
<td>$0 \leq p \leq 1$</td>
</tr>
</tbody>
</table>

**Self-dual operator $D$** is the special case of generalized compensatory operators in which $M$ is defined by the operator $M(F,G)=(F+G)/2$ such that $M$ is an arithmetic mean derived from convex combination of $F$ and $G$ using compensatory operator of $(F \land G). (1-p) + (F \lor G)p$, for $p=0.5$. The operators $F$ and $G$ are dual. Therefore, $G=F^*$ and finally, we can generally define the self-dual operators as given by Equation (1.6.7-13). The operator $F$ is one of the previously defined operators, $T$, $S$, $Q_s$, $Q_b$, $A_b$, $A_s$, $B_b$, $B_s$, $M$, and symmetric sums $S_s$ which will be discussed later.
The self dual operators $D(x,y)$ and its dual $D^*(x,y)$ must satisfy at least the, (a) boundary conditions $D(0,0)=D^*(0,0)=0$ and $C(1,1)=D^*(1,1)=1$; (b) commutativity; (c) continuity and increasing; (d) self-duality $D(x,y)=D(1-x,1-y)=D^*(x,y)$; and (e) $D(x,1-x)$, properties.

\[
\begin{align*}
D(x,y) &= \frac{F(x,y) + F^*(x,y)}{2} \\
\text{where } &x, y \in [0,1] \text{ and } F \in \{T, S, Q_t, Q_s, A_t, A_s, B_t, B_s, M, S_s\}
\end{align*}
\]

Symmetric sum $S_s$ is the special case of self dual operator which is first defined by Silvert. The general form of symmetric sums is given by Equation (1.6.7-14). The operator $F$ is one of the previously defined operators, $T, S, Q_t, Q_s, A_t, A_s, B_t, B_s, M$. Some examples of symmetric sum operators $S_s(x,y)$ are given in Table 6.

\[
\begin{align*}
S_s(x,y) &= \frac{F(x,y)}{F(x,y) + F(1-x,1-y)} \\
\text{where } &x, y \in [0,1] \text{ and } F \in \{T, S, Q_t, Q_s, A_t, A_s, B_t, B_s, M, S_s\}
\end{align*}
\]

Table 6
Illustration of Some Symmetric Operators

<table>
<thead>
<tr>
<th>Type of $F(x,y)$</th>
<th>$F(x,y)$</th>
<th>$S_s(x,y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>t-norm, $T$</td>
<td>$x,y$</td>
<td>$\frac{x\cdot y}{1 - x - y + 2\cdot x\cdot y}$</td>
</tr>
<tr>
<td>t-conorm, $S$</td>
<td>$\text{Max}(x,y)$</td>
<td>$\frac{\text{max}(x,y)}{1 +</td>
</tr>
<tr>
<td>Quasi t-norm, $Q_t$</td>
<td>$x\cdot y(x+y\cdot x\cdot y)$</td>
<td>$\frac{x\cdot y(x + y - x\cdot y)}{1 - x - y + 2\cdot x\cdot y(x + y - x\cdot y)}$</td>
</tr>
</tbody>
</table>
The symmetric operator $S_s(x,y)$ and its dual $S_s^*(x,y)$ must satisfy at least, (a) all properties defined for self-dual operators; (b) the condition, $S_s(x,1-x)=0.5$; and (c) the continuity, if $F(0,x)=0$ for $\forall x \in [0,1]$, then $S_s(0,1)$ is not defined, that is, $S_s(x,y)$ is not continuous at points $(0,1)$ and $(1,0)$, otherwise $S_s(x,y)=S_s(x,y)=0.5$. 
CHAPTER II

QCW EDGE DETECTION

In this chapter an edge characteristic function (ECF) is proposed using a structure of ideal edge patterns based on quadruple child windowing (QCW) within a specified block of an image. Also, a correlation, using distance and angle, between QCW pattern (defined within processing block) and an ideal edge pattern is applied to create an edge estimator within a processing block of an intensity image.

2.1 Intensity or Gray Tone Distance

Let \( X_{ij} \) and \( X_{mn} \) be the respectively intensities (gray levels) of \((i,j)^{th}\) and \((m,n)^{th}\) pixels. The intensity distance is defined as

\[
d((m,n),(i,j)) = |X_{mn} - X_{ij}|
\]  

(2.1-1)

Thus, the difference between the intensity values is called intensity distance. This is applied in section 2.1.3 to define ECF with respect to four defined distances within a specified block of an image based on QCW.

2.1.1 Analyzing Structure (Morphological Operation)

Morphological operations provide information concerning the form or structure of an image. Morphology is mathematically introduced by Matheron and
Serra [10]. The morphological operators decompose an image to its main features and characteristics of its objects. Operations on binary images such as dilation and erosion (respectively adding and removing pixels at the boundaries of objects), perimeter determination (finding boundaries of objects), and area estimation (estimating for total area of the objects) are examples of morphological operations. The operations are applied to certain sub-groups of pixels within the running block (called filter window). We investigate sub-grouping of the pixels on/within a $W \times W$ block of an image based on an operation called quadruple child windowing ($QCW$). $QCW$ divides each $W \times W$ block into four sub-groups, then applies certain operations to obtain desired features such as edges.

2.1.2 Presentation of Binary Edge Patterns

Let $B$ be a running block $3 \times 3$ over a binary image $BI$ at point $(m,n)$ as illustrated in Figure 3. As known, edges are points where the intensity changes abruptly. To present edge patterns in a binary image, without loss of the generality, the edge points can be considered as white pixels surrounded by some black pixels in neighborhoods. Consider block $B$ with a white pixel (high intensity) in the center. The possible structures of edge patterns can be classified as Figure 19.

The illustrated configurations for classes 1 and 2 denote the vertical edge patterns in a binary image. Similarly, classes 3 and 4 indicate horizontal edge patterns. Therefore, according to the above classification, if any $3 \times 3$ block within a
binary image matches with one of these 9 patterns then the middle pixel of that block is considered as an edge pixel.

![Figure 19. Presentation of Nine Binary Edge Patterns Within a Block of 3x3.](image)

Using the intensity distance Equation (2.1-1), each of above defined edge patterns can be coded as 8-bit words (or vector), \([d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7]\), where the components of this vector denote the eight intensity distances between middle pixel \((m,n)\) and its eight neighborhoods. Generally these distances are found by

\[
d_k = d_k(m,n) = |X_{mn} - A_k| \quad \text{for} \quad k=0,1,2,...,7
\]  

(2.1.2-1)

\(A_k\)s are the intensities of neighbors of the middle pixel \((m,n)\). This is illustrated in Figure 20.

![Figure 20. Presentation of Intensity Distances Within the Block of 3x3.](image)
According to this definition we can assign 9 codes for all 9 edge patterns as given in Table 7.

**Table 7**

**Binary Edge Pattern Codes in Block of 3x3**

<table>
<thead>
<tr>
<th>Classes</th>
<th>B</th>
<th>B</th>
<th>B</th>
<th>B</th>
<th>B</th>
<th>B</th>
<th>B</th>
<th>B</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>I 0 0 0</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
</tbody>
</table>

In edge extraction of a binary image, the edge characteristic function $f$ can be derived according to the assigned edge pattern codes. This can be achieved by logic design (using Karnough-map) to define $f$ in terms of $d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7$.

Binary image edge detector can be easily designed to find edge pixels over a binary image using $f$. As illustrated in Figure 21, the detector has eight binary inputs (eight distances obtained from each 3x3 processing block as inputs) and one binary output, $Y$. If any processing block (running block over a binary image) matches one of the 9 edge patterns, then the output of detector goes to 1 (the intensity value of middle pixel $X_{mn}=1$); otherwise the output is zero.

**Figure 21. A Binary Edge Detector for 3x3 Block.**
In the case of a noisy image this kind of detector has errors and may detect a noise pixel as edge. Therefore, noise removal algorithms must be used before using this method for edge detection. One technique by which we can remove the noise as well as decrease the number of inputs to the edge detector is to use representative pixels of sub-windows. That is, by dividing the pixels on/within the main block (window) into smaller groups (sub-windows), it is possible to find an appropriate representative pixel for each sub-group. Further we will find, to present the edge patterns in each processing block, we need at least four representatives neighboring the middle pixel \((m,n)\). Thus, we choose four sub-widows within the block \(B\) of an image and we call these quadruple child windows, \((QCW)\). We apply noise removal operations over each sub-window to find a proper representative intensity for each sub-window. Finally, by determining four intensity distances of \(d_1, d_2, d_3, d_4\) between the middle pixel \((m,n)\) and four representatives, we derive a four dimension ECF, written \(f(d_1, d_2, d_3, d_4)\), to detect the edge within any desired size of main block \(B\).

2.1.3 Quadruple Child Windowing

Let \(B\) be a running \(W\times W\) block (where \(W\) is odd) over an intensity image \(II\) at point \((m,n)\). This block \(B\) can be defined as

\[
B = [X_{ij}] \quad \text{for} \quad \left\{ \begin{array}{l}
i = m - \frac{W - 1}{2}, \ldots, m + \frac{W - 1}{2} \\
j = n - \frac{W - 1}{2}, \ldots, n + \frac{W - 1}{2}
\end{array} \right. \quad (2.1.3 - 1)
\]
$X_{mn}$ denotes the intensity value of the middle pixel of the processing block. Now consider four specified child windows $B_1, B_2, B_3, B_4$ defined inside the main block $B$. Each child window contains pixels $X_{ij}$ on/within the main block $B-\{(m,n)\}$. The QCW on block $B$ at pixel $(m,n)$ is a partition which results in four groups of pixels as $B_k = \{(p,q)\}$ for $k=1,2,3,4$ where the elements $(p,q) \in B-\{(m,n)\}$. Child windows are exhaustive and mutually exclusive. This is mathematically represented in Equation (2.1.3-2).

1) \[
B \xrightarrow{QCW} \{\{B_k\} \mid (m,n) \notin B_k \text{ for } k = 1,2,3,4\}
\]
2) \[
\{\{B_1 \cup B_2 \cup B_3 \cup B_4\} \cup \{(m,n)\}\} \subseteq B
\]
\[
B_1 \cap B_2 \cap B_3 \cap B_4 = \emptyset
\]

The structure of QCW (the configurations of these four child windows) within the running block $B$ can be geometrically altered region by region over an image based on a specific application. For example, in an image smoothing algorithm, this structure can be fixed and for the edge detection may change region by region depending on the edge patterns within the block $B$.

We investigate the various structures of QCW (QCW patterns) within a main block $B$ with different sizes of $3 \times 3, 5 \times 5, 7 \times 7$ as follows:

1. **Main Block of $3 \times 3$.** The possible QCW patterns within block $3 \times 3$ can be illustrated as shown in Figure 22. We consider these structures an standard QCW configurations within block $3 \times 3$. 
Figure 22. QCW Configurations Within a Block of 3×3.

As seen, some pixels on/within the block 3×3 participate in sub-grouping. parts a and b of Figure 22 illustrate respectively a cross-shaped and diagonal-shaped QCW. Further we will see the combination of the results obtained by pixel manipulation over cross-QCW and diagonal-QCW gives us a better result in edge detection. This combination of diagonal and cross QCW patterns is illustrated in part c of Figure 22. The type of structure of QCW affects the outcome of the operations over main block B. The cross-QCW results differ from the diagonal-QCW and also from the combination of cross and diagonal types. The result quality of each type depends on the edge pattern and the transition of gray scale within the main block B.

2. **Main Block of 5×5.** The possible structures for QCW patterns over block 5×5 are illustrated as Figure 23. We consider these structures an standard QCW configurations within block 5×5.
Figure 23. QCW Configurations Within a Block of 5x5.

The above QCW configurations are based on sub-grouping of the pixels which are diagonally or crossly located neighboring the middle pixel. Depending on specific applications, we may take all pixels into consideration in QCW. The possible morphological structures for 5x5 windows are illustrated in [10]. For example, one
possible configuration, which is applied for detecting the edge within a 5×5 block in Chapter IV, is illustrated in Figure 24.

![Figure 24. A Possible QCW Configuration (Non-Standard) Within a Block of 5×5.](image)

3. **Main Block of 7×7.** The possible structures for QCW patterns over a 7×7 block can be easily figured. Each sub-window in a cross-QCW structure within a block of 7×7 consists of three pixels. Consequently, it may contain three or nine pixels in the two possible diagonal-QCW structures. The other configurations in which all pixels are involved in structure of QCW pattern are given in [10]. As the size of the block $B$ increases, the number of pixels participating in QCW increases. This obviously helps to improve the noise removal, but it also diminishes the quality of the output in edge enhancement. We have applied 5×5 and 3×3 block processing for edge detection algorithms. In the case of processing a noisy image, a 5×5 block is adopted to remove the noise by child window filtering.
2.1.4 Child Window Processing

Let \( B_1, B_2, B_3, B_4 \) be an appropriate QCW over a \( W \times W \) block \( B \). Each \( B_k \) can be considered as a child-filter specified by an operation (modification) \( g \) on pixels belonging in each the child window. This operation may apply for removing the noise, increasing the contrast or for other purposes. The result of this operation is a scalar function of each child window (output of each child filter) is considered as the representative intensity value for that child window. Commonly used tools for the operation \( g \) include \textit{max}, \textit{min}, \textit{median}, \textit{mean} operators or any aggregation operator defined in section 1.6.7. In section 1.2.7, we defined a median filter as a noise removal tool, now we can apply median operation on pixels belonging to each child window to remove noise before edge detecting process. Applying median filtering over four child windows results in four scalar outputs \( Y_1, Y_2, Y_3, Y_4 \) representing four intensity representatives respectively relating to four child windows \( B_1, B_2, B_3, B_4 \). These representatives can be generally expressed by

\[
Y_k = g\{ X_{ij} \mid (i,j) \in B_k \} \quad \text{for } k = 1,2,3,4. \tag{2.1.4-1}
\]

By choosing an appropriate QCW pattern discussed in section 2.1.3, we apply median filtering over the four child windows to obtain \( Y_1, Y_2, Y_3, Y_4 \) as representatives of QCW. Consequently, QCW patterns which are equivalent to those QCW patterns introduced in section 2.1.3, can be reconstructed by means of representative pixel intensities \( Y_1, Y_2, Y_3, Y_4 \) in neighboring the middle pixel intensity \( X_{mn} \). The possible QCW configurations are represented in Figure 25.
In case of processing of a noisy image, $X_{mn}$ must be replaced with the representative intensity $X'_{mn}$ obtained by the following operation $g$ over the $B-(m,n)$. Operation $g$ can be either defined by Equation 1.6.7-3 or chosen from Table 4.

$$X'_{mn} = g\{ X_{ij} \mid (i,j) \in \{ B-(m,n) \} \} \quad (2.1.4-2)$$

or

$$X'_{mn} = g\{ Y_1, Y_2, Y_3, Y_4 \} \quad (2.1.4-3)$$

2.1.5 Presentation of Binary QCW-Edge Patterns

Assume $B_1, B_2, B_3, B_4$ denote an appropriate QCW within a $WxW$ block $B$ of a binary image. Also let $Y_1, Y_2, Y_3, Y_4$ be four intensity representatives of four child windows. Now we can represent possible binary QCW-edge patterns considering intensity values of middle pixel $X_{mn}$ and the representatives $Y_1, Y_2, Y_3, Y_4$. Consider the diagonal configuration for QCW pattern illustrated in Figure 25. The physical binary edge patterns can be illustrated as Figure 26. Classes 1 and 2 ($BEP_1$ and $BEP_2$) represent vertical edge patterns while class 3 and 4 indicate horizontal edge patterns. In all nine defined configurations, edge points posses high intensity (white).
Figure 26. Presentation of Binary Diagonal QCW-Edge Pattern Configurations.

The cross configuration of QCW can be considered as a 45° degree rotation of the diagonal shape; that is, if we rotate an image 45° degree then the vertical and horizontal edges turn to diagonal edges. Now consider cross-shaped QCW as illustrated in Figure 2.4.1-1, then QCW-edge patterns based on the representative pixels $Y_1, Y_2, Y_3, Y_4$ and the middle pixel $X_{mn}$ can be presented as Figure 27.

As seen in Figure 27 for binary cross QCW-edge pattern configurations, the patterns are characterized by a 45° degree clock-wise rotation of the respective diagonal QCW configurations from Figure 26. Classes 1 and 2 are results of rotation of the vertical edges and now they represent the edges in a northeast to southwest direction. Classes 3 and 4 are results of rotation of the horizontal edges and now they represent the edges in a northwest to southeast direction.
Figure 27. Presentation of Binary Cross QCW-Edge Pattern Configurations.

If any QCW pattern within a block $B$ running over an binary image matches with one of the above ideal edge pattern classes, then the middle pixel $X_{mn}$ is considered an edge point. This can be implemented by comparing the processing QCW pattern within block $B$ with all nine ideal QCW-edge patterns. To achieve this purpose, we can assign a 4-bit code word addressing each of the above defined QCW-edge patterns, $BEP_p$ for $p=1,2,\ldots,9$. These code words can be meaningfully constructed by means of four intensity distances between the binary middle pixel intensity $X_{mn}$ and the four binary representatives $Y_1, Y_2, Y_3, Y_4$. These distances are generally obtained by following Equation.

$$
    d_k = d_k(m,n) = |X_{mn} - Y_k| \quad \text{for} \quad k = 1,2,3,4. \quad (2.1.5-1)
$$
$Y_k$ represents the result of operation $g$ over the pixels belonging to child window $B_k$. (output of child filter $k$-th). Here, the operation $g$ is a median filtering as a noise removal tool. These intensity distances $d_1, d_2, d_3, d_4$ are illustrated in Figure 28.

By convention, we consider that a black pixel possesses 0 intensity value and a white pixel has intensity 1. We can now assign a code word for each of the binary QCW-edge patterns given in Figure 26 and 27 using Equation (2.1.5-1). These words are tabulated in Table 8.

### Table 8

<table>
<thead>
<tr>
<th>Classes</th>
<th>$BEP_1$</th>
<th>$BEP_2$</th>
<th>$BEP_3$</th>
<th>$BEP_4$</th>
<th>$BEP_5$</th>
<th>$BEP_6$</th>
<th>$BEP_7$</th>
<th>$BEP_8$</th>
<th>$BEP_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Codes</td>
<td>1010</td>
<td>0101</td>
<td>1100</td>
<td>0011</td>
<td>0111</td>
<td>1110</td>
<td>1101</td>
<td>1011</td>
<td>1111</td>
</tr>
</tbody>
</table>

As seen in Table 8, the nine classes of edge patterns are specified by 9 code words $BEP_p$ for $p=1,2,\ldots,9$. These code words can be generally defined as

![Figure 28. Intensity Distances in QCW-Patterns.](image-url)
The element $d_k$ is defined by Equation (2.1.5-1). As a comparison, the code word assigned for QCW-edge patterns in Table 8 possess fewer components than those code words given in Table 7. Since the number of intensity distances within the block $B$ is decreased from at least 9 to 4 using QCW technique, the determination of binary ECF based on intensity distances $d_1, d_2, d_3, d_4$ is more efficient. This Binary edge characteristic function is determined in next section.

2.1.6 QCW-Edge Characteristic Function for Binary Images

In preceding sections, we have explained QCW over a $W \times W$ block $B$ of a binary image, introduced the possible configurations of binary edge patterns based on QCW, and finally, assigned a 4-bit code word for each edge pattern as an identification based on defined intensity distances within QCW as $d_1, d_2, d_3, d_4$.

Now our special attention is given to derive a logical function in terms of $d_1, d_2, d_3, d_4$ by using the tools of switching theory. These tools include switching algebra, truth tables, and the minimization procedures. By employing these tools, we derive the logical function considered as the binary QCW edge characteristic function, QCW-ECF. This system function, $E_{qcw}(d_1, d_2, d_3, d_4)$, is illustrated in Figure 29.

![Figure 29. Edge detector Functioning Based on QCW-ECF.](image)
The binary $QCW$-edge detector can be designed based on logical function $E_{qcw}$. The vector $PBP$ represents the $QCW$ pattern within the processing block $B$. Now, we explain how to derive the binary $QCW$-$ECF$ step by step by means of logic design as follows:

1. **Truth table.** In the first step of our design, we construct the truth table. Let $d_1, d_2, d_3, d_4$ (vector $PBP$) be the inputs and $E$ be a single output, then the truth table can be constructed based on the idea that the output $E$ is active high ($=1$), if and only if $d_1, d_2, d_3, d_4$ match with one of the code words assigned for edge patterns ($PBP = BEP_p$). Since there are four input variables, then there are $2^4 = 16$ possible 4-bit word $PBP$ while nine words out of 16 words are just the edge patterns $BEP_p$ for $p=1,2,...,9$. Table 9 illustrates the truth table for this binary edge detector.

### Table 9

**Truth Table for the Binary $QCW$ Edge Detector**

<table>
<thead>
<tr>
<th>$PBP$</th>
<th>$MI\text{nterms}$</th>
<th>Pattern</th>
<th>$E$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0100</td>
<td>4</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0101</td>
<td>5</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0110</td>
<td>6</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0111</td>
<td>7</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1000</td>
<td>8</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1001</td>
<td>9</td>
<td>NE</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1010</td>
<td>10</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1011</td>
<td>11</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1100</td>
<td>12</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1101</td>
<td>13</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1110</td>
<td>14</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1111</td>
<td>15</td>
<td>E</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Note:** NE: Input is Not an Edge pattern.
E: Input is an Edge pattern.
Edge characteristic function according to this table is the sum of min-terms 3, 5, 7, 10, 11, 12, 13, 14, 15 which is given in Equation (2.1.6-1).

\[ E = E_{qcw}(d_1, d_2, d_3, d_4) = \Sigma (3, 5, 7, 10, 11, 12, 13, 14, 15) \quad (2.1.6 - 1) \]

2. **Minimization of** \( E_{qcw} \). In the second step, with the aid of the Karnough map, the algebraic expression corresponding to \( E_{qcw} \) can be minimized. Figure 30 shows the map of edge classes along with their respective inputs, and the minimization procedure for ECF given in Equation (2.1.6-1) is also illustrated where (+) and (.) are respectively the Boolean logic operators OR and AND. To minimize \( E_{qcw} \), we classify the above mentioned min-terms 3, 5, 7, 10, 11, 12, 13, 14, 15 as

\[
\begin{array}{cccc}
\text{d'3 . d'4} & \text{00} & \text{01} & \text{11} & \text{10} \\
\text{d'3 . d4} & 01 & 1 & 1 & 1 \\
\text{d3 . d4} & 11 & 1 & 1 & 1 \\
\text{d3 . d'4} & 10 & 1 & 1 & 1 \\
\end{array}
\]

Figure 30. Karnough Map.

\[ E = \Sigma (12, 13, 14, 15) + \Sigma (3, 7, 11, 15) + \Sigma (5, 7, 13, 15) + \Sigma (10, 11, 14, 15) \quad (2.1.6 - 2) \]

\[
\begin{align*}
E_{h1} &= \Sigma (12, 13, 14, 15) = d_1 \cdot d_2 \\
E_{h2} &= \Sigma (3, 7, 11, 15) = d_3 \cdot d_4 \\
E_{v1} &= \Sigma (5, 7, 13, 15) = d_2 \cdot d_4 \\
E_{v2} &= \Sigma (10, 11, 14, 15) = d_1 \cdot d_3 \\
\end{align*}
\]
Considering diagonal configuration of QCW, the expressions $E_{h1}$ and $E_{h2}$ denote the horizontal edges and the expressions $E_{v1}$ and $E_{v2}$ indicate vertical edges. The algebraic expression defining the horizontal edges can be obtained as

$$E_h = E_{h1} + E_{h2} = d_1 \cdot d_2 + d_3 \cdot d_4 \quad (2.1.6-7)$$

Consequently, the algebraic expression for the vertical edges is obtained as

$$E_v = E_{v1} + E_{v2} = d_2 \cdot d_4 + d_1 \cdot d_3 \quad (2.1.6-8)$$

As said, vertical and horizontal edges $E_v$, $E_h$ is just defined for diagonal-QCW configurations. They are called diagonal edges for cross-QCW configurations. Obviously, when the QCW configuration is based on some interest pixels such as QCW configuration represented in Figure 24, then $E_v$ and $E_h$ do not precisely indicate the vertical and horizontal edges. Anyway, the minimal expression $E$, can be finally written in terms of algebraic expressions $E_h$ and $E_v$ as the Equation (2.1.6-9). This minimal expression corresponding to $E_{qCW}$ expressed by the sum of products of input variables is considered as QCW-Edge Characteristic Function, QCW-ECF, for binary images.

$$E = E_{qCW}(d_1, d_2, d_3, d_4) = E_h + E_v = d_1 \cdot d_2 + d_3 \cdot d_4 + d_2 \cdot d_4 + d_1 \cdot d_3 \quad (2.1.6-9)$$

The Equation (2.1.6-8) can be expressed in terms of product of sums and is valid for both the diagonal and the cross configurations of QCW. Inputs $d_1, d_2, d_3, d_4$ can be obtained by Equation (2.1.5-1).
Each processing block $B$ may generally contain a diagonal, vertical or horizontal edges. Obviously, the diagonal-\textit{QCW} works the best for diagonal edge detection and consequently the cross-\textit{QCW} is good for vertical and horizontal edges. The combination of the detected edges by cross-\textit{QCW} configuration, denoted by $E_{\text{cross}}$, and by diagonal-\textit{QCW}, denoted by $E_{\text{diagonal}}$, within each processing block $B$ improves the result. Since one of the configuration diagonal-\textit{QCW} or cross-\textit{QCW} may function well within each processing block, then we can combine the results of both configuration as

$$E_{\text{combined}} = (E_{\text{cross}} + E_{\text{diagonal}})$$

(2.1.6-11)

$E_{\text{cross}}$ can be obtained by Equation (2.1.6-11) based on $d_1, d_2, d_3, d_4$ given in configuration (a) of Figure 28. $E_{\text{diagonal}}$ can be also obtained by Equation (2.1.6-11) based on $d_1, d_2, d_3, d_4$ given in configuration (b) of Figure 28.

2.1.7 \textit{QCW}-Edge Characteristic Function for Intensity Images

The two-valued (binary) logical function, binary \textit{QCW-ECF} in Equation (2.1.6-10), possesses the value 1 if the variables $d_1, d_2, d_3, d_4$ are the binary values matching one of the edge pattern code words, otherwise it possesses the value 0. In an intensity image, the input variables $d_1, d_2, d_3, d_4$ possess a continuum of intensity values on $[0,1]$ (a discrete gray level $l$ between 0-255). Therefore, we should derive a multi-valued logical function $E_{\text{qcw}}$ to define edge based on multi-valued input
variables $d_1, d_2, d_3, d_4$. This can be achieved by applying the multi-valued logic operators instead of Boolean logic operators in Equation (2.1.6-10). Using operator conversions given in Table 1, the Equation (2.1.6-10) can be extended to Equation (2.1.7-2) as the generalized \textit{QCW-ECF} for both binary and intensity images. It is possible to derive directly the \textit{ECF} for intensity image from binary \textit{ECF}, Equation (2.1.6-10), without the conversion of operators. One may consider the Boolean operators (+) and (.) as the ordinary arithmetic operation. Since an intensity values for $d_1, d_2, d_3, d_4$ are ranged over interval $[0,1]$, the maximum possible intensity value resulted by Equation (2.1.6-10) for $E_{qcw}$ is 4. Then, by dividing this equation by 4, we can define a function representing a normalized intensity \textit{ECF}. This function can be simply expressed as Equation (2.1.7-2), but it results in a low intensity value.

\[ E_{qcw}(d_1, d_2, d_3, d_4) = 0.25 (d_1 + d_4) \cdot (d_2 + d_3) \quad (2.1.7-1) \]

Generalized \textit{QCW-ECF} for intensity images based on t-norm and t-conorm can be defined by extension of Equation (2.1.6-10) to (2.1.7-2). For further reference, we call it the general \textit{JK-type} edge characteristic function based on quadruple child windowing.

\[
E = E_{qcw}(d_1, d_2, d_3, d_4) = T_{qcw}(S_{qcw}(d_1, d_4), S_{qcw}(d_2, d_3))
\]

\[
\text{for } d_1, d_2, d_3, d_4 \in [0,1] \quad , \quad T_{qcw} \xrightarrow{\text{dual}} S_{qcw}
\]

\[
\text{where } T_{qcw} \in \{T, Q_t, A_t, B_t\}, \quad S_{qcw} \in \{S, Q_s, A_s, B_s\}
\]
$E$ is the degree of edginess of $PBP$ at middle pixel $(m,n)$. $T_{qcw}$ and $S_{qcw}$ are $t$-norm and $t$-conorm operators which can be desirably derived based on the given definitions in section 1.6.7 and Equations (1.6.7-1) and (1.6.7-2). As discussed in section 1.6.7, we can derive a lot of $t$-norms and $t$-conorms. Therefore, the generalized $QCW$-$ECF$ given in Equation (2.1.7-2) is a base to generate a lot of edge characteristic functions.

Consequently, we can extend the Boolean logical expressions concerning the horizontal and vertical edges, $E_h$ and $E_v$ for diagonal-$QCW$ configuration, given by Equations (2.1.6-7) and (2.1.6-7), to multi-valued logical functions as

$$E_h(d_1, d_2, d_3, d_4) = S_{qcw}(T_{qcw}(d_1, d_2), T_{qcw}(d_3, d_4)) \quad (2.1.7-3)$$

$$E_v(d_1, d_2, d_3, d_4) = S_{qcw}(T_{qcw}(d_2, d_4), T_{qcw}(d_1, d_3)) \quad (2.1.7-4)$$

Also by extension of Equation (2.1.6-9), another expression is obtained for the intensity $QCW$-$ECF$ as

$$E_{qcw}(d_1, d_2, d_3, d_4) = S_{qcw}(E_h, E_v) \quad (2.1.7-5)$$

$E_h$ and $E_v$ are defined in Equations (2.1.7-3) and (2.1.7-4). It is reassuring to note that the Equations (2.1.7-2) and (2.1.7-5) do not always result in the same value, since $t$-norms and $t$-conorms do not satisfy the property of associativity. This is simulated in Chapter IV and, based on choosing the appropriate $QCW$ configuration, the result is competitive with the best traditional algorithms in edge detection.
As said, since either the diagonal-\textit{QCW} or cross-\textit{QCW} configuration may work properly within each processing block, then we can combine the results of both configuration as

\[ E_{\text{combined}} = S_{\text{com}}(E_{\text{cross}}, E_{\text{diagonal}}) \quad (2.1.7-6) \]

This equation is the extension of Equation (2.1.6-11). The operator \( S_{\text{com}} \) is a t-conorm operator previously defined in section 1.6.7 such that \( S_{\text{com}} \in \{ S, Q_s, A_s, B_s \} \). Both \( E_{\text{cross}} \) and \( E_{\text{diagonal}} \) can be individually obtained by Equation (2.1.7-2) or (2.1.7-5) based on \( d_1, d_2, d_3, d_4 \) illustrated respectively in parts (a) and (b) of Figure 28.

As discussed in section 1.6.7, we have \( Q_t(x,y) \leq T(x,y) \leq \min(x,y) \leq g(x,y) \leq \max(x,y) \leq S(x,y) \leq Q_s(x,y) \) given by Equation (1.6.7-4). We just consider Quasi t-(co)norms and standard t-(co)norms to generate the least and the biggest values among the other t-(co)norms. Referring to the general operator given by Equation (1.6.7-8), Equation \( E_{\text{qcw}}(d_1, d_2, d_3, d_4) \) defined by t-norm combination of two t-conorms must be a t-conorm operation over \( d_1, d_2, d_3, d_4 \). Based on the Table 10 containing some desired t-(co)norms, we derive five special cases of \textit{QCW-ECF} from general Equation (2.1.7-2). The quasi-type t-(co)norms mentioned in rows 3,4,5 of this table, are derived using the Equation (1.6.7-2) and the standard and algebraic t-(co)norm operators represented in first and second rows. These five special cases of \textit{QCW-ECF} are constructed based on the operators mentioned in rows 1,2,3,4,5 of Table 10. We call these edge characteristic functions \textit{JK-type} 1, 2, 3, 4, 5 and can be obtained as follows:
Table 10
Illustration of Some t-(co)norn Operators Applied for QCW-ECF

<table>
<thead>
<tr>
<th>No.</th>
<th>Operators</th>
<th>t-norms, (T_{qcw})</th>
<th>t-conorms, (S_{qcw})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Standard operators</td>
<td>Min: (\text{Min}(x, y))</td>
<td>Max: (\text{max}(x, y))</td>
</tr>
<tr>
<td>2</td>
<td>Algebraic operators</td>
<td>Product: (x.y)</td>
<td>Sum: (x+y-x.y)</td>
</tr>
<tr>
<td>3</td>
<td>Quasi operator</td>
<td>(Q_r = T(T', S): \text{[min}(x, y) \text{]} \cdot (x+y-x.y))</td>
<td>(Q_r = S(S'', T''): \text{[max}(x, y) \text{]} \cdot (1-x-y) \cdot \text{max}(x, y))</td>
</tr>
<tr>
<td>4</td>
<td>Quasi operator</td>
<td>(Q_r = T(T', S): \text{[max}(x, y) \text{]} \cdot \text{[max}(x, y) \text{]} \cdot \text{max}(x, y))</td>
<td>(Q_r = S(S'', T''): \text{[min}(x, y) \text{]} \cdot \text{[min}(x, y) \text{]} \cdot \text{[min}(x, y) \text{]} \cdot (1-x-y) \cdot \text{[min}(x, y) \text{]} \cdot \text{[min}(x, y) \text{]} )</td>
</tr>
<tr>
<td>5</td>
<td>Quasi operator</td>
<td>(Q_r = T(T', S): \text{[max}(x, y) \text{]} \cdot \text{[max}(x, y) \text{]} \cdot \text{max}(x, y))</td>
<td>(Q_r = S(S'', T''): \text{[min}(x, y) \text{]} \cdot \text{[min}(x, y) \text{]} \cdot \text{[min}(x, y) \text{]} \cdot (1-x-y) \cdot \text{[min}(x, y) \text{]} \cdot \text{[min}(x, y) \text{]} )</td>
</tr>
</tbody>
</table>

1. **JK-type 1** of generalized QCW-ECF can be obtained from Equation (2.1.7-2) by simply using the standard operators \(\text{min}\) and \(\text{max}\) which are respectively t-norm \(T_{qcw}\) and t-conorms \(S_{qcw}\). Therefore, the JK-type 1 of QCW-ECF can be defined as

\[
E = E_{qcw}(d_1, d_2, d_3, d_4) = \text{min}[\text{max}(d_1, d_4), \text{max}(d_2, d_3)]
\]  

Equation (2.1.7-7)

Consequently we can apply these operators to find JK-type 1 of vertical and horizontal edges, \(E_v\) and \(E_h\), as follows

\[
E_h = E_h(d_1, d_2, d_3, d_4) = \text{max}(\text{min}(d_1, d_2), \text{min}(d_3, d_4))
\]  

Equation (2.1.7-8)

\[
E_v = E_v(d_1, d_2, d_3, d_4) = \text{max}(\text{min}(d_2, d_4), \text{min}(d_1, d_3))
\]  

Equation (2.1.7-9)

\[
E = E_{qcw}(d_1, d_2, d_3, d_4) = \text{max}(E_h, E_v)
\]  

Equation (2.1.7-10)
In case of considering both diagonal and cross QCW-ECF, to combine these two configuration we can use Equation (2.1.7-11) as the special case of Equation (2.1.7-6) where the t-conorm $S_{com}$ is considered the standard operator, $\max$.

$$E_{combined} = \max(E_{cross}, E_{diagonal}) \quad (2.1.7-11)$$

2. **JK-type 2** of generalized QCW-ECF can be derived from Equation (2.1.7-2) by simply using the algebraic operators, product $(x.y)$ and sum $(x+y-x.y)$, (refer to the second row of table (2.1.7-1)) considered respectively t-norm $T_{qcw}$ and t-conorms $S_{qcw}$. Therefore, the JK-type2 of QCW-ECF can be defined by Equation (2.1.7-14).

$$S_{14} = S_{qcw}(d_1, d_4) = d_1 + d_4 - d_1.d_4 \quad (2.1.7-12)$$

$$S_{23} = S_{qcw}(d_2, d_3) = d_2 + d_3 - d_2.d_3 \quad (2.1.7-13)$$

$$E = E_{qcw}(d_1, d_2, d_3, d_4) = T_{qcw}(S_{14}, S_{23}) = S_{14}.S_{14}$$

$$= (d_1 + d_4 - d_1.d_4).(d_2 + d_3 - d_2.d_3) \quad (2.1.7-14)$$

Consequently, we can apply this operators to find JK-type2 of vertical and horizontal edges, $E_v$ and $E_h$. Therefore, using Equations (2.1.7-3), (2.1.7-4) and (2.1.7-5), we have $T_{12} = T_{qcw}(d_1, d_2) = d_1.d_2$, $T_{34} = T_{qcw}(d_3, d_4) = d_3.d_4$, $T_{13} = T_{qcw}(d_1, d_3) = d_1.d_3$, $T_{24} = T_{qcw}(d_2, d_4) = d_2.d_4$. Finally we can write

$$E_h = E_h(d_1, d_2, d_3, d_4) = S_{qcw}(T_{12}, T_{34}) = T_{12} + T_{34} - T_{12}.T_{34}$$

$$= d_1.d_2 + d_3.d_4 - d_1.d_2.d_3.d_4 \quad (2.1.7-15)$$

$$E_v = E_v(d_1, d_2, d_3, d_4) = S_{qcw}(T_{24}, T_{13}) = T_{24} + T_{13} - T_{24}.T_{13}$$

$$= d_2.d_4 + d_1.d_3 - d_2.d_4.d_1.d_3 \quad (2.1.7-16)$$
To combine the values resulting from two diagonal and cross-QCW configurations, we can apply the algebraic operator sum in Equation (2.1.7-6) as \( t\)-conorm \( S_{com}(x,y) = (x+y+x.y) \) to derive the Equation (2.1.7-18).

\[
E_{combined} = S_{com}(E_{cross}, E_{diagonal}) = E_{cross} + E_{diagonal} - E_{cross} \cdot E_{diagonal} \quad (2.1.7-18)
\]

3. **JK-type 3** of generalized QCW-ECF can be derived from Equation (2.1.7-2) by simply using the combination of algebraic and standard operators. For this combination, we apply Quasi \( t\)-(co)norms as derived and mentioned in third row of Table 10. The Quasi operators \( Q_t \) and \( Q_s \) are considered respectively \( t\)-norm \( T_{qcw} \) and \( t\)-conorms \( S_{qcw} \). Therefore, the JK-type3 of QCW-ECF can be defined by Equation (2.1.7-21).

\[
S_{14} = S_{qcw}(d_1, d_4) = d_1.d_4 + (1 - d_1.d_4).[\text{max}(d_1, d_4)] \quad (2.1.7-19)
\]

\[
S_{23} = S_{qcw}(d_2, d_3) = d_2.d_3 + (1 - d_2.d_3).[\text{max}(d_2, d_3)] \quad (2.1.7-20)
\]

\[
E = E_{qcw}(d_1, d_2, d_3, d_4) = T_{qcw}(S_{14}, S_{23}) = [\text{min}(S_{14}, S_{23})].(S_{14} + S_{23} - S_{14} \cdot S_{23}) \quad (2.1.7-21)
\]

Consequently, we can apply theses operators to find JK-type2 of vertical and horizontal edges, \( E_v \) and \( E_h \). Therefore, using Equations (2.1.7-3), (2.1.7-4) and (2.1.7-5), we can write

\[
T_{12} = T_{qcw}(d_1, d_2) = [\text{min}(d_1, d_2)].(d_1 + d_2 - d_1.d_2) \quad (2.1.7-22)
\]
\begin{align}
T_{34} &= T_{qcw}(d_3, d_4) = \min(d_3, d_4). (d_3 + d_4 - d_3.d_4) \quad (2.1.7-23) \\
E_h &= E_h(d_1, d_2, d_3, d_4) = S_{qcw}(T_{12}, T_{34}) \\
&= T_{12}.T_{34} + (1 - T_{12}.T_{34}).[\max(T_{12}, T_{34})] \quad (2.1.7-24) \\
T_{24} &= T_{qcw}(d_2, d_4) = \min(d_2, d_4). (d_2 + d_4 - d_2.d_4) \quad (2.1.7-25) \\
T_{34} &= T_{qcw}(d_1, d_3) = \min(d_1, d_3). (d_1 + d_3 - d_1.d_3) \quad (2.1.7-26) \\
E_v &= E_v(d_1, d_2, d_3, d_4) = S_{qcw}(T_{24}, T_{13}) \\
&= T_{24}.T_{13} + (1 - T_{24}.T_{13}).[\max(T_{24}, T_{13})] \quad (2.1.7-27) \\
E &= E_{qcw}(d_1, d_2, d_3, d_4) = S_{qcw}(E_h, E_v) \\
&= E_h.E_v + (1 - E_h.E_v).[\max(E_h.E_v)] \quad (2.1.7-28) \\
\end{align}

In case of considering the combination of the diagonal and cross-QCW configuration to edge detection, we can apply the Quasi operator \( Q_t \) (mentioned in the third row of the Table 2.1.7.1 for Equation (2.1.7-6) as \( t \)-conorm \( S_{com} \equiv Q_t \), to derive the Equation (2.1.7-29).

\[ E_{combined} = S_{com}(E_{cross}, E_{diagonal}) \]

\[ = E_{cross}.E_{diagonal} + (1 - E_{cross}.E_{diagonal}).[\max(E_{cross}.E_{diagonal})] \quad (2.1.7-29) \]

4. **JK-type 4** of generalized QCW-ECF can be derived from Equation (2.1.7-2) by simply using the Quasi combination of algebraic and standard operators as mentioned in fourth row of Table 10. Therefore, by considering \( T_{qcw} \equiv Q_t \) and \( S_{qcw} \equiv Q_t \), then the JK-type 4 of QCW-ECF can be defined by Equation (2.1.7-32).

\[ S_{14} = S_{qcw}(d_1, d_4) = d_1 + d_4 - d_1.d_4 + (1 - d_1 - d_4 + d_1.d_4).[\min(d_1, d_4)] \quad (2.1.7-30) \]
\[ S_{23} = S_{qw}(d_2, d_3) = d_2 + d_3 - d_2 d_3 + (1 - d_2 - d_3 + d_2 d_3) [\min(d_2, d_3)] \]  
(2.1.7-31)

\[ E = E_{qw}(d_1, d_2, d_3, d_4) = T_{qw}(S_{14}, S_{23}) \]
\[ = (S_{14} \cdot S_{23}) [\max(S_{14}, S_{23})] \]  
(2.1.7-32)

Consequently, we can apply these operators to find JK-type of vertical and horizontal edges, \( E_v \) and \( E_h \). Therefore, using Equations (2.1.7-3), (2.1.7-4) and (2.1.7-5), we can write

\[ T_{12} = T_{qw}(d_1, d_2) = (d_1, d_2) [\max(d_1, d_2)] \]  
(2.1.7-33)

\[ T_{34} = T_{qw}(d_3, d_4) = (d_3, d_4) [\max(d_3, d_4)] \]  
(2.1.7-34)

\[ E_h = E_h(d_1, d_2, d_3, d_4) = S_{qw}(T_{12}, T_{34}) \]
\[ = T_{12} + T_{34} - T_{12} T_{34} + (1 - T_{12} - T_{34} + T_{12} T_{34}) [\min(T_{12}, T_{34})] \]  
(2.1.7-35)

\[ T_{24} = T_{qw}(d_2, d_4) = (d_2, d_4) [\max(d_2, d_4)] \]  
(2.1.7-36)

\[ T_{13} = T_{qw}(d_1, d_3) = (d_1, d_3) [\max(d_1, d_3)] \]  
(2.1.7-37)

\[ E_v = E_v(d_1, d_2, d_3, d_4) = S_{qw}(T_{24}, T_{13}) \]
\[ = T_{24} + T_{13} - T_{24} T_{13} + (1 - T_{24} - T_{13} + T_{24} T_{13}) [\min(T_{24}, T_{13})] \]  
(2.1.7-38)

\[ E = E_{qw}(d_1, d_2, d_3, d_4) = S_{qw}(E_h, E_v) \]
\[ = E_h + E_v - E_h E_v + (1 - E_h - E_v + E_h E_v) [\min(E_h, E_v)] \]  
(2.1.7-39)

In case of processing both cross and diagonal QCW configurations, we can apply the Quasi operator \( Q \) (mentioned in forth row of Table 10) for Equation (2.1.7-6) as \( t \)-conorm \( S_{\text{com}} = Q \) to derive the following equation.

\[ E_{\text{combined}} = S_{\text{com}}(E_{\text{cross}}, E_{\text{diagonal}}) \]
= E_{cross} + E_{diagonal} \cdot E_{cross} \cdot E_{diagonal}

+ (1 - E_{cross} \cdot E_{diagonal}) \cdot [\min(E_{cross} \cdot E_{diagonal})] \quad (2.1.7-40)

5. **JK-type 5** of generalized QCW-ECF can be derived from Equation (2.1.7-2) by simply using the Quasi combination of algebraic operators as mentioned in fifth row of the Table 10. Therefore, by considering $T_{qcw} = Q$, and $S_{qcw} = Q_s$, then the JK-type 5 of QCW-ECF can be defined by Equation (2.1.7-43).

\[
S_{14} = S_{qcw}(d_1, d_4) = d_1 + d_4 - (d_1 \cdot d_4) \cdot (d_1 + d_4 - d_1 \cdot d_4) \quad (2.1.7-41)
\]

\[
S_{23} = S_{qcw}(d_2, d_3) = d_2 + d_3 - (d_2 \cdot d_3) \cdot (d_2 + d_3 - d_2 \cdot d_3) \quad (2.1.7-42)
\]

\[
E = E_{qcw}(d_1, d_2, d_3, d_4) = T_{qcw}(S_{14}, S_{23})
\]

\[
= (S_{14} \cdot S_{23})(S_{14} + S_{23} - S_{14} \cdot S_{23}) \quad (2.1.7-43)
\]

Consequently, using Equations (2.1.7-3), (2.1.7-4) and (2.1.7-5), we can apply theses operators to find JK-type 5 of vertical and horizontal edges, $E_v$ and $E_h$ as follows.

\[
T_{12} = T_{qcw}(d_1, d_2) = (d_1 \cdot d_2) \cdot (d_1 + d_2 - d_1 \cdot d_2) \quad (2.1.7-44)
\]

\[
T_{34} = T_{qcw}(d_3, d_4) = (d_3 \cdot d_4) \cdot (d_3 + d_4 - d_3 \cdot d_4) \quad (2.1.7-45)
\]

\[
E_h = E_h(d_1, d_2, d_3, d_4) = S_{qcw}(T_{12}, T_{34})
\]

\[
= T_{12} + T_{34} - (T_{12} \cdot T_{34})(T_{12} + T_{34} - T_{12} \cdot T_{34}) \quad (2.1.7-46)
\]

\[
T_{24} = T_{qcw}(d_2, d_4) = (d_2 \cdot d_4) \cdot (d_2 + d_4 - d_2 \cdot d_4) \quad (2.1.7-47)
\]

\[
T_{13} = T_{qcw}(d_1, d_3) = (d_1 \cdot d_3) \cdot (d_1 + d_3 - d_1 \cdot d_3) \quad (2.1.7-48)
\]

\[
E_v = E_h(d_1, d_2, d_3, d_4) = S_{qcw}(T_{24}, T_{13})
\]
\[ E = E_{qcw}(d_1, d_2, d_3, d_4) = S_{qcw}(E_h, E_v) \]
\[ = E_h + E_v - (E_h \cdot E_v)(E_h + E_v - E_h \cdot E_v). \]  

(2.1.7-50)

Also, we can apply the Quasi operator \( Q \) (mentioned in forth row of Table 10) for Equation (2.1.7-6) as \( t \)-conorm \( S_{com} \equiv Q \), to derive the following Equation.

\[ E_{combined} = S_{com}(E_{cross}, E_{diagonal}) \]
\[ = E_{cross} + E_{diagonal} - (E_{cross} \cdot E_{diagonal})(E_{cross} + E_{diagonal} - E_{cross} \cdot E_{diagonal}). \]  

(2.1.7-51)

The edge detectors functionally find the edge (an intensity value for edge point) based on the above mentioned equations. Assume, \( B \) to be a processing block \( W \times W \) at point \((m,n)\) of an intensity image. Let \( B_1, B_2, B_3, B_4 \) be \( QCW \) within the block \( B \), and also \( Y_1, Y_2, Y_3, Y_4 \) be representative intensity values for \( QCW \) obtained by Equation (2.1.5-2), then processing block pattern \( PBP \) is a vector defined by

\[ PBP = [d_k]_p \text{ for } k = 1,2,3,4. \]  

(2.1.7-52)

The terms \( d_1, d_2, d_3, d_4 \) are four intensity distances obtained by Equation (2.1.5-1) based on \( QCW \) within the block \( B \). \( PBP \) is also considered as the characteristic vector of the intensity of \( QCW \)-pattern within the processing block \( B \). The processing block \( B \) with a certain pattern \( PBP \) can be evaluated by the edge detector which is operating functionally based on \( QCW-ECF \) given by the Equations (2.1.7-2) or (2.1.7-4) or (2.1.7-6). This has been illustrated in Figure 29. The function
\( E_{qcw} \) generates \( E \) as the evaluation value for the intensity pattern of processing block \( B \). This evaluation value indicates the maximum degree of correspondence of processing block pattern \( PBP \) to the binary edge patterns \( BEP_p \) (for \( p=1,2,\ldots,9 \)) defined in section 2.1.5. The evaluation value \( E \) is obtained as

\[
E = E_{qcw}(d_1, d_2, d_3, d_4) = E_{qcw}(PBP).
\]  

\( E \) possesses a value within \([0,1]\). The value \( E=1 \) indicates a perfect correspondence between \( PBP \) and one of the binary edge patterns. Similarly, \( E=0 \) shows no correspondence. It follows that a value between 0 and 1 for \( E \) refers to the degree of correspondence of \( PBP \) to the defined binary edge patterns. On the other hand, this degree of correspondence \( E \) indicates the degree of edginess of processing block pattern \( PBP \). Since here we are dealing with normalized value of gray level (intensity value within \([0,1]\)) and the degree of edginess is also between \([0,1]\), we can directly consider the value \( E \) as the actual intensity value of the edge pixel. In the block processing technique for edge detection this value of \( E \) is replaced instead of middle pixel intensity \( X_{mn} \) of a running block \( B \) over an intensity image.

### 2.2 Correlation Between QCW- Pattern and Binary Edge Patterns

Finding correlation coefficient is a an appropriate measurement for the degree of correspondence of the QCW pattern with the binary edge patterns. That is determination of maximum correlation between \( PBP \) vector and the space vector

\[
BEP = \{BEP_p, p=1,2,\ldots,9\}. \]

Correlation between \( PBP \) and \( BEP \) is shown in Figure 31.
The correlation method results in a maximum evaluation value in interval [0,1] as the maximum degree of correspondence between processing block pattern \( PBP \) to one of the 9 binary edge patterns \( BEP_p \). This value represents the degree of edginess of pattern \( PBP \). In the case of the normalized gray scale (intensity image), this degree of edginess can be considered as the actual intensity value of the edge pixel. The closer the \( PBP \) vector to the space vector \( BEP \), the brighter the edge pixel. In processing a binary image this correlation result in a value 0 or 1, while in an intensity image it results in a value within interval [0,1].

![Possible binary code](image)

Possible binary code
Words \( 2^4 = 16 \)

![Illustration of Correlation Between Vectors PBP and BEP](image)

- **Figure 31. Illustration of Correlation Between Vectors \( PBP \) and \( BEP_p \).**

### 2.2.1 Distance Correlation

Consider a \( QCW \) within a processing block \( B \) of an image (binary or intensity) and let \( PBP \) be the relevant characteristic vector specifying this \( QCW \) pattern. Then the distances between the \( PBP \) vector and the \( BEP_p \) vectors are generally defined as \( DC_p \) by Equation (2.2.1-1). This equation represents the correlation between the processing block pattern and the binary edge patterns.
\[ DC_p = \| PBP - BEP_p \| \quad \text{for} \quad p = 1, 2, 3, \ldots, 9 \quad (2.2.1-1) \]

Distance correlation results in a value denoting the degree of closeness of \( PBP \) vector to \( BEP_p \) vector. The space of the edge pattern vectors consists of nine vectors. Therefore, there are nine values indicating the degree of correspondence of \( PBP \) to the edge pattern vector space. Since we are looking for closest \( BEP_p \) vector to \( PBP \), then the minimum value of \( DC_p \) meets our need. As \( DC_p \) decreases toward zero the degree of edginess of \( PBP \) approaches the highest value 1. We can define the degree of edginess of \( PBP \) as

\[
E_{dc} = 1 - \frac{\min_p \left\{ \|PBP - BEP_p\| \right\}}{\max_p \left\{ \|PBP - BEP_p\| \right\}} \quad \text{for} \quad p = 1, 2, 3, \ldots, 9 \quad (2.2.1-2)
\]

According to this definition, the closer the vector \( PBP \) is to the binary code word space \( BEP \), the function \( E_{dc} \) results in the greater value as the degree of edginess for the processing block pattern. Considering this value directly as the actual edge intensity value, then the bigger value of \( E_{dc} \) results in a brighter edge point.

### 2.2.2 Angular Correlation

Consider a \( QCW \) within a processing block \( B \) of an image (binary or intensity) and let \( PBP \) be a vector containing the intensity distances defined within this \( QCW \) pattern. The degree of correspondence of \( PBP \) to binary edge patterns \( BEP_p \) is to find the minimum angle \( \Theta_p \) (or maximum \( \cos(\Theta_p) \)) between \( PBP \) vector and \( BEP_p \).
vectors. The angular correlation between the processing block pattern and binary edge patterns is generally obtained by

\[ Cos(\Theta_p) = \frac{<PBP, BEP_p>}{\|PBP\| \cdot \|BEP_p\|} \quad \text{for} \quad p = 1, 2, \ldots, 9 \quad (2.2.2 - 1) \]

This equation results in a value within interval \([0,1]\). Therefore for \(p = 1, \ldots, 9\), there are 9 different values representing the degree of correspondence of PBP vector to \(BEP_p\) vectors. When the \(PBP\) vector exactly matches with one of \(BEP_p\) vectors, then this equation possesses the value 1 (since angle between two vectors is zero) denoting the maximum degree of correspondence as well as maximum degree of edginess of \(PBP\).

By increasing the \(\Theta_p = 0\), the \(Cos(\Theta_p)\) is approaching 0 indicating no correlation while \(Cos(\Theta_p) = 1\) is denoting the maximum correlation. The degree of edginess based on the defined angular correlation in Equation (2.2.2-1) is defined as

\[ E_{ac} = \max_p \left\{ Cos(\Theta_p) \right\} \quad \text{for} \quad p = 1, 2, \ldots, 9 \quad (2.2.2 - 2) \]

2.3 Edge Detectors Based on QCW-ECF

In preceding sections, we have introduced edge characteristic functions (ECF) based on \(QCW\) within a processing block \(B\) of an image (binary or intensity). We showed that all these equations result in values denoting the degree of correspondence
of QCW pattern PBP to binary edge pattern space PEP. On the other hand, indicating the degree of edginess of middle pixel \((m,n)\) centered on the processing block \(B\). In the edge detection algorithm using block processing, the above equations can be directly applied to find the edge pixels. The result of the equation \(E_{qcw}\) is considered as intensity value of edge point and replaced by middle intensity pixel \(X_{mn}\). Finally the output image represents an edge map considering degree of edginess for each pixel of main image.

Edge detectors usually generate a binary edge map. The edge characteristic functions are applied to find the degree of edginess as an evaluation value for middle pixel \(X_{mn}\). If this evaluation value is greater than some threshold \((T_H)\), then the edge detector send one to output \((X_{mn} \text{ is replaced by } 1)\) indicating \(X_{mn}\) is an edge point. Consequently, if this degree is less than some threshold \((T_L)\), then the edge detector sends 0 to output \((X_{mn} \text{ is replaced by } 0)\) denoting \(X_{mn}\) is not an edge point. The block diagram in Figure 32 depicts the edge detector functioning based on QCW-ECF and thresholding.

![Figure 32. Edge Detection Using Thresholds.](image-url)
**Edge Detection algorithm based on QCW-ECF** can be briefly written as follows:

1. Choose a proper size of block $B$, $W \times W$, with the middle intensity $X_{mn}$.

2. Do the QCW within block $B$ using the diagonal, cross, or/and other desired configurations (based on desired pixels neighboring the middle pixel), to find four child windows $B_1, B_2, B_3, B_4$.

3. Choose an appropriate aggregation operator $g$ (such as averaging, median, ...) and apply it over each child window to find four representative intensities $Y_k = g\{ X_{ij} \mid (i,j) \in B_k \}$ for $k=1,2,3,4$. Also, in case of processing a noisy image, replace the middle intensity pixel $X_{mn}$ with the intensity representative $X'_{mn}$ where $X'_{mn} = g\{ X_{ij} \mid (i,j) \in \{B - (m,n)\} \}$.

4. Find the processing block pattern vector $PBP = [d_1, d_2, d_3, d_4]$ by finding the four difference intensities (distances) $d_k = |X_{mn} - Y_k|$, for $k=1,2,3,4$.

5. Choose or construct a desired $t$-norm $T_{qcw}$ and $t$-conorm $S_{qcw}$ using Tables 2 and 10.

6. Apply the $JK$-type generalized QCW-ECF, distance correlation, or angular correlation given respectively by Equations 2.1.7-2, 2.2.1-2, and 2.2.2-2 for each individual QCW configuration (diagonally, crossly or any desired morphological structure), to find $E = E_{qcw}(PBP) = E_{qcw}(d_1, d_2, d_3, d_4)$ the degree of correspondence of vector $PBP$ to the binary edge pattern space $BEP$ given by Table 8 (or degree of edginess of $PBP$ at middle pixel $(m,n)$). Since we applied the normalized gray scale indicating intensity is within continuous interval $[0,1]$, then this degree of
correspondence directly represents the intensity of edge pixel (if thresholding is not applied) and is replaced instead of middle pixel intensity $X_{mn}$.

7. If there are more than one configuration, combine the result of each pair of configurations obtained from step 6 using a proper $t$-conorm, $S_{com}$, to find the $E_{combined}$.

8. In case of using thresholds for generating the binary edge, the edge detector sends the binary digit 1 to output if $E$ (or $E_{combined}$ in case of combination) $\geq T_H$. Also, it sends binary digit 0 if $E$ (or $E_{combined}$) $\leq T_L$. 
In this chapter, fuzzy tools are applied to define QCW-ECF based on fuzzy logic. One significant problem in edge detection occurs when an monochrome image has a background of varying intensity or gray level. Finding an edge of the object at those points that have a similar intensity with background is difficult. Also, another problem is that in many cases, edge detection techniques are dependent on the appropriate choice of thresholds. The edge detection algorithm based on fuzzy logic with flexible rules as well as tunable membership functions would be an effective approach to remove the above mentioned problems.

3.1 Fuzzy Intensity Distances

Let $X_{ij}$ and $X_{mn}$ be fuzzy intensity singletons of $(i,j)^{th}$ and $(m,n)^{th}$ pixels within a fuzzy intensity image defined by Equation (1.4.1-1). Fuzzy intensity distance is a fuzzy singleton with a fuzzy value $d((m,n),(i,j))$, obtained by Equation (2.1-1), and its corresponding membership degree $\mu_d$ obtaining on a certain fuzzy plane (such as fuzzy plane illustrated in Figure 7. This is defined as

$$ F_d = \frac{\mu_d}{d(X_{mn}, X_{ij})} \quad \text{where} \quad \begin{cases} d((m,n),(i,j)) = |X_{mn} - X_{ij}| \\ \mu_d = P(d((m,n),(i,j))) \end{cases} \quad (3.1-1) $$
The term $F_d$ is considered a fuzzy singleton indicating the fuzzy intensity distance between two fuzzy singletons $F_{X_{mn}}$ and $F_{X_{ij}}$ representing two fuzzy intensity values of points $(m,n)$ and $(j,j)$ of image. Also, $P$ is a membership function(s) in fuzzy plane.

3.1.1 Fuzzy Quadruple Child Windowing, $FQCW$

Consider the fuzzy intensity image $FII$ represented by Equation (1.4.1-1). The $QCW$ process defined in section 2.1.3 can be applied within a $W \times W$ block over the fuzzy intensity image, $FII$. Let $FB$ be a block (fuzzy intensity block) with size $W \times W$ over an fuzzy intensity image $FII$. This block can be represented as

$$FB = [F_{X_{ij}}] = \left[ \frac{\mu_{ij}}{X_{ij}} \right]$$

for

$$i = m - \frac{W - 1}{2}, \ldots, m + \frac{W - 1}{2}$$

$$j = n - \frac{W - 1}{2}, \ldots, n + \frac{W - 1}{2}$$

(3.1.1-1)

Point $(i,j) = (m,n)$ denotes the middle pixel of $FB$ and $F_{X_{mn}}$ is a fuzzy singleton which possesses intensity value of $X_{mn}$ with membership degree of $\mu_{mn}$. Also, $F_{X_{ij}}$ denotes the fuzzy singletons neighboring the middle pixel. $FB$ can be considered as the result of fuzzification of block $B$ defined by Equation (2.1.3-1) over an intensity image. Fuzzification of the intensity image $II$ can be implemented by generating fuzzy blocks $FB$ when the block $B$ running over an intensity image.

Now we apply again the $QCW$ process to break down the block $FB$ into four child windows as $FB_1$, $FB_2$, $FB_3$, $FB_4$. We call this process as Fuzzy quadruple Child Windowing ($FQCW$), which is mathematically represented by Equation (3.1.1-2).
This is exactly the fuzzy approach for $QCW$ defined by Equation (2.1.3-1). The structure of $FQCW$ within the running fuzzy intensity block $FB$ can be geometrically altered based on application during the processing of fuzzy image. The structures of $FQCW$ can be chosen as explained and illustrated for structures of $QCW$ within the block of $(3 \times 3)$, $(5 \times 5)$, $(7 \times 7)$ in section 2.1.3.

Considering the running block $FB$ over an image $FII$, one can apply either a fixed structure of $QCW$ or different structure of $QCW$ within block $FB$. In the case of the latter, all configurations of $QCW$ are simultaneously considered to choose the best structure for $QCW$ based on the results at each processing pixel $X_{mn}$. To process an image by applying various structures of $QCW$ yields the best result but it takes more time than applying the fixed structure, because in each block all structures should be examined in order to find the best structure.

### 3.1.2 Fuzzy Child Window Processing

Let $\{FB_1, FB_2, FB_3, FB_4\}$ be an appropriate $QCW$ over a $W \times W$ block $FB$. The fuzzy child-filter can be defined by an operation (modification) $g$ on pixels on/within each child window $FB_k$. For our purposes, the filtering operation (we use median filtering) over four child windows results in four fuzzy singletons $FY_1, FY_2, FY_3, FY_4$ which are respectively considered as four fuzzy intensity representative values for...
child windows $FB_1$, $FB_2$, $FB_3$, $FB_4$. Finally the fuzzy singleton resulting from a fuzzy child window processing can be presented as

$$FY_k = \frac{\mu_k}{Y_k} \quad for \quad k = 1,2,3,4 \quad (3.1.2-1)$$

According to Equation 2.1.4-1, we have $Y_k = g(B_k)$ for $k=1,2,3,4$. The membership degree $\mu_1$, $\mu_2$, $\mu_3$, $\mu_4$ corresponding to these four fuzzy values $Y_1$, $Y_2$, $Y_3$, $Y_4$ are obtained using a certain fuzzy plain (fuzzification of these values). If $P$ denotes the continuous membership function(s) defined in a fuzzy plane, then these membership degrees are generally defined as

$$\mu_k = P(Y_k) \quad where \quad \mu_k \in [0,1] \quad for \quad k = 1,2,3,4 \quad (3.1.2-2)$$

Fuzzy child window processing is sequentially depicted in Figure 33.

Figure 33. Fuzzy Child Window Processing.

Fuzzy child window processing results in four fuzzy singletons $FY_1$, $FY_2$, $FY_3$, $FY_4$ within a fuzzy block $FB$. Consequently, $FQCW$ patterns can be illustrated by Figure 34 using these four fuzzy singletons as representatives of child windows and the $FX_{mn}$ as the fuzzy singleton representing the middle pixel.
Figure 34. Illustration of FQCW Configurations.

Figure 34 represents the geometrical configuration of fuzzy quadruple child windowing (FQCW), which is exactly like QCW but the pixels are fuzzy singletons. In case of processing of a noisy image, $X_{mn}$ must be replaced with the representative intensity pixel $X'_{mn}$ obtained by the following operation $g$ over the $B - (m,n)$.

Operation $g$ can be defined by Equation 1.6.7-3 or chosen from Table 4.

\[
X'_{mn} = g \{ X_{ij} \mid (i,j) \in \{ B - (m,n) \} \} \quad (2.1.2-3)
\]

or

\[
X'_{mn} = g\{ Y_1, Y_2, Y_3, Y_4 \} \quad (2.1.2-4)
\]

Consequently, the fuzzy singleton $FX'_{mn}$ can be defined by

\[
FX'_{mn} = \frac{\mu_{mn}}{\mu_{X_{mn}}} \quad \text{for} \quad \{ m = 1, 2, \ldots, M \} \quad \{ n = 1, 2, \ldots, N \} \quad (3.1.2 - 4)
\]

\[
\mu_{mn} = P(X'_{mn}) \quad \text{where} \quad \mu_{mn} \in [0,1] \quad \text{for} \quad \{ m = 1, 2, \ldots, M \} \quad \{ n = 1, 2, \ldots, N \} \quad (3.1.2 - 5)
\]


3.1.3 *FQCW* Intensity Pattern

Consider *FQCW* (diagonal or cross structure) given in Figure 34 within a block *FB*. Let \( FY_1, FY_2, FY_3, FY_4 \) be 4 fuzzy singletons as the intensity representatives of four child windows and \( FX_{mn} \) be the fuzzy singleton denoting middle pixel intensity. We can define 4 fuzzy singletons indicating the fuzzy intensity distances \( Fd_1, Fd_2, Fd_3, Fd_4 \) as

\[
F_{d_k} = \frac{\mu d_k}{d_k} \quad \text{for } k = 1, 2, 3, 4 \quad (3.1.3-1)
\]

The intensity distances \( d_1, d_2, d_3, d_4 \) and their associated membership degrees can be obtained by

\[
d_k = d(X_{mn}, Y_k) = |X_{mn} - Y_k| \quad (3.1.3-2)
\]

\[
\mu d_k = P(d_k) \quad (3.1.3-3)
\]

\( P \) is a membership function(s) defined over continuous intensity scale \([0,1]\) (or discrete gray level 0 to \( L-1 \)). One may choose the fuzzification of \( d_k \) using the fuzzy plane illustrated in Figure 7. The given fuzzy intensity block *FB* possesses an intensity pattern based on *FQCW*. This pattern can be specified by fuzzy vector \( FPBP \) containing intensity distances \( Fd_1, Fd_2, Fd_3, Fd_4 \) as

\[
FPBP = [Fd_k] \quad \text{for } k = 1, 2, 3, 4. \quad (3.1.3-2)
\]

Therefore, each fuzzy intensity block *FB* which is under processing can be
specified by its pattern characteristic vector \( FPBP \) based on \( FQCW \). This fuzzy vector \( FPBP \) can be considered as the fuzzified vector \( PBP \) which has been defined as a pattern characteristic vector for intensity block \( B \) based on \( QCW \). This vector is applied to define \( FQCW \)-edge Characteristic function (\( FQCW-ECF \)) in next section.

3.1.4 \( FQCW \)-Edge Characteristic Function

Consider \( FQCW \) within a fuzzy intensity block \( FB \) with its relevant code vector \( FPBP \) presenting the intensity pattern. If this \( FQCW \) intensity pattern corresponds with one of the nine binary edge patterns illustrated in Figure 26, or 27. In other words, if the fuzzy vector \( FPBP \) matches one of the binary code words defined in Table 8, then \( FQCW \) intensity pattern is considered an ideal edge pattern. This is true when the fuzzy singletons \( Fd_1, Fd_2, Fd_3, Fd_4 \) possess the ideal values like \( 1/1, 1/0 \). Normally, the fuzzy intensity block which is under processing, for each fuzzy singleton \( Fd_k \) in vector \( FPBP \), the distance \( d_k \in [0,1] \) (or gray level \( l \in \{0,1, \ldots, L-1\} \)) and its associated membership degree \( \mu d_k \in [0,1] \). We are looking to find an appropriate function represented by Equation (3.1.4-1) to evaluate the \( FQCW \) intensity pattern (\( FPBP \) vector) by comparison with nine possible ideal (binary) edge patterns given in Figures 26 or 27. The binary \( QCW \)-edge patterns illustrated in these figures can be considered as a special case of \( FQCW \)-edge patterns when the \( FPBF \) vector contains fuzzy singletons \( 1/1, 1/0 \) which referring respectively to 1 and 0 in a crisp set.
\[ E = E_{fqcw}(FPBP) = E_{fqcw}(F_d_1, F_d_2, F_d_3, F_d_4) \]
\[ = E_{fqcw}\left(\frac{\mu d_1}{d_1'}, \frac{\mu d_2}{d_2'}, \frac{\mu d_3}{d_3'}, \frac{\mu d_4}{d_4'}\right) \]  

(3.1.4-1)

\[ E_{fqcw} \] is the edge characteristic function based on \( FQCW \). The result \( E \) is a non-fuzzy (crisp) number which represents the degree of closeness to maximum intensity of an edge point 1 or maximum gray level \( L-1 \). In other words, that is maximum degree of correspondence of the \( FQCW \) intensity pattern to the defined binary edge pattern space or degree of edginess of the middle pixel of the fuzzy block \( FB \).

As the general Equation (2.1.7-2) was developed to define QCW-ECF (in both diagonal and cross configurations) for evaluating the QCW intensity pattern (or pattern characteristic vector \( PBP \)) based on degree of correspondence to the nine binary edge patterns, it is time to define the Equation (3.1.4-2). This equation represents the degree of correspondence of the \( FQCW \) intensity pattern to binary edge patterns. On the other hand, this equation defines the maximum degree of correspondence of the vector \( FPBP \) to the nine binary edge code word space, \( BEP = \{BEP_p, \} \) for \( p=1,2,\ldots,9 \).

\[
\begin{align*}
E &= E_{fqcw}\left(\frac{\mu d_1}{d_1'}, \frac{\mu d_2}{d_2'}, \frac{\mu d_3}{d_3'}, \frac{\mu d_4}{d_4'}\right) = \\
&= T_{qcw}\left[S_{qcw}\left(\frac{\mu d_1}{d_1}, \frac{\mu d_4}{d_4}\right), S_{qcw}\left(\frac{\mu d_2}{d_2}, \frac{\mu d_3}{d_3}\right)\right]
\end{align*}
\]  

(3.1.4-2)

where

\[
\begin{align*}
E, d_1, d_2, d_3, d_4, \mu d_1, \mu d_2, \mu d_3, \mu d_4 \in [0,1] \\
T_{qcw} \in \{T, Q_i, A_i, B_i\}, \quad S_{qcw} \in \{T, Q_s, A_s, B_s\}
\end{align*}
\]
The operators \( T_{qcw} \) and \( S_{qcw} \) are respectively any \( t \)-norm and \( t \)-conorm defined in section 1.6.7. The function \( E_{fqcw} \) is denoted here as \( FQCW-ECF \). That is fuzzy QCW-edge characteristic function. Applying this function for edge detection has two important advantages, (1) thresholding is not needed. In this equation, since the four membership degrees \( \mu d_1, \mu d_2, \mu d_3, \mu d_4 \) associated with the distances \( d_1, d_2, d_3, d_4 \) play the roles of weights. Also, since these membership degrees are adjustable by properly tuning each membership function, then this makes the \( FQCW-ECF \) more flexible and powerful in edge extraction without using thresholds, and (2) Since the function \( E_{fqcw} \) is a special operation over fuzzy variables \( d_1, d_2, d_3, d_4 \) and the result is a crisp value \( E \), then it can be considered as a special defuzzification method by which simultaneously fuzzy singletons are mapped to a crisp value representing the degree of edginess. These two important advantages make this function powerful in edge detection and the resulted edge points by means of this equation are extraordinary as we will observe in Chapter IV.

Consequently, by considering the diagonal QCW configuration and the Equations (2.1.7-3), (2.1.7-4) and (2.1.7-5), we can also express \( E_{fqcw} \) for \( FQCW-ECF \) by the terms of vertical \((E_v)\) and horizontal \((E_h)\) edges as follows.

\[
E_h = S_{qcw} \left[ T_{qcw} \left( \mu d_1 \cdot d_1, \mu d_2 \cdot d_2 \right), T_{qcw} \left( \mu d_3 \cdot d_3, \mu d_4 \cdot d_4 \right) \right] \quad (3.1.4 - 3)
\]

\[
E_v = S_{qcw} \left[ T_{qcw} \left( \mu d_2 \cdot d_2, \mu d_4 \cdot d_4 \right), T_{qcw} \left( \mu d_1 \cdot d_1, \mu d_3 \cdot d_3 \right) \right] \quad (3.1.4 - 4)
\]

\[
E = E_{fqcw} \left( \text{FPBP} \right) \ E_{fqcw} \left( Fd_1, Fd_2, Fd_3, Fd_4 \right) = S_{qcw} \left( E_h, E_v \right) \quad (3.1.4 - 5)
\]
Also, Equation (2.1.7-1) can be extended to $E_{fqcw}$ as

$$E = E_{fqcw}(Fd_1, Fd_2, Fd_3, Fd_4) = \frac{(\mu d_1 \cdot d_1 + \mu d_4 \cdot d_4)(\mu d_2 \cdot d_2 + \mu d_3 \cdot d_3)}{4} \quad (3.1.4-6)$$

Since either the diagonal-$FQCW$ or cross-$FQCW$ configuration functions well within each processing block, then for better result, we can combine the results of both configuration as

$$E_{combined} = S_{qcw}(E_{fcross}, E_{fdiagonal}) \quad (3.1.4-7)$$

$E_{fcross}$ and $E_{fdiagonal}$ can be obtained by Equation (3.1.4-2) based on $d_1, d_2, d_3, d_4$ given respectively in configurations (a) and (b) of Figure 34.

To develop the particular $QCW-ECF$ ($JK$-type 1, 2, 3, 4, 5 discussed in section 2.1.7) for the fuzzy model $FQCW-ECF$, we can simply replace the variables $d_1, d_2, d_3, d_4$ respectively with $(\mu d_1 \cdot d_1), (\mu d_2 \cdot d_2), (\mu d_3 \cdot d_3), (\mu d_4 \cdot d_4)$. For simplicity, let define the parameter $D_k$ as

$$D_k = \mu d_k \cdot d_k \quad \text{for } k=0,1,2,3,4 \quad (3.1.4-8)$$

The special cases of the generalized $FQCW-ECF$ given by Equation (3.1.4-2) can be derived as follows:

1. **JK-type 1** of generalized $FQCW-ECF$ can be obtained by using the Table 10 and Equation (3.1.4-2) as follows.

$$E = E_{fqcw}(FPBP) = \min[ \max(D_1, D_4), \max(D_2, D_3) ] \quad (3.1.4-9)$$
\[ E_h = E_h(FPBP) = \max(\min(D_1, D_2), \min(D_3, D_4)) \quad (3.1.4-10) \]
\[ E_v = E_v(FPBP) = \max(\min(D_2, D_4), \min(D_1, D_3)) \quad (3.1.4-11) \]
\[ E = \min(E_h, E_v) \quad (3.1.4-12) \]
\[ E_{combined} = \max(E_{cross}, E_{diagonal}) \quad (3.1.4-13) \]

2. **JK-type 2** of generalized FQCW-ECF can be obtained by using the Table 10 and Equation (3.1.4-2) as follows.

\[ S_{14} = S_{qcw}(D_1, D_4) = D_1 + D_4 - D_1.D_4 \quad (3.1.4-14) \]
\[ S_{23} = S_{qcw}(D_2, D_3) = D_2 + D_3 - D_2.D_3 \quad (3.1.4-15) \]
\[ E = E_{qcw}(FPBP) = T_{qcw}(S_{14}, S_{23}) = S_{14} \times S_{14} \]
\[ = (D_1 + D_4 - D_1.D_4)(D_2 + D_3 - D_2.D_3) \quad (3.1.4-16) \]

We have \( T_{12} = T_{qcw}(D_1, D_2) = D_1.D_2, T_{34} = T_{qcw}(D_3, D_4) = D_3.D_4, T_{13} = T_{qcw}(D_1, D_3) = D_1.D_3, T_{24} = T_{qcw}(D_2, D_4) = D_2.D_4. \) Then,

\[ E_h = E_h(FPBP) = S_{qcw}(T_{12}, T_{34}) = T_{12} + T_{34} - T_{12} . T_{34} \]
\[ = D_1.D_2 + D_3.D_4 - D_1.D_2.D_3.D_4 \quad (3.1.4-17) \]
\[ E_v = E_v(FPBP) = S_{qcw}(T_{24}, T_{13}) = T_{24} + T_{13} - T_{24} . T_{13} \]
\[ = D_2.D_4 + D_1.D_3 - D_2.D_4.D_1.D_3 \quad (3.1.4-18) \]
\[ E = E_{qcw}(FPBP) = S_{qcw}(E_h, E_v) = E_h + E_v - E_h . E_v \quad (3.1.4-19) \]
\[ E_{combined} = S_{com}(E_{cross}, E_{diagonal}) = E_{cross} + E_{diagonal} - E_{cross} . E_{diagonal} \quad (3.1.4-20) \]
3. **JK-type 3** of generalized FQCW-ECF can be obtained by using the Table 10 and Equation (3.1.4-2) as follows.

\[ S_{14} = S_{qcw}(D_1, D_4) = D_1.D_4 + (1 - D_1.D_4).[\max(D_1, D_4)] \]  
(3.1.4-21)

\[ S_{23} = S_{qcw}(D_2, D_3) = D_2.D_3 + (1 - D_2.D_3).[\max(D_2, D_3)] \]  
(3.1.4-22)

\[ E = E_{fqcw} (FPBP) = T_{qcw}(S_{14}, S_{23}) \]
\[ = [\min(S_{14}, S_{23})].(S_{14}+S_{23} -S_{14}.S_{23}) \]  
(3.1.4-23)

\[ T_{12} = T_{qcw}(D_1, D_2) = [\min(D_1, D_2)].(D_1+ D_2 - D_1.D_2) \]  
(3.1.4-24)

\[ T_{34} = T_{qcw}(D_3, D_4) = [\min(D_3, D_4)].(D_3+ D_4 - D_3.D_4) \]  
(3.1.4-25)

\[ E_h = E_h (FPBP) = S_{qcw}(T_{12}, T_{34}) \]
\[ = T_{12}.T_{34} + (1- T_{12}.T_{34}).[\max(T_{12}, T_{34})] \]  
(3.1.4-26)

\[ T_{24} = T_{qcw}(D_2, D_4) = [\min(D_2, D_4)].(D_2+ D_4 - D_2.D_4) \]  
(3.1.4-27)

\[ T_{34} = T_{qcw}(D_1, D_3) = [\min(D_1, D_3)].(D_1+ D_3 - D_1.D_3) \]  
(3.1.4-28)

\[ E_v = E_v (FPBP) = S_{qcw}(T_{24}, T_{13}) \]
\[ = T_{24}.T_{13} + (1- T_{24}.T_{13}).[\max(T_{24}, T_{13})] \]  
(3.1.4-29)

\[ E = E_{fqcw} (FPBP) = S_{qcw}(E_h , E_v) \]
\[ = E_h.E_v + (1 - E_h.E_v).[\max(E_h.E_v)] \]  
(3.1.4-30)

\[ E_{combined} = E_{cross} . E_{diagonal} + \]
\[ = (1-E_{cross}.E_{diagonal}).[\max(E_{cross} . E_{diagonal})] \]  
(3.1.4-31)

4. **JK-type 4** of generalized FQCW-ECF can be obtained by using the Table 10 and Equation (3.1.4-2) as follows.
\[ S_{14} = S_{qcw}(D_1, D_4) = D_1 + D_4 - D_1.D_4 + (1 - D_1 - D_4 + D_1.D_4)[\min(D_1, D_4)] \quad (3.1.4-32) \]

\[ S_{23} = S_{qcw}(D_2, D_3) = D_2 + D_3 - D_2.D_3 + (1 - D_2 - D_3 + D_2.D_3)[\min(D_2, D_3)] \quad (3.1.4-33) \]

\[ E = E_{fqcw}(FPBP) = T_{qw}(S_{14}, S_{23}) = (S_{14} \cdot S_{23})[\max(S_{14}, S_{23})] \quad (3.1.4-34) \]

\[ T_{12} = T_{qw}(D_1, D_2) = (D_1.D_2)[\max(D_1, D_2)] \quad (3.1.4-35) \]

\[ T_{34} = T_{qw}(D_3, D_4) = (D_3.D_4)[\max(D_3, D_4)] \quad (3.1.4-36) \]

\[ E_h = E_h(FPBP) = S_{qw}(T_{12}, T_{34}) \]

\[ = T_{12} + T_{34} - T_{12} \cdot T_{34} + (1 - T_{12} - T_{34} + T_{12} \cdot T_{34})[\min(T_{12}, T_{34})] \quad (3.1.4-37) \]

\[ T_{24} = T_{qw}(D_2, D_4) = (D_2.D_4)[\max(D_2, D_4)] \quad (3.1.4-38) \]

\[ T_{13} = T_{qw}(D_1, D_3) = (D_1.D_3)[\max(D_1, D_3)] \quad (3.1.4-39) \]

\[ E_h = E_h(FPBP) = S_{qw}(T_{24}, T_{13}) \]

\[ = T_{24} + T_{13} - T_{24} \cdot T_{13} + (1 - T_{24} - T_{13} + T_{24} \cdot T_{13})[\min(T_{24}, T_{13})] \quad (3.1.4-40) \]

\[ E = E_{fqcw}(FPBP) = S_{qw}(E_h, E_v) \]

\[ = E_h + E_v - E_h \cdot E_v + (1 - E_h - E_v + E_h \cdot E_v)[\min(E_h, E_v)] \quad (3.1.4-41) \]

\[ E_{combined} = S_{com}(E_{f\text{cross}}, E_{f\text{diagonal}}) \]

\[ = (1 - E_{f\text{cross}}) - E_{f\text{diagonal}} + E_{f\text{cross}} \cdot E_{f\text{diagonal}})[\min(E_{f\text{cross}}, E_{f\text{diagonal}})] + \]

\[ E_{f\text{cross}} + E_{f\text{diagonal}} - E_{f\text{cross}} \cdot E_{f\text{diagonal}} \quad (3.1.4-42) \]

5. **JK-type 5** of generalized FQCW-ECF can be obtained by using the Table 10 and Equation (3.1.4-2) as follows.

\[ S_{14} = S_{qcw}(D_1, D_4) = D_1 + D_4 - (D_1.D_4)(D_1 + D_4 - D_1.D_4) \quad (3.1.4-43) \]

\[ S_{23} = S_{qcw}(D_2, D_3) = D_2 + D_3 - (D_2.D_3)(D_2 + D_3 - D_2.D_3) \quad (3.1.4-44) \]
\[ E = E_{fqcw}(FPBP) = T_{qcw}(S_{14}, S_{23}) \]
\[ = (S_{14} \cdot S_{23})(S_{14} + S_{23} - S_{14} \cdot S_{23}) \tag{3.1.4-45} \]
\[ T_{12} = T_{qcw}(D_{1}, D_{2}) = (D_{1} \cdot D_{2}).(D_{1} + D_{2} - D_{1} \cdot D_{2}) \tag{3.1.4-46} \]
\[ T_{34} = T_{qcw}(D_{3}, D_{4}) = (D_{3} \cdot D_{4}).(D_{3} + D_{4} - D_{3} \cdot D_{4}) \tag{3.1.4-47} \]
\[ E_{h} = E_{h}(FPBP) = S_{qcw}(T_{12}, T_{34}) \]
\[ = T_{12} + T_{34} - (T_{12} \cdot T_{34})(T_{12} + T_{34} - T_{12} \cdot T_{34}) \tag{3.1.4-48} \]
\[ T_{24} = T_{qcw}(D_{2}, D_{4}) = (D_{2} \cdot D_{4}).(D_{2} + D_{4} - D_{2} \cdot D_{4}) \tag{3.1.4-49} \]
\[ T_{13} = T_{qcw}(D_{1}, D_{3}) = (D_{1} \cdot D_{3}).(D_{1} + D_{3} - D_{1} \cdot D_{3}) \tag{3.1.4-48} \]
\[ E_{h} = E_{h}(FPBP) = S_{qcw}(T_{24}, T_{13}) \]
\[ = T_{24} + T_{13} - (T_{24} \cdot T_{13})(T_{24} + T_{13} - T_{24} \cdot T_{13}) \tag{3.1.4-50} \]
\[ E = E_{fqcw}(FPBP) = S_{qcw}(E_{h}, E_{v}) \]
\[ = E_{h} + E_{v} - (E_{h} \cdot E_{v})(E_{h} + E_{v} - E_{h} \cdot E_{v}) \tag{3.1.4-52} \]
\[ E_{fc_{combined}} = S_{com}(E_{fcross}, E_{Diagonal}) \tag{3.1.4-53} \]
\[ = E_{fcross} + E_{fdiagonal} - (E_{fcross} \cdot E_{fdiagonal})(E_{fcross} + E_{fdiagonal} - E_{fcross} \cdot E_{fdiagonal}) \]

3.2 Correlation of FQCW Pattern With Binary Edge Patterns

There is another method by which we can evaluate the FQCW pattern within a processing block $FB$. This method is similar to methods explained in section 2.2, but here we deal with fuzzy vector $FPBP$ which compares with edge code word space $BEP$. Since we can consider binary values as a special case of fuzzy values 0 and 1 with associating membership degree 1, (1/0 or 1/1), this comparison is possible.
3.2.1 Distance Correlation of FQCW Pattern

Consider a FQCW within a processing block \( FB \) of an intensity image. Let \( FPBP \) be a the fuzzy vector characterizing the FQCW pattern, then the distances between \( FPBP \) vector and the \( BEP_p \) vectors are generally defined as Equation (2.2.1-1). This equation represents the correlation between processing block pattern \( FB \) and the binary edge patterns.

\[
FDC_p = \| DV - BEP_p \| \quad \text{for } j=1,2,3,\ldots 9 \quad (3.2.1-1)
\]

\( DV \) is a vector obtained by the conversion of the fuzzy vector \( FPBP \) to a crisp value vector as

\[
DV = [\mu d_1 \cdot d_1, \mu d_2 \cdot d_2, \mu d_3 \cdot d_3, \mu d_4 \cdot d_4] \quad (3.2.1-2)
\]

Distance correlation results in a value denoting the degree of correspondence of the \( FPBP \) vector to the edge code word \( BEP_p \). We are looking for the maximum degree of correspondence to the edge code word space \( BEP \), then the minimum value of \( FDC_p \) meets our need. We can define the degree of edginess of \( FPBP \) by Equation (3.2.1-3). According to this definition, the closer the \( FPBP \) vector is to the edge pattern vector space, the greater is the degree of edginess \( E_{fdc} \) is in value.

\[
E_{fdc} = 1 - \frac{\min_p \left\| DV - BEP_p \right\|}{\max_p \left\| DV - BEP_p \right\|} \quad \text{for } p = 1,2,3,4 \quad (3.2.1-3)
\]
3.2.2 Angular Correlation of $FQCW$ Pattern

Consider a $FQCW$ within a processing block $FB$ of an intensity image and let $FPBP$ be the characteristic vector of this $FQCW$ pattern. Angular correlation is a proper measurement for degree of correspondence of $FPBP$ to binary edge patterns $BEP_p$. That is to find the minimum angle (or maximum cosine angle) between $FPBP$ vector and $BEP_p$ vectors. The angular correlation between the processing block pattern and the binary edge patterns is generally obtained by

$$Cos(\Phi_p) = \frac{<DV \cdot BEP_p>}{||DV|| \cdot ||BEP_p||} \text{ for } p = 1, 2, \ldots, 9 \quad (3.2.2-1)$$

$DV$ is defined by Equation (3.2.1-2). This Equation results in a value within interval $[0,1]$. Therefore, for $p=1, \ldots, 9$, there are nine different values representing the degree of correspondence of $FPBP$ vector to $BEP_p$ vectors. The maximum value for the cosine angle between vector $DV$ and edge code word space will be obtained if the vector $DV$ is completely matched with one of the edge code words. That means the maximum degree of edginess $1$ occurs if $Cos(\Phi_p)$ possesses the maximum value $1$. Therefore, the degree of edginess based on the defined angular correlation in Equation (3.2.2-1) is defined by Equation (3.2.2-2).

$$E_{fac} = \max_p \{Cos(\Phi_p)\}, \text{ for } p = 1, 2, \ldots, 9 \quad (3.2.2-2)$$
3.3 Tunable Edge Detector Based on FQCW-ECF

The edge detector based on FQCW-ECF can be designed using a fuzzy logic controller to extract the edge as desired. An edge detecting algorithm based on fuzzy logic with flexible rules as well as tunable membership functions would be an effective approach in edge detection. The Figure 35 illustrates a general idea of this edge detecting algorithm using the fuzzy logic controller for block processing of an image in order to detect the edge.

![Diagram of Edge Detection Using Fuzzy Logic Controller](image)

Figure 35. Illustration of Edge Detection Using Fuzzy Logic Controller.

The input is a processing block pattern (QCW intensity pattern of block B) denoted by vector PBP. This pattern is fuzzified by fuzzy logic controller to generate the FQCW intensity pattern, FPBP vector, as an input to fuzzy edge detector. While this fuzzy detector is defuzzifying the fuzzy vector FPBP to a crisp value, it is also generating the maximum degree of correspondence of pattern PBP to the binary edge patterns BEP or degree of edginess of the middle pixel (m,n) centered on the block B. The feedback is applied to tune the membership functions for gaining the appropriate membership degree $\mu d_k$ for $d_k$ (for $k=1,2,3,4$) based on the result $E$ in output. The
design of an edge detector based on FQCW-ECF using fuzzy logic controller is explained in detail further in Chapter IV.

**Edge detection algorithm based on FQCW-ECF algorithm** can be summarized as follows:

1. Choose a proper size of block $FB$, $W \times W$, with the middle fuzzy singleton indicating the intensity pixel $FX_{mn}$.

2. Do the $FQCW$ within block $B$ using the diagonal, cross, or/and other desired configurations (based on desired pixels neighboring the middle pixel), to find four child windows $FB_1$, $FB_2$, $FB_3$, $FB_4$.

3. Choose an appropriate aggregation operator $g$ (averaging operator) and apply it over each child window to find four representative intensities $FY_1$, $FY_2$, $FY_3$, $FY_4$. It can be done by Equations (3.1.2-1), (3.1.2-2), (3.1.2-3). In case of noisy image, replace the fuzzy representative intensity pixel $FX_{mn}$ with $FX'_{mn}$ based on Equations (3.1.2-4) and (3.1.2-5).

4. Construct the fuzzy processing block pattern vector $FPBP=[\mu d_1/d_1, \mu d_2/d_2, \mu d_3/d_3, \mu d_4/d_4]$ using Equations (3.1.3-1), (3.1.3-2), (3.1.3-3).

5. Choose or derive a desired $t$-norm $T_{qcw}$ and $t$-conorm $S_{qcw}$ using Tables 2 and 10.

6. Apply the JK-type generalized FQCW-ECF, distance or angular correlation given respectively by Equations (3.1.4-2 or 3.1.4-5), (3.2.1-3), and (3.2.2-2) for each individual $FQCW$ configuration (diagonally, crossly or any desired morphological structure), to find $E = E_{qcw}(FPBP) = E_{qcw}(\mu d_1/d_1, \mu d_2/d_2, \mu d_3/d_3, \mu d_4/d_4)$ the degree of
correspondence of vector $FPBP$ to the binary edge pattern space $BEP$ given by Table 8 (or degree of edginess of $FPBP$ at middle pixel $(m,n)$). Since we applied the normalized gray scale indicating intensity within continuous interval $[0,1]$. Therefore, this degree of correspondence directly represents the intensity of edge pixel (if thresholding is not applied) and is replaced instead of middle pixel intensity $X_{mn}$.

7. If there are more than one configuration, combine the result of each pair of configurations obtained from step 6 using a proper t-conorm $S_{\text{fcom}}$, to find the $E_{\text{fcombined}} = S_{\text{fcom}}(E_{\text{fcross}}, E_{\text{fcross}})$.

8. Apply tuning of membership function to adjust $\mu d_1, \mu d_2, \mu d_3, \mu d_4$ till to obtain the desired output in edge detector (part 6 or 7), that is optimization of edge detection.
CHAPTER IV

SIMULATION AND MODELING

Generally speaking, there are the three facets to engineering systems: (1) input(s); (2) output(s); and (3) system itself. If input and output of system are known, then creating the system is a design problem. In a simple system design, the important factor is a precise mathematical description by which the relation between input and output is defined for that system. For modularity, a complex system consists of a set of sub-systems, each with their own input/output characteristics. However, each sub-system can be modeled by its own characteristic as a module to finally build up the main complex system model. Once modeling is completed, simulation can be done using computer software to demonstrate or to enhance the system performance, sensitivity and optimization.

4.1 Optimal Edge Detector System Using Fuzzy Logic System

In preceding chapter we attempted to find an appropriate characteristic function for edge extraction within a block B of an image. we now apply the obtained edge characteristic function ($ECF$) as a mathematical description of edge detector system. We are going to design a tunable edge detector system ($EDS$) which is controllable by a fuzzy logic controller system ($FLC$). The optimal edge is detected by means of multi-dimensional optimization module ($MOM$) which properly tunes the
membership functions of input/output variables of FLC. Fuzzy image processing module F-IPM, the controller FLC, and the optimizer system MOM are the fuzzy part of our EDS system. F-IPM provides the FLC and MOM with the necessary image data. The inter-module communication of EDS is represented in Figure 36.

Figure 36. Optimal Edge Detector System (EDS) Using FLC.
4.1.1 Introducing *EDS* Modules

The designed and simulated *EDS* consists of 18 modules. *EDS* has been simulated according to the fuzzy and non-fuzzy algorithms which have been discussed in preceding chapters. Once modeling of *EDS* is completed and the tasks of *EDS* and its associated modules are specified, then the *EDS* can be simulated by individual simulation of each module and finally linking all modules together in order to obtain the desired task of the *EDS*. Here the simulation is done by *MATLAB* image processing and fuzzy logic tool box version 5.1. The simulation is capable of executing all defined algorithms simultaneously and making a comparison between algorithms. The simulated *EDS* modules are as follows:

1. **IMCM**: Physical image to matrix conversion module. This module converts the given color or monochrome images to an intensity (monochrome) image matrix. Each component of this matrix represents the intensity of respective pixel in a physical image and has a value within interval [0,1] which is considered as the normalization of set \{0,1,2,...L-1\} with \(L\) gray levels.

2. **MM**: Main module initializes parameters and provides interfacing or linking to the other modules. Also, it provides the desired information for displaying (monitoring).

3. **IPM**: Image processing module consists of a fuzzy section *F-IPM* and non fuzzy section *NF-IPM*. This module is a base to provide other modules with the necessary data from the image matrix. This module applies the block processing method to properly modify the data for uses of the other modules (as you observe its
inter-communications with other modules in Figure 36). *NF-IPM* processes image
data to make a suitable data for implementing the non-fuzzy algorithms. The *F-IPM*
modifies image data to prepare the necessary data for implementing the fuzzy
algorithms.

4. **PFM**: Processing function module provides the non-fuzzy part of *EDS*
with the *ECF* in order to implement the non-fuzzy algorithms for edge detection.

5. **FLC**: Fuzzy logic controller is designed with the flexible fuzzy rules and
tunable input/output membership functions to enhance the optimal edge trace. *FLC*
consists of *fuzzification, FIM, FRB, CIF, defuzzification* and *MOM* modules.

6. **FUZZ**: This fuzzification module is used to convert crisp data $Y_1, Y_2, Y_3,
Y_4, X_{mn}$ (concerning *QCW*-pattern within block $B$ of an intensity image II)
respectively to the fuzzy singleton data $FY_1, FY_2, FY_3, FY_4, FX_{mn}$. The fuzzified data
are processed in *FIM* to make the decision for the output of *FLC*.

7. **FMFG**: Fuzzy membership function generator is used to produce the
desired functions used in fuzzy plane $P$ for fuzzification of image data. This module
is able to generate triangular and trapezoid shaped functions (section 1.3.3) over the
intensity interval $[0,1]$ or gray scale $\{0,1,\ldots L-1\}$ as Figure 7. The membership
functions are used in fuzzification of the input data to *FLC* and implicating the output
membership degrees of the fuzzy inference machine *FIM*.

8. **FRB**: Fuzzy rule base is the module where the knowledge concerning the
performance strategy of *FLC* is linguistically stored in the form of fuzzy rules.
9. **FIM:** Fuzzy Inference Module is used to implicate the membership degrees for output based on the fired fuzzy rules from the rule base. The input of **FIM** is fuzzy singletons $FY_1, FY_2, FY_3, FY_4, FX_{mn}$ and the implicated output based on fuzzy rules are fuzzy singletons $Fd_1, Fd_2, Fd_3, Fd_4$.

10. **CIM:** Contrast intensification module is used to intensify the membership degrees of fuzzy singletons in **FIM** outputs. The outputs of **FIM** are brought to this module for intensification of membership degrees by using the function $T_s$ (section 1.4.2).

11. **DEFUZZ:** This defuzzification module of **FLC** is applied to convert fuzzy singletons (intensified or not) $Fd_1, Fd_2, Fd_3, Fd_4$ to a crisp value $E_{mn}$ which also indicates the degree of edginess of the middle pixel $X_{mn}$. This module is capable of implementing the fuzzification according to **COA, MOM, COM**, and all defined $E_{fqcw}$ (defined in sections 1.3.7, 3.1.5, 3.2.1, 3.2.2).

12. **MOM:** Multidimensional optimization module which is applied to tune input/output membership function(s) of **FLC** in order to optimize the edge trace in output of **EDS**. Cyclic coordinate algorithm (**CYCLIC**) along with the golden ratio line search algorithm (**GOLDEN**) are used to minimize the performance index $PI=\|e\|^2$. The term $e$ is the error matrix between the desired edge matrix $I_{Ed}$ and the actual edge trace matrix $I_E$ in output of **EDS**. The vector $V$ is constructed containing the overlap percentages and peak points of input/output membership functions of **FLC** as the tuning variables. ‘$V$’ is the input of **MOM** and the output is $V^*$ indicating the optimal value for vector $V$, (section 1.5.5).
13. **I-Error**: Image error module consists of two modules NFI-Error and FI-Error which are used to determine the cumulative error, cumulative square error, matrix square error (SQE), mean error (ME), and matrix correlation (TDC) for all explained non-fuzzy and fuzzy algorithms of edge detection (sections 1.6.4, 1.6.5).

4.1.2 **EDS Task Descriptions**

Figure 37 illustrates a tunable edge detector system (EDS) which is able to find edge based on the explained fuzzy and non-fuzzy algorithms in the preceding chapters. The optimal edge is detectable through a certain optimization algorithm. A simple feedback system is used to take the output of EDS towards an optimal point. The following block diagram gives the general idea about the inter-communications between the fuzzy logic controller FLC, fuzzy image processing module F-IPM and multi-dimensional optimization module MOM.

![Figure 37. Inter-Communication of Fuzzy Part of EDS.](image-url)
Child window filtering is properly implemented by NF-IPM for the non-fuzzy edge detection algorithm. Child window processing results in four representatives denoted by \( Y_1, Y_2, Y_3, Y_4 \). These four intensity representatives, as well as the middle pixel intensity \( X_{mn} \), represent the QCW-pattern within block \( B \) and are also considered inputs to the FLC system. NF-IPM generates the proper QCW pattern(s) specified by its (their) patterning characteristic vector(s) \( PBP \) for the processing block \( B \) of the intensity image \( II \). The QCW pattern(s) \( (Y_1, Y_2, Y_3, Y_4, X_{mn}) \) within the block \( B \) and its relevant vector(s) \( PBP \) are the input of FLC. The FLC output is the degree of edginess relating to the middle intensity pixel \( X_{mn} \) of the block \( B \).

Multidimensional optimization based on the cyclic coordinate algorithm (explained in section 1.5.6) is adopted to minimize the desired error function as performance index \( PI \) (cost function) of EDS by tuning the input/output membership functions of FLC. This optimization is used just for fuzzy approaches to edge detection. Block processing technique is applied to find the edge pixels of the image by using QCW techniques which have been explained in chapter 2. As you observe in Figure 37, the error matrix \( e = I_{Ed} - I_E \) (difference between the desired edge trace \( I_{Ed} \) and actual edge trace \( I_E \)) is used to derive performance index, \( PI=\|e\|^2 \), as an objective function in the optimization procedure for conducting the EDS system towards an optimal state. The output of multidimensional optimization module \( MOM \) is the vector \( V \) which contains the tuning variables of the input/output membership function of FLC.

The non-fuzzy part of EDS can be simply illustrated as Figure 38. The
input of \textit{NF-IPM} is the intensity image $II$. After choosing proper QCW configuration(s) within block $B$, this module executes the suitable child window processing for the non-fuzzy algorithm and creates the QCW pattern(s) with its (their) relevant PBP as input to \textit{PFM}. \textit{PFM} which contains the functions relating to the non-fuzzy edge detection algorithms generates $E_{mn}$, the degree of edginess of the middle pixel intensity $X_{mn}$.

![Diagram](image)

Figure 38. Inter-Communication of Non-Fuzzy Part of \textit{EDS}.

Tasks of the optimal edge detector system (\textit{EDS}) depicted in Figure 36 can be summarized as follows:

1. Converting the physical monochrome or color image to an intensity (monochrome) image matrix $II$ with size $M \times N$ and $L$ gray levels, section 1.2.3. Done by the module \textit{IMCM}.

2. Constructing a movable block (window or mask) $B$ with the selective size of block $W \times W$ and middle intensity pixel $X_{mn}$. Done by the module \textit{IPM}.

3. Sliding the block $B$ over the given image pixel by pixel from top to bottom to process all regions of the given image, section 1.2.6. Done by the module \textit{IPM}.
4. Selecting and constructing the suitable QCW configuration(s) (cross, diagonal and combination of both), section 2.1.3. Done by module IPM.

5. Child window processing (filtering) using median, min, max, mean operators based on the inquiry of fuzzy or non-fuzzy algorithms to find representative pixels \(Y_1, Y_2, Y_3, Y_4\), section 2.1.4. Done by the module IPM.

6. Constructing the processing block pattern vector \(PBP = [d_1, d_2, d_3, d_4]\) which is defined as the patterning characteristic of QCW within the processing block \(B\), Equation 2.1.5-7. Done by the module IPM.

7. Constructing the code words \(BEP_p\) for 9 classes of the defined binary edge patterns, Table 8. Done by the module IPM.

8. Detecting edge based on non-fuzzy algorithms within the processing block \(B\), and individually storing the edge trace matrix \(I_E\) resulting from each algorithm, sections 1.2.10 (derivative methods), 2.1.6, 2.1.7, 2.2.1, 2.2.2 (QCW-ECF algorithms). Done by PFM.

9. Generating tunable membership functions, trapezoid and triangular shaped, over the gray level \(\{0,1,2,...,L-1\}\) or intensity interval \([0,1]\), sections 1.3.3 and 1.3.8. Done by the module FMFG.

10. Fuzzification of QCW pattern (crisp values \(Y_1, Y_2, Y_3, Y_4, X_{mn}\)) within the block \(B\) in order to obtain \(FQCW\) pattern (fuzzy singletons \(FY_1, FY_2, FY_3, FY_4, FX_{mn}\)), section 3.1.2. Done by the fuzzification module in FLC.

11. Storing information or knowledge pertaining to the desired performance of FLC in the form of linguistic fuzzy rules, IF-THEN, in fuzzy rule base FRB.
12. Applying *Max-Min* fuzzy reasoning, to implicate the membership degrees of *FIM* outputs based on the fired control rules from *FRB*, done by the module *FIM*, (section 1.3.6). The inputs to *FIM* are fuzzy singletons $FY_1$, $FY_2$, $FY_3$, $FY_4$, $FX_{mn}$ defining *FQCW* pattern. The implicated outputs are fuzzy singletons $Fd_1$, $Fd_2$, $Fd_3$, $Fd_4$, which are components of vector *FPBP*, section 3.1.3.

13. Defuzzifying the fuzzy vector $FPBP = [Fd_1, Fd_2, Fd_3, Fd_4]$ based on defuzzification methods *COA, MOM, COM* or *FQCW-ECF, E_{qvw},* section 3.1.4 (special cases of *QCW* edge characteristic functions *JK-type1,2,3,4,5*). The value $E_{mn}$ indicates the degree of edginess of vector $FPBP$ at point $(m,n)$ and is replaced instead of middle intensity value $X_{mn}$ in processing block. Done by the defuzzification module.

14. Determining the error matrix $e$ which is deference between desired edge trace $I_{Ed}$ and the actual edge trace matrix $I_E$ in output of *EDS* (for normalization of components, each edge trace matrix before calculation of error is divided by its maximum component). Also, calculating the commutative error and commutative square error in each middle pixel while block $B$ is sliding over the image. This is computed simultaneously for all running algorithm during the process, section 1.6.4. Done by the *FI-Error* and the *NFI-Error* modules.

15. Determining the performance index (cost function) $PI = ||e||^2$. This is used as an objective function to be minimized by *MOM* in order to optimize the *EDS* system, section 1.5-6. Done by $PI_{left}$, $PI_{right}$ modules.
16. In case of optimization, optimizing the edge trace in EDS output based on minimizing the $PI=\|e\|^2$ as the objective function. The optimization is done by tuning the input/output membership functions of FLC. The overlap percentages and peak points (the components of vector $V$) of the membership functions are used as the tuning variables. The cyclic coordinate algorithm is applied to derive the result of defuzzification ($E_{mn} = E_{fqcw}$) in the FLC output toward an optimal value ($E^*_{mn} = E^*_{fqcw}$) by tuning $\mu d_1$, $\mu d_2$, $\mu d_3$, $\mu d_4$, section 1.5.6. Done by the MOM module.

4.1.3 Experimental Results

Due to different variation of intensities and also variety of objects from one image to the another one, an efficient edge detection algorithm must be such powerful to possess that flexibility to produce the proper edge traces for any type of images, disregards to the degree of smoothness, sharpness, noisiness, and pixel to pixel intensity transition. Also, it must generate the appropriate edge traces independent to the size of the objects and satisfactorily results in a same quality for small and big objects. Toward this purpose, To investigate the efficiency of the aforementioned algorithms, we find the experimental results through three examples over three types of $M \times M$ images as: (1) an image illustrating an small object in example one; (2) an image given in example two representing the similar foreground and background intensities (in some regions), smoothness, and various intensity transition in different regions; and (3) an image covered by Gussian and Salt & Pepper noises.

To judge about the obtained results, we compare the $M \times M$ edge trace matrix
resulted from each algorithm (without thresholding), denoted by $IE$, with a $M \times M$ ideal binary edge trace matrix, denoted by $Id$. For this purpose, we apply three kinds of measurements as the performance indexes as; (1) mean error, $ME$, and total mean error $TME$ defined by Equations (1.6.5-2) and (1.6.5-3); (2) square error, $SQE$, defined by Equation (1.6.5-4), but multiplied by $100/(M \times M - NIEP)$ such that $NIEP$ is the number of ideal edge pixels; and (3) two dimensional correlation, $TDC$, given by Equation (1.6.6-1), where $A = IE$ and $B = Id$ in all three cases.

The outputs, edge trace matrices $IE1$, $IE2$, ..., $IE21$, have been obtained without using thresholding technique. These matrices are respectively the results of the algorithm mentioned in the tables given in examples 1, 2, 3. The $IE1$, $IE2$, and $IE3$ matrices are results of non-linear derivative algorithms. The $IE4$, $IE5$, and $IE6$ matrices represent the results from simple intensity difference (classic) methods explained in section 1.4.4. The $IE7$, ..., $IE17$ matrices are resulted from some special cases of generalized $QCW-ECF$. The $IE18$ matrix is the optimal edge trace based on $FQCW-ECF$ (JK-type1) generated by fuzzy logic system and optimized by the cyclic coordinate algorithm. The $IE19$, $IE21$ are also edge matrices based on $FQCW-ECF$ but not optimized. They are defuzzified in $FLC$ respectively using $COA$ and $MOM$ methods. To obtain $IE20$, the membership degrees $\mu_d1$, $\mu_d2$, $\mu_d3$, $\mu_d4$ resulting from $FLC$ are intensified 7 times ($s=7$) by means of recursive Equation (1.4.2-2) before applying $COA$ defuzzification method.

For $QCW$ and $FQCW$ within a block of $5 \times 5$, the morphological structure given in Figure 39 is applied. Also, for diagonal and cross $QCW$ configurations (both
5×5 and 3×3), the structures represented in Figures 22 and 23 are applied.

![Diagram of structures](image)

Figure 39. One Possible (F)QCW Configuration Within a 5×5 Block B.

In case of noisy images in example 3, to remove the noises, the median filtering is applied within each child window $B_k$. Averaging operation is also used within the region $B-\{(m,n)\}$ and the result of this averaging (Equation 1.2.5-1) $X'_{mn}$ is considered as the representative of entire block $B$. That is, $X_{mn} = X'_{mn}$.

The fuzzy logic rule base designed for edge detection algorithm contains the rules given by Figure 40.

![Fuzzy Logic Rule Base](image)

Figure 40. Fuzzy Rules Applied in FLC for Fuzzy Algorithms in Examples 1,2,3.
To find the optimal edge trace using fuzzy logic, we use cyclic coordinate algorithm to tune input/output membership functions of FLC in order to obtain the appropriate $\mu_{d1}$, $\mu_{d2}$, $\mu_{d3}$, $\mu_{d4}$. In this optimization, tuning the membership functions are done by adjusting the percentage of overlaps and peak points.

We consider one fixed percentage of overlap for input MFs, denoted by $P_{ovlin}$, and another fixed value for output MFs, denoted by $P_{ovlout}$. Five peak points, denoted by $C_{in1}$, $C_{in2}$, $C_{in3}$, $C_{in4}$, $C_{in5}$, for input MFs and five peak points, denoted by $C_{out1}$, $C_{out2}$, $C_{out3}$, $C_{out4}$, $C_{out5}$, for outputs are also considered along with mentioned percentage of overlap to adjust the MFs such that the performance index $SQE = \frac{100}{(M.M-NIEP)\|Id - \hat{I}\|^2}$ is minimized. The vector $V=[ P_{ovlin}, C_{in1}, C_{in2}, C_{in3}, C_{in4}, C_{in5}, P_{ovlout}, C_{out1}, C_{out2}, C_{out3}, C_{out4}, C_{out5}]$ is considered for this multidimensional optimization using cyclic coordinate algorithm explained in 1.5.5 and 1.5.6.

Cyclic coordinate algorithm finds the optimal components for the vector $V$ (containing the input/output MFs of FLC) by minimizing the $SQE$. That is, $V_{initial} \rightarrow V^{*_{optimal}}$.

For all three examples, the input and output membership functions of FLC are initialized symmetrically before optimization meaning that the vector $V$ is initialized as $V_{initial}=[0.5, 0.167, 0.333, 0.5, 0.667, 0.833, 0.5, 0.167, 0.333, 0.5, 0.667, 0.833]$. 
**Example 1.** To represent the efficiency of QCW-ECF and its fuzzy associated FQCW-ECF against the other derivatives algorithm for small object edge detection, we have provided the first experiment as follows. A 8 × 8 intensity image matrix \( I_{l1} \) \((L=16)\) containing a small size object (ring) has been applied as an input to the EDS. The data and figures resulted from EDS are given in Table 11 and Figures 41 to 46.

**Table 11**

Comparison Table for Fuzzy and Non-Fuzzy Algorithms in Example 1

<table>
<thead>
<tr>
<th>Trace</th>
<th>Methods</th>
<th>Eq.</th>
<th>Block size</th>
<th>(F)QCW structure Fig.</th>
<th>Defuzzification</th>
<th>TME</th>
<th>SQE %</th>
<th>TDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE1</td>
<td>Roberts</td>
<td>1.2.10-1</td>
<td>2x2</td>
<td></td>
<td>0.082</td>
<td>4.765</td>
<td>0.858</td>
<td></td>
</tr>
<tr>
<td>IE2</td>
<td>Sobel</td>
<td>1.2.10-2</td>
<td>3x3</td>
<td></td>
<td>0.092</td>
<td>5.495</td>
<td>0.836</td>
<td></td>
</tr>
<tr>
<td>IE3</td>
<td>Wallis</td>
<td>1.2.10-5</td>
<td>3x3</td>
<td></td>
<td>0.063</td>
<td>4.282</td>
<td>0.884</td>
<td></td>
</tr>
<tr>
<td>IE4</td>
<td>Classic</td>
<td>1.4.4-1</td>
<td>5x5</td>
<td></td>
<td>0.250</td>
<td>30.76</td>
<td>-124</td>
<td></td>
</tr>
<tr>
<td>IE5</td>
<td>Classic</td>
<td>1.4.4-2</td>
<td>5x5</td>
<td></td>
<td>0.026</td>
<td>1.335</td>
<td>0.967</td>
<td></td>
</tr>
<tr>
<td>IE6</td>
<td>Classic</td>
<td>1.4.4-3</td>
<td>5x5</td>
<td></td>
<td>0.063</td>
<td>7.692</td>
<td>0.832</td>
<td></td>
</tr>
<tr>
<td>IE7</td>
<td>Angular Corr.</td>
<td>2.2.2-1</td>
<td>5x5</td>
<td>39</td>
<td>0.064</td>
<td>7.408</td>
<td>0.833</td>
<td></td>
</tr>
<tr>
<td>IE8</td>
<td>Distance Corr.</td>
<td>2.2.2-2</td>
<td>5x5</td>
<td>39</td>
<td>0.012</td>
<td>0.077</td>
<td>0.998</td>
<td></td>
</tr>
<tr>
<td>IE9</td>
<td>JK-type1</td>
<td>2.1.7-7</td>
<td>5x5</td>
<td>39</td>
<td>0.018</td>
<td>0.654</td>
<td>0.984</td>
<td></td>
</tr>
<tr>
<td>IE10</td>
<td>JK-type2</td>
<td>2.1.7-14</td>
<td>5x5</td>
<td>39</td>
<td>0.013</td>
<td>0.129</td>
<td>0.997</td>
<td></td>
</tr>
<tr>
<td>IE11</td>
<td>JK-type3</td>
<td>2.1.7-21</td>
<td>5x5</td>
<td>39</td>
<td>0.013</td>
<td>0.129</td>
<td>0.997</td>
<td></td>
</tr>
<tr>
<td>IE12</td>
<td>JK-type4</td>
<td>2.1.7-32</td>
<td>5x5</td>
<td>39</td>
<td>0.025</td>
<td>0.328</td>
<td>0.993</td>
<td></td>
</tr>
<tr>
<td>IE13</td>
<td>JK-type5</td>
<td>2.1.7-43</td>
<td>5x5</td>
<td>39</td>
<td>0.019</td>
<td>0.197</td>
<td>0.995</td>
<td></td>
</tr>
<tr>
<td>IE14</td>
<td>JK-type1</td>
<td>2.1.7-10</td>
<td>5x5</td>
<td>23 (b)</td>
<td>0.005</td>
<td>0.053</td>
<td>0.999</td>
<td></td>
</tr>
<tr>
<td>IE15</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>3x3</td>
<td>22 (c)</td>
<td>0.036</td>
<td>0.617</td>
<td>0.936</td>
<td></td>
</tr>
<tr>
<td>IE16</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5x5</td>
<td>23 (e)</td>
<td>0.018</td>
<td>0.654</td>
<td>0.984</td>
<td></td>
</tr>
<tr>
<td>IE17</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5x5</td>
<td>23 (d)</td>
<td>0.018</td>
<td>0.654</td>
<td>0.984</td>
<td></td>
</tr>
<tr>
<td>IE18</td>
<td>JK-type1(Fuzzy)</td>
<td>3.1.4-9</td>
<td>5x5</td>
<td>39</td>
<td>JK-type1</td>
<td>0.000</td>
<td>0.000</td>
<td>1.000</td>
</tr>
<tr>
<td>IE19</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5x5</td>
<td>39</td>
<td>COA</td>
<td>0.034</td>
<td>0.706</td>
<td>0.984</td>
</tr>
<tr>
<td>IE20</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5x5</td>
<td>39</td>
<td>COA, (INT)</td>
<td>0.022</td>
<td>0.369</td>
<td>0.988</td>
</tr>
<tr>
<td>IE21</td>
<td>Fuzzy model</td>
<td>1.3.7-4</td>
<td>5x5</td>
<td>39</td>
<td>MOM</td>
<td>0.034</td>
<td>0.706</td>
<td>0.984</td>
</tr>
</tbody>
</table>
Figure 41. Optimized Input/Output MF of FLC for Example 1 Obtained by MOM.
Figure 42. Illustration of Edge Trace Matrices Generated by EDS in Example 1.
Figure 43. Illustration of Edge Trace Matrices Generated by EDS in Example 1.
Figure 44. Illustration of Edge Intensity Surfaces for Comparison of Intensity Levels at Edge Points Between the Ideal Edge Trace and the Edge Traces Resulting From Some Algorithms in Example 1.
SQE LEVELS
Illustration of SQE level at each iteration during the optimization of edge trace IE18 (fuzzy version)

![SQE Levels Graph](image)

Figure 45. Illustration of SQE Trajectory (SQE Minimization) During the Optimization of Edge Trace IE18 Using Cyclic Coordinate Algorithm to Tune the Input/Output Membership Functions of FLC Illustrated in Figure 41 (a), (b).

Mean Error
Illustration of mean error at each pixel

![Mean Error Graph](image)

Figure 46. Illustration of Mean Error at Each Pixel for Fuzzy and Some Non-Fuzzy Algorithms Obtained by EDS in Example 1.
As observed from obtained edge trace matrices (Figures 42 and 43) and the illustrated edge intensity surfaces (Figure 44), obviously, in small images the derivative methods (Sobel, Roberts, and Wallis) creates the scattered data at edge points. This makes some difficulties in choosing an appropriate threshold levels $T_L$ and $T_H$ ($T_L$ and $T_H$ are illustrated in Figure 32), since the lower level and upper level thresholds can not be distinguished. In Roberts $IE1$ and Sobel $IE2$, some edge points have the less intensity than some other points which are not supposed to be edge according to the ideal edge trace $Id$. That is, it is not possible to find $T_L$ and $T_H$ such that $T_L \leq T_H$. In Wallis edge trace $IE3$, the $T_L$ and $T_H$ are distinguishable, but difference between $T_L$ and $T_H$ is small and it causes the selection range of threshold levels to be so limited. On the contrary, the $QCW$ (specially $FQCW$ which is not dependent on thresholding) does not possess the above mentioned problem. All data concerning the edge points generated by the algorithms of $QCW$-$ECF$ family are organized and not scattered. Consequently, distinguishing the appropriate lower level and upper level thresholds ($T_L$ and $T_H$) is easy since we can easily find $T_H$ and $T_L$ such that $T_L \leq T_H$ and $T_H - T_L \gg 0$.

Also a quick review on the Table 11 proves that generally the algorithms of $QCW$ and specially $FQCW$ families have three advantages in performances compare to the other derivative algorithms for small object edge detection as: (1) bigger correlation $TDC$ (close to 1); (2) less total mean error $TME$ (even mean error $ME$ at each pixel, illustrated in Figure 46); and finally (3) less square error $SQE$. 
One significant result, \( IE18 \), has been obtained by \( FQCW-ECF \) (JK-type1) (fuzzy approach to edge detection) and optimized by cyclic coordinate algorithm. The optimal edge trace, \( IE18 \), generated by \( EDS \) without using threshold technique has the best correlation to the ideal edge trace \( Id \) and the least \( TME \) and \( SQE \) comparing with the non-fuzzy algorithms.

The general conclusion of this example is that, for small object edge detection, the algorithms of \( QCW-ECF \) family (which are based on correlation or degree of correspondence to the binary edge patterns) performs better than the derivative algorithms such as Roberts, Sobel, and Wallis. They are all dependent on thresholding technique but implementation of thresholding for \( QCW-ECF \) family is much easier than the other compared algorithms. At the end, \( FQCW-ECF \) generates the better results without using threshold technique comparing to its non-fuzzy associates \( QCW-ECF \) and the other mentioned non-fuzzy algorithms.

**Example 2.** To represent the efficiency of \( QCW-ECF \) and its fuzzy associated \( FQCW-ECF \) against the other derivative algorithms for edge detection of an image with a similar foreground and background intensity, we have provided the second experiment as follows. A \( 130\times130 \) intensity image matrix \( II2 \) (\( L=256 \)) has been applied as an input to the \( EDS \) system for edge detection without using thresholding technique. Some data and graphs resulted from \( EDS \) are given in Table 12 and Figures 47 to 51.
Table 12

Comparison Table for Fuzzy and Non-Fuzzy Algorithms Applied for 130×130 Pixels Image II2 in Example 2, Illustrated in Figure 48

<table>
<thead>
<tr>
<th>Trace</th>
<th>Methods</th>
<th>Eq.</th>
<th>Block size</th>
<th>(F) QCW structure</th>
<th>Defuzzification</th>
<th>TME</th>
<th>SQE %</th>
<th>TDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE1</td>
<td>Roberts</td>
<td>1.2.10-1</td>
<td>2×2</td>
<td></td>
<td></td>
<td>0.273</td>
<td>9.039</td>
<td>0.186</td>
</tr>
<tr>
<td>IE2</td>
<td>Sobel</td>
<td>1.2.10-2</td>
<td>3×3</td>
<td></td>
<td></td>
<td>0.171</td>
<td>5.388</td>
<td>0.276</td>
</tr>
<tr>
<td>IE3</td>
<td>Wallis</td>
<td>1.2.10-5</td>
<td>3×3</td>
<td></td>
<td></td>
<td>0.117</td>
<td>3.873</td>
<td>0.128</td>
</tr>
<tr>
<td>IE4</td>
<td>Classic</td>
<td>1.4.4-1</td>
<td>5×5</td>
<td></td>
<td></td>
<td>0.152</td>
<td>3.642</td>
<td>0.402</td>
</tr>
<tr>
<td>IE5</td>
<td>Classic</td>
<td>1.4.4-2</td>
<td>5×5</td>
<td></td>
<td></td>
<td>0.227</td>
<td>9.186</td>
<td>-0.124</td>
</tr>
<tr>
<td>IE6</td>
<td>Classic</td>
<td>1.4.4-3</td>
<td>5×5</td>
<td></td>
<td></td>
<td>0.259</td>
<td>10.14</td>
<td>0.322</td>
</tr>
<tr>
<td>IE7</td>
<td>Angular Corr.</td>
<td>2.2.2-1</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.121</td>
<td>3.391</td>
<td>0.372</td>
</tr>
<tr>
<td>IE8</td>
<td>Distance Corr.</td>
<td>2.2.2-2</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.114</td>
<td>3.401</td>
<td>0.385</td>
</tr>
<tr>
<td>Q</td>
<td>JK-type1</td>
<td>2.1.7-7</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.147</td>
<td>4.216</td>
<td>0.430</td>
</tr>
<tr>
<td>IE10</td>
<td>JK-type2</td>
<td>2.1.7-14</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.117</td>
<td>3.407</td>
<td>0.413</td>
</tr>
<tr>
<td>IE11</td>
<td>JK-type3</td>
<td>2.1.7-21</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.117</td>
<td>3.407</td>
<td>0.413</td>
</tr>
<tr>
<td>IE12</td>
<td>JK-type4</td>
<td>2.1.7-32</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.113</td>
<td>3.473</td>
<td>0.369</td>
</tr>
<tr>
<td>IE13</td>
<td>JK-type5</td>
<td>2.1.7-43</td>
<td>5×5, 39</td>
<td></td>
<td></td>
<td>0.112</td>
<td>3.411</td>
<td>0.381</td>
</tr>
<tr>
<td>IE14</td>
<td>JK-type1</td>
<td>2.1.7-10</td>
<td>5×5, 23 (b)</td>
<td></td>
<td></td>
<td>0.160</td>
<td>4.842</td>
<td>0.442</td>
</tr>
<tr>
<td>IE15</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>3×3, 22 (c)</td>
<td></td>
<td></td>
<td>0.163</td>
<td>4.863</td>
<td>0.435</td>
</tr>
<tr>
<td>IE16</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5×5, 23 (e)</td>
<td></td>
<td></td>
<td>0.164</td>
<td>4.958</td>
<td>0.441</td>
</tr>
<tr>
<td>IE17</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5×5, 23 (d)</td>
<td></td>
<td></td>
<td>0.164</td>
<td>4.956</td>
<td>0.444</td>
</tr>
<tr>
<td>F</td>
<td>JK-type1 (Fuzzy)</td>
<td>3.1.4-9</td>
<td>5×5, 39</td>
<td>JK-type1</td>
<td></td>
<td>0.106</td>
<td>3.172</td>
<td>0.493</td>
</tr>
<tr>
<td>Q</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5×5, 39</td>
<td>COA</td>
<td></td>
<td>0.147</td>
<td>4.313</td>
<td>0.321</td>
</tr>
<tr>
<td>C</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5×5, 39</td>
<td>COA (INT)</td>
<td></td>
<td>0.140</td>
<td>4.297</td>
<td>0.335</td>
</tr>
<tr>
<td>W</td>
<td>Fuzzy model</td>
<td>1.3.7-4</td>
<td>5×5, 39</td>
<td>MOM</td>
<td></td>
<td>0.156</td>
<td>4.557</td>
<td>0.434</td>
</tr>
</tbody>
</table>
V_{optimal}^* = [.5135, .294, .333, .4213, .5393, .706, .3989, .1667, .333, .5150, .7825, .8722]

Figure 47. Optimized Input/Output Membership Functions of FLC for Example 2.
Figure 48. Illustration of Edge of Image II2 Generated by EDS in Example 2.
Figure 49. Illustration of Edge of Image II2 Generated by EDS for Example 2.
**Figure 50.** Illustration of *SQE* Trajectory (*SQE* Minimization) During the Optimization of Edge of Sample Image Using Cyclic Coordinate Algorithm to Tune the Input/Output Membership Functions of *FLC* Illustrated in Figure 47 (a), (b), for Example 2.

**Figure 51.** Illustration of Mean Error at Each Pixel for Some (Fuzzy and Non-Fuzzy) Algorithms for Edge Detection of Image II2 in Example 2.
The image II2 displays a perspective of a building with different intensity of regions. All algorithms applied in example 1 have been reused to detect the edge of image II2. Table 12 contains the data resulting this experiment and helps to analyze the edge trace images IE1,...,IE21 and the efficiencies of the given algorithms. Edge trace Id is a desired binary images (ideal) chosen as an comparison base for the other obtained traces.

All QCW-ECF result in edge traces with a (a) less total mean error TME (even pixel by pixel mean error ME as illustrated in Figure 51), (b) less mean square error SQE, and (c) bigger correlation, comparing to the Sobel and Roberts algorithms. Comparing to the QCW-ECF, Wallis algorithm produces an edge trace IE3 with the competitive TME and SQE, but the low correlation TDC. As observed in Figure 48, the Wallis algorithm results in a low intensity edge trace IE3. The edge trace IE16 resulting from JK-type1 possesses the maximum TME= 0.164 and SQE= 4.95 among the other QCW-ECF, but it has a high correlation TDC= 0.441 to ideal edge trace Id. The edge trace IE16 is illustrated in Figure 49 and obtained by combining of diagonal and cross QCW configuration (illustrated in Figure 23 (e)) within a block of 5x5 using the Equation (2.1.7-11). JK-type1 generates a higher intensity edge among the other types, JK-type2,3,4 of QCW-ECF. Also, edge trace IE13 resulting from JK-type5 possesses the minimum TME= 0.112 and competitive SQE= 3.401 among the other QCW-ECF but has a less correlation TDC= 0.441 to ideal edge trace Id. The edge trace IE13 is illustrated in Figure 49 and obtained by QCW configuration
(illustrated in Figure 39 (a)) within a block of 5x5 using the Equation (2.1.7-42). JK-type5 generates a lower intensity edges among the other types of QCW-ECF.

As you see all above algorithms do not have enough sensitivity to detect the edges of an object which has similar intensity with background. Fuzzy logic algorithm with flexible rules and tunable membership functions (in inputs/output of FLC) is a significant approach in this regard. As a part of experiment 2, we apply the JK-type1 of the proposed FQCW-ECF to remove the above mentioned problem.

Edge traces IE18, IE19, IE20, and IE21 are resulted from fuzzy logic algorithms for edge detection. IE18 is resulted from JK-type1 of FQCW-ECF (given by Equation (3.1.4-9)) which functions as an edge characteristic function and the defuzzificator in FLC. IE18, IE19 are the edge traces obtained respectively by COA and MOM defuzzification methods in FLC. IE20 is an edge trace resulted from COA defuzzification method in which the membership degrees µd1, µd2, µd3, µd4 (output of inference machine of FLC) are intensified (s= 7 times) before defuzzification. IE21 resulted from MOM defuzzification possesses bigger TME, SQE, and even higher TDC comparing to those of IE20 and IE19. The edge trace IE21 resulted from fuzzy algorithm of JK-type1 in which the membership functions (MF) of FLC have been trained in advance. The MFs are trained such a way to make the fuzzy edge detector system sensitive to those part of image which have similar intensity in foreground and background (like roof of the building in image II2). For this purpose, a 10x10 block of image is cut from top of the image II2 where the roof of the building has very similar intensity to the background (both dark gray DG). Then, like example 1, we
tune the membership functions using cyclic coordinate algorithm regarding the minimization of performance index $SQE$ (for $10 \times 10$ image sample illustrated in Figure 50). The optimal membership function resulted from this optimization using cyclic coordinate algorithm is depicted in Figures (a) and (b) of Figure 47. These membership functions tuned optimized for a desired part of the image $II2$ have been applied for edge detection of entire images. This is illustrated as edge trace $IE18$ in Figure 49. As seen in this edge trace, the edges of the roof of the building are appeared, but in other edge traces which the $MF$ are not trained the edges are not detected. We can tune the membership functions to obtain the optimal edge trace for whole image $II2$. But, this example shows the flexibility of fuzzy algorithm for edge detection of desired part of an image.

**Example 3.** In third example, we attempt to investigate the sensitivity of aforementioned edge detection methods against two types of noises, Gaussian and Salt & Pepper covering all over an image. We have provided the third experiment in two sections as follows.

**Example 3a.** A $130 \times 130$ intensity image matrix $NI3a$ ($L=256$), covered by Gaussian noise with zero mean and 0.005 variance, have been applied as an input to the $EDS$ system for edge detection without using thresholding technique. Data obtained from this experiment are given in Table 13. Some relevant images and graphs are also represented in Figures 52 to 55.
Table 13

Comparison Table for Fuzzy and Non-Fuzzy Algorithms Applied for 130x130 Noisy Image (Gussian Noise $\sigma^2=0.005$) N/3a

<table>
<thead>
<tr>
<th>Trace</th>
<th>Methods</th>
<th>Eq.</th>
<th>Block size</th>
<th>(F)QCW structure Fig.</th>
<th>Defuzzification</th>
<th>TME</th>
<th>SQE %</th>
<th>TDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE1</td>
<td>Roberts</td>
<td>1.2.10-1</td>
<td>2x2</td>
<td></td>
<td>0.271</td>
<td>8.913</td>
<td>0.135</td>
<td></td>
</tr>
<tr>
<td>IE2</td>
<td>Sobel</td>
<td>1.2.10-2</td>
<td>3x3</td>
<td></td>
<td>0.208</td>
<td>6.277</td>
<td>0.229</td>
<td></td>
</tr>
<tr>
<td>IE3</td>
<td>Wallis</td>
<td>1.2.10-5</td>
<td>3x3</td>
<td></td>
<td>0.121</td>
<td>3.502</td>
<td>-0.035</td>
<td></td>
</tr>
<tr>
<td>IE4</td>
<td>Classic</td>
<td>1.4.4-1</td>
<td>5x5</td>
<td></td>
<td>0.295</td>
<td>10.58</td>
<td>0.329</td>
<td></td>
</tr>
<tr>
<td>IE5</td>
<td>Classic</td>
<td>1.4.4-2</td>
<td>5x5</td>
<td></td>
<td>0.315</td>
<td>12.45</td>
<td>0.124</td>
<td></td>
</tr>
<tr>
<td>IE6</td>
<td>Classic</td>
<td>1.4.4-3</td>
<td>5x5</td>
<td></td>
<td>0.387</td>
<td>18.14</td>
<td>0.247</td>
<td></td>
</tr>
<tr>
<td>IE7</td>
<td>Angular Corr.</td>
<td>2.2.2-1</td>
<td>5x5</td>
<td>39</td>
<td>0.126</td>
<td>3.492</td>
<td>0.230</td>
<td></td>
</tr>
<tr>
<td>IE8</td>
<td>Distance Corr.</td>
<td>2.2.2-2</td>
<td>5x5</td>
<td>39</td>
<td>0.119</td>
<td>3.580</td>
<td>0.261</td>
<td></td>
</tr>
<tr>
<td>IE9</td>
<td>JK-type1</td>
<td>2.1.7-7</td>
<td>5x5</td>
<td>39</td>
<td>0.182</td>
<td>5.222</td>
<td>0.302</td>
<td></td>
</tr>
<tr>
<td>IE10</td>
<td>JK-type2</td>
<td>2.1.7-14</td>
<td>5x5</td>
<td>39</td>
<td>0.125</td>
<td>3.600</td>
<td>0.273</td>
<td></td>
</tr>
<tr>
<td>IE11</td>
<td>JK-type3</td>
<td>2.1.7-21</td>
<td>5x5</td>
<td>39</td>
<td>0.125</td>
<td>3.600</td>
<td>0.273</td>
<td></td>
</tr>
<tr>
<td>IE12</td>
<td>JK-type4</td>
<td>2.1.7-32</td>
<td>5x5</td>
<td>39</td>
<td>0.119</td>
<td>3.575</td>
<td>0.227</td>
<td></td>
</tr>
<tr>
<td>IE13</td>
<td>JK-type5</td>
<td>2.1.7-43</td>
<td>5x5</td>
<td>39</td>
<td>0.117</td>
<td>3.484</td>
<td>0.226</td>
<td></td>
</tr>
<tr>
<td>IE14</td>
<td>JK-type1</td>
<td>2.1.7-10</td>
<td>5x5</td>
<td>23 (b)</td>
<td>0.186</td>
<td>5.564</td>
<td>0.300</td>
<td></td>
</tr>
<tr>
<td>IE15</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>3x3</td>
<td>22 (c)</td>
<td>0.258</td>
<td>8.573</td>
<td>0.238</td>
<td></td>
</tr>
<tr>
<td>IE16</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5x5</td>
<td>23 (e)</td>
<td>0.188</td>
<td>6.120</td>
<td>0.313</td>
<td></td>
</tr>
<tr>
<td>IE17</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5x5</td>
<td>23 (d)</td>
<td>0.195</td>
<td>6.115</td>
<td>0.307</td>
<td></td>
</tr>
<tr>
<td>IE18</td>
<td>JK-type1(Fuzzy)</td>
<td>3.1.4-9</td>
<td>5x5</td>
<td>39</td>
<td></td>
<td>0.116</td>
<td>3.251</td>
<td>0.432</td>
</tr>
<tr>
<td>IE19</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5x5</td>
<td>39</td>
<td></td>
<td>0.176</td>
<td>5.313</td>
<td>0.244</td>
</tr>
<tr>
<td>IE20</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5x5</td>
<td>39</td>
<td></td>
<td>0.165</td>
<td>4.822</td>
<td>0.229</td>
</tr>
<tr>
<td>IE21</td>
<td>Fuzzy model</td>
<td>1.3.7-4</td>
<td>5x5</td>
<td>39</td>
<td></td>
<td>0.187</td>
<td>5.559</td>
<td>0.298</td>
</tr>
</tbody>
</table>
Figure 52. Illustration of the Edge of a Noisy Image, $N_3a$, in Example 3a.
Figure 53. Illustration of Mean Error at Each Pixel for (Fuzzy and Non- Fuzzy) Algorithms Concerning Edge Detection of Noisy Image $N/3a$ in Example 3a, the Gaussian Noise With Variance $\sigma^2 = 0.005$. 
Figure 54. Illustration of $SQE$ Versus $\sigma^2$ of a Noisy Image $NI3a$ for Example 3a.

Figure 55. Illustration of $TDC$ Versus $\sigma^2$ of a Noisy Image $NI3a$ for Example 3a.
**Example 3b.** A 130×130 intensity image matrix $NI3b$ ($L=256$), covered by Salt & Pepper noise with 0.02 density, have been applied as an input to the EDS system for edge detection without using thresholding technique. Data obtained from this experiment are also given in Table 14. Some relevant images and graphs are also illustrated in Figures 56 to 59.

**Table 14**

Comparison Table for Fuzzy and Non-Fuzzy Algorithms Applied for 130×130 Noisy Image (Salt & Pepper Noise $d=0.02$) $NI3b$

<table>
<thead>
<tr>
<th>Trace</th>
<th>Methods</th>
<th>Eq.</th>
<th>Block size</th>
<th>(F)QCW structure Fig.</th>
<th>Defuzzification</th>
<th>TME</th>
<th>SQE %</th>
<th>TDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE1</td>
<td>Roberts</td>
<td>1.2.10-1</td>
<td>2×2</td>
<td></td>
<td>0.265</td>
<td></td>
<td>8.577</td>
<td>0.130</td>
</tr>
<tr>
<td>IE2</td>
<td>Sobel</td>
<td>1.2.10-2</td>
<td>3×3</td>
<td></td>
<td>0.194</td>
<td></td>
<td>5.910</td>
<td>0.223</td>
</tr>
<tr>
<td>IE3</td>
<td>Wallis</td>
<td>1.2.10-5</td>
<td>3×3</td>
<td></td>
<td>0.121</td>
<td></td>
<td>3.423</td>
<td>0.014</td>
</tr>
<tr>
<td>IE4</td>
<td>Classic</td>
<td>1.4.4-1</td>
<td>5×5</td>
<td></td>
<td>0.271</td>
<td></td>
<td>6.850</td>
<td>0.193</td>
</tr>
<tr>
<td>IE5</td>
<td>Classic</td>
<td>1.4.4-2</td>
<td>5×5</td>
<td></td>
<td>0.291</td>
<td></td>
<td>11.92</td>
<td>0.045</td>
</tr>
<tr>
<td>IE6</td>
<td>Classic</td>
<td>1.4.4-3</td>
<td>5×5</td>
<td></td>
<td>0.407</td>
<td></td>
<td>20.41</td>
<td>0.154</td>
</tr>
<tr>
<td>IE7</td>
<td>Angular Corr.</td>
<td>2.2.2-1</td>
<td>5×5</td>
<td>39</td>
<td>0.128</td>
<td></td>
<td>3.262</td>
<td>0.225</td>
</tr>
<tr>
<td>IE8</td>
<td>Distance Corr.</td>
<td>2.2.2-2</td>
<td>5×5</td>
<td>39</td>
<td>0.121</td>
<td></td>
<td>3.273</td>
<td>0.337</td>
</tr>
<tr>
<td>IE9</td>
<td>JK-type1</td>
<td>2.1.7-7</td>
<td>5×5</td>
<td>39</td>
<td>0.170</td>
<td></td>
<td>3.797</td>
<td>0.317</td>
</tr>
<tr>
<td>IE10</td>
<td>JK-type2</td>
<td>2.1.7-14</td>
<td>5×5</td>
<td>39</td>
<td>0.124</td>
<td></td>
<td>3.256</td>
<td>0.278</td>
</tr>
<tr>
<td>IE11</td>
<td>JK-type3</td>
<td>2.1.7-21</td>
<td>5×5</td>
<td>39</td>
<td>0.124</td>
<td></td>
<td>3.256</td>
<td>0.278</td>
</tr>
<tr>
<td>IE12</td>
<td>JK-type4</td>
<td>2.1.7-32</td>
<td>5×5</td>
<td>39</td>
<td>0.121</td>
<td></td>
<td>3.324</td>
<td>0.230</td>
</tr>
<tr>
<td>IE13</td>
<td>JK-type5</td>
<td>2.1.7-43</td>
<td>5×5</td>
<td>39</td>
<td>0.119</td>
<td></td>
<td>3.274</td>
<td>0.228</td>
</tr>
<tr>
<td>IE14</td>
<td>JK-type1</td>
<td>2.1.7-10</td>
<td>5×5</td>
<td>23 (b)</td>
<td>0.172</td>
<td></td>
<td>4.245</td>
<td>0.306</td>
</tr>
<tr>
<td>IE15</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>3×3</td>
<td>22 (c)</td>
<td>0.158</td>
<td></td>
<td>4.667</td>
<td>0.293</td>
</tr>
<tr>
<td>IE16</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5×5</td>
<td>23 (e)</td>
<td>0.184</td>
<td></td>
<td>4.835</td>
<td>0.331</td>
</tr>
<tr>
<td>IE17</td>
<td>JK-type1</td>
<td>2.1.7-11</td>
<td>5×5</td>
<td>23 (d)</td>
<td>0.189</td>
<td></td>
<td>4.419</td>
<td>0.332</td>
</tr>
<tr>
<td>IE18</td>
<td>JK-type1(Fuzzy)</td>
<td>3.1.4-9</td>
<td>5×5</td>
<td>39</td>
<td>JK-type1</td>
<td>0.120</td>
<td>3.006</td>
<td>0.426</td>
</tr>
<tr>
<td>IE19</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5×5</td>
<td>39</td>
<td>COA</td>
<td>0.160</td>
<td>3.848</td>
<td>0.262</td>
</tr>
<tr>
<td>IE20</td>
<td>Fuzzy model</td>
<td>1.3.7-5</td>
<td>5×5</td>
<td>39</td>
<td>COA (INT)</td>
<td>0.159</td>
<td>4.829</td>
<td>0.251</td>
</tr>
<tr>
<td>IE21</td>
<td>Fuzzy model</td>
<td>1.3.7-4</td>
<td>5×5</td>
<td>39</td>
<td>MOM</td>
<td>0.179</td>
<td>3.828</td>
<td>0.313</td>
</tr>
</tbody>
</table>
Figure 56. Illustration of Edge of Image \( N3b \) Generated by \( EDS \) in Example 3b.
Figure 57. Illustration of Mean Error at Each Pixel for (Fuzzy and Non-Fuzzy) Algorithms Concerning Edge Detection of Noisy Image NI3b in Example 3b, Salt & Pepper Noise With Density $d=0.02$. 
Figure 58. Illustration of $SQE$ Vs. Density $d$ of a Noisy Image $NI3b$ in Example 3b.

Figure 59. Illustration of $TDC$ Vs. Density $d$ of a Noisy Image $NI3b$ in Example 3b.
As observed in Figures 52 and 53, the derivative algorithms Roberts, Sobel, and Wallis algorithms are strongly sensitive to the noises. Wallis algorithm which generates very low intensity edge IE3 as seen in example 2 Figure 47, can not detect edge points properly since noises are brighter than image pixels. On the contrary, all methods derived from generalized QCW-ECF (such as JK-type1,2,...) possess much less sensitivity to the noises comparing with derivative methods. This is because of using median filtering within each child window $B_k$ and averaging operation within the main block $B-\{(m,n)\}$. In fact, these two operations are noise removal tools applied in QCW-ECF algorithms for edge detection of noisy images.

As seen in Tables 13 and 14, in case of noisy images, all of QCW-ECF significantly generate edge traces with a (a) less total mean error $TME$ (even mean error at each pixel as illustrated in Figures 53 and 57), (b) less mean square error $SQE$, and (c) bigger correlation, comparing with the Sobel and Roberts algorithms. Comparing with the QCW-ECF, Wallis algorithm produces an edge trace with the competitive $TME$ and $SQE$, but with negative or almost zero correlation $TDC$. The mean error at each pixel $ME$ is illustrated in Figures 53 and 57 for both type of noises, Gaussian and Salt & Pepper. In these figures, the upper-bound and lower-bound of mean error curves (generated by QCW-ECF family) are illustrated.

Figures 54 and 58 illustrate the changes of $SQE$ respect to the changes of noise variance $\sigma^2$ (for Guassian noise) and density $d$ for (for Salt & Pepper noise). As seen, the QCW-ECF family has the least $SQE$ comparing to the derivative algorithms. Also Figures 55 and 59 represent the changes of $TDC$ respectively by changes of
noise variance $\sigma^2$ and noise density $d$. As seen, the $QCW-ECF$ family has also the least $TDC$.

As seen in Figures 52 and 56, some noise pixels on edge traces $IE1$ and $IE2$ resulting from derivative algorithms are brighter than edge pixels. Thus, the thresholding technique does not help to eliminate these noise spots from edge traces. In spite of using noise removal operations in $QCW-ECF$, there are still some unwanted noise spots on some edge traces, specially on $IE9$ and $IE16$ ($JK$-type1). But these noise spots have very low intensity and are easily removable by thresholding.

$FQCW-ECF$ ($JK$-type1) is trained by tuning the membership functions ($MF$) of $FLC$ to be sensitive to the noises. For this purpose, a $10 \times 10$ block of image is individually cut from a part of the image $II3a$ and $II3b$ where the noise spots are on the images. Then, like example 1 and 2, we tune the membership functions using cyclic coordinate algorithm through the minimization of performance index $SQE$. The optimal vector $V_{optimal}^*$ containing the membership function parameters resulted from optimization using cyclic coordinate algorithm are (a) $V_{optimal}^* = [.527, .303, .356, .445, .548, .715, .413, .167, .333, .518, .793, .862]$ for example $3a$, and (b) $V_{optimal}^* = [.518, .299, .333, .4310, .547, .719, .403, .166, .333, .528, .791, .868]$ for example $3b$. In both case of noisy image, membership functions are tuned such that $FQCW-ECF$ ($JK$-type1) results in edge traces $IE18$ illustrated in Figures 52 and 53 with a minimum $SQE$ and maximum correlation $TDC$ to the ideal edge trace $Id$. On the other hand, Figures 54, 55, 58, and 59 prove that, for edge detection of a noisy image, $FQCW-ECF$ ($JK$-type1) is a reliable tool in generating the edge traces with a least $SQE$ and
highest correlation to the ideal one without using thresholds. It has the tuning flexibility to be adjusted for edge detection of any kind of noisy images.

4.1.4 Summary of Experimental Results

Most edge extraction techniques based on derivative approximation such as Sobel, Roberts, and Wallis methods possess the disadvantages of, (a) sensitivity to noise; (b) dependency to the size of object on image; (c) inflexibility to the size of the block for block processing; (d) insensitivity to all combinations of the edges (diagonal, horizontal and vertical); and (e) difficulties in thresholding, specially in noisy images.

But the proposed edge characteristic functions based on quadruple child windowing, \textit{QCW-ECF}, possess the advantage of, (a) less sensitive to the noise comparing to the above mentioned; (b) independent to the size of object for edge detection; (c) flexible in size of block for block processing; (d) sensitivity to all kinds of edges, diagonal, vertical and horizontal; and (e) easy to use thresholding technique, thresholding sometimes is not needed.

For many cases, the intensity edge detection techniques depend on the appropriate choice of thresholds. Thus, finding the edge in intensity images at those pixels which have the same intensity with background. Edge detecting algorithms based on fuzzy logic with tunable membership functions are an effective solution to remove the above mentioned problems. The proposed \textit{FQCW-ECF} is an effective tool in fuzzy image processing for edge detection. \textit{FQCW-ECF} possesses the
advantages of, (a) having the same advantages as QCW-ECF; (b) being double purpose tool for both edge detection and defuzzification method in FLC; (c) independent of the threshold; (d) sensitive to similar foreground and background intensities; (e) tunable membership degree provides a significant tool in training algorithms; (f) very low sensitivity to the noises; and (g) being an explicit edge detection function which can be easily applied in optimization.

The Cyclic Coordinate Algorithm is straight-forward approach to tuning the membership function in FLC and optimization of edge detection. This optimization algorithm provides significant results.
CHAPTER V

SUMMARY AND CONCLUSION

Most derivative edge extraction techniques such as Sobel, Roberts, Prewitt, and Wallis methods are limited to making some non-linear manipulation of pixels over a $2 \times 2$ or $3 \times 3$ block as a means of edge enhancement before thresholding. All these mentioned methods possess the disadvantage of sensitivity to noise and dependency on the size of the block in block processing. Their efficiency depend on the size of object on image and have problem in small object edge detection. Finding the edge in an image in which the foreground object and its background are of similar intensity is difficult, especially in case of noisy images.

Most of the edge detection techniques also depend on thresholds. Since choosing an appropriate thresholds is very difficult in case of a noisy and smoothed image, an edge detection algorithm based on fuzzy logic with tunable membership functions is an effective approach. The efficiency can be also improved by training an edge determination algorithm to be sensitive to edges in a consistent manner. The $FQCW-ECF$ introduced in this paper is an effective tool in fuzzy image processing for edge detection. $QCW-ECF$ and its fuzzy extension $FQCW-ECF$ are flexible and can be applied to blocks of any size. The $(F)QCW-ECF$ results in a degree of edginess concerning the middle pixel of the processing block. In other words, it indicates the maximum degree of correspondence of the processing block pattern to
the binary edge patterns. Also the $FQCW-ECF$ is a double-purpose function used as both an edge detection tool and a defuzzification method in fuzzy logic system. They are also sensitive to the diagonal, vertical and horizontal edges. Due to tuning flexibility, the $FQCW-ECF$ provides an ability to train the edge detection algorithm toward an optimal result by minimizing the performance index. This improves the efficiency of the edge detection. Flexibility in deriving the desired edge characteristic function from the proposed general $QCW-ECF$ and its fuzzy associate $FQCW-ECF$ makes these algorithms powerful in using for any kind of application of edge detection. I also developed other methods to estimate edge trace using distance and angle correlation between the processing block pattern (within a block of binary or intensity or fuzzy intensity image) and the binary edge patterns.

Multidimensional optimization using the cyclic coordinate algorithm is a straightforward approach for edge detection optimization based on minimizing the desired performance index. The optimization is done by tuning the input/output membership functions of the Fuzzy Logic Controller ($FLC$) in order to adjust the membership degrees (as weights) of $FQCW-ECF$.

5.1 Future Research

We extended the binary $QCW-ECF$ to the intensity $QCW-ECF$ and its fuzzy version, $FQCW-ECF$, by employing the operator conversions $(\cdot)$, $(\wedge)$ to ($t$-norm) and also $(+),(\vee)$ to ($t$-conorm). We also derived some special cases of $QCW-ECF$ and
FQCW-ECF using the t-(co)norm operators given by Table 10. Deriving QCW-ECF using some other t-(co)norm may improve the performance.

We also employed multidimensional optimization method, cyclic coordinate algorithm, to tune the membership functions in FLC (or the membership degrees $\mu_{d1}$, $\mu_{d2}$, $\mu_{d3}$, $\mu_{d4}$ as weights in FQCW-ECF) and train the edge detection algorithm toward optimal result. Using neural network to train the FQCW-ECF for optimal edge detection can be effective approach for improving the results.
REFERENCES


[16] *Simulation and Modeling*, F. Severance, Electrical & computer Engineering Department, Western Michigan University.
