

ON IMAGE SEGMENTATION USING NEURAL NETWORKS AND FUZZY SETS

ASHISH GHOSH

Electronics and Communication Sciences Unit

Indian Statistical Institute

Calcutta

A thesis submitted to the *Indian Statistical Institute*
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

1992

Dedicated to my father

whom I lost during the course of this work

Acknowledgments

It appears customary to begin the acknowledgments of a thesis by recording one's respect and gratitude to his supervisor for his able guidance, encouragement and other kinds of co-operation. But I find it superfluous, particularly since Professor Sankar K. Pal is much more to me than merely my supervisor. I have been working under his guidance since my master's level and essentially he is holding my hands in taking faltering steps in a new direction of life. Besides that I find him as a nice person with the special quality of becoming a close friend even to students which makes our academic discussion easier.

I consider this thesis as a kind of joint venture by myself, Dr. Nikhil R. Pal and Professor Sankar K. Pal whose records can be found from my list of publications. Thanks are due to them for their kind permission to include the joint works in my thesis. I owe a lot to Dr. Nikhil R. Pal who earnestly inspired me and helped me during the course of this work.

I am grateful to Professor D. Dutta Majumder for taking the responsibility of supervising my work in the absence of Professor Sankar K. Pal and also for his valuable suggestions while preparing this manuscript. I am also grateful to Dr. Malay K. Kundu for the valuable co-operation he has extended in carrying out my research.

I joined this Institute five years ago as an M.Tech student. The list of teachers who taught me during my master's level, encouraged me, helped me in various ways is very large. I am in their debt and it is hardly possible for me to choose a few of them for special mention. Each of my friends and colleagues in our research group, in the department, in the hostel and outside gave me moral support, inspiration and a great environment to work. I thank them all. I am highly grateful to my family members whose love and co-operation were crucial for the successful completion of this dissertation.

Calcutta

November, 1992

Ashish Ghosh

Contents

1	Introduction and scope of the thesis	1
1.1	Introduction	2
1.2	Preliminaries of image processing and analysis	3
1.2.1	Digitization	5
1.2.2	Image quality improvement	5
1.2.3	Primitive extraction	5
1.2.4	Image data compression	7
1.2.5	Image representation and description	7
1.3	Review of image segmentation techniques	8
1.3.1	Classical methods	9
1.3.2	Fuzzy set theoretic methods	19
1.3.3	Neural network based methods	26
1.3.4	Segmentation of color images	29
1.3.5	Objective evaluation of segmentation results	30

1.4	Integration of the theories of fuzzy sets and neural networks . . .	31
1.5	Scope of the thesis	33
1.5.1	Gray level thresholding using fuzzy correlation [122] . . .	34
1.5.2	Gray level thresholding and pixel classification using fuzzy geometry [273,275,276]	34
1.5.3	Kohonen type self-organizing neural network for object extraction [180,192]	35
1.5.4	Hopfield type neural network for object background clas- sification [181,193]	35
1.5.5	Gibbs distribution and Hopfield type neural network for object extraction [182,191]	36
1.5.6	Multi-layer neural network with fuzziness measures for object extraction [194]	36
1.5.7	Robustness analysis of neural network based segmenta- tion techniques under component failure [195]	37
1.5.8	Conclusions and further scope of work	38
2	Gray level thresholding using fuzzy correlation	39
2.1	Introduction	40
2.2	Correlation between two membership functions [120]	41
2.3	Correlation between two fuzzy representations (properties) of an image	42
2.3.1	Correlation between a fuzzy representation of an image and its nearest two tone version	44

2.4	Histogram thresholding by maximization of fuzzy correlation . .	46
2.5	Correlation measure using local information	54
2.5.1	Correlation between two properties of a segmented image	56
2.6	Histogram thresholding by maximization of fuzzy correlation with local information	59
2.6.1	Membership function selection	59
2.6.2	Histogram thresholding	60
2.7	(Dis)Similarity between fuzzy correlation and quadratic index of fuzziness	61
2.8	Implementation and results	63
2.9	Conclusions and discussion	74
3	Gray level thresholding and pixel classification using fuzzy ge- ometry	75
3.1	Introduction	76
3.2	Existing fuzzy geometrical properties	77
3.3	Some new fuzzy geometrical properties	80
3.4	Object background classification by degree of adjacency mini- mization	85
3.4.1	Formulation of algorithm (<i>Algorithm 1</i>)	86
3.4.2	Implementation and results	87
3.5	Object extraction by optimizing IOAC	91
3.5.1	Formulation of algorithm (<i>Algorithm 2</i>)	91

3.5.2	Implementation and results	93
3.6	Pixel classification for object extraction	96
3.6.1	Formulation of algorithm (<i>Algorithm 3</i>)	96
3.6.2	Implementation and results	98
3.7	Conclusions and discussion	101
4	Kohonen type self-organizing neural network for object extraction	103
4.1	Introduction	104
4.2	Kohonen's model of self-organizing feature map	105
4.3	Self-organization and object extraction	106
4.3.1	Image processing in the notion of neural networks	106
4.3.2	Formulation of the object extraction algorithm	107
4.3.3	Selection of optimal threshold	109
4.3.4	Features of the modified model	111
4.4	Simulation and results	114
4.5	Conclusions and discussion	121
5	Hopfield type neural network for object background classification	123
5.1	Introduction	124
5.2	Hopfield's model	125
5.3	Description of the proposed model	127

5.4	Object extraction algorithms	129
5.4.1	Formulation of an energy function	130
5.4.2	Assignment of weights	132
5.4.3	Realization with a neural network	133
5.5	Simulation and results	133
5.6	Conclusions and discussion	145
6	Gibbs distribution and Hopfield type neural network for object extraction	147
6.1	Introduction	148
6.2	Background on Gibbs distribution	150
6.2.1	Basic definition of Gibbs distribution	150
6.2.2	Gibbs distribution used for scene modeling	152
6.3	Image model and problem formulation	153
6.3.1	Gibbsian model for image data	153
6.3.2	Object extraction technique	154
6.4	Algorithm for solving the MAP estimation problem with a neural network	156
6.4.1	Formulation of the algorithm	156
6.4.2	Realization with a neural network	158
6.5	Simulation and results	162
6.6	Conclusions and discussion	173

7	Multi-layer neural network with fuzziness measures for object extraction	174
7.1	Introduction	175
7.2	Measures of fuzziness of a fuzzy set	176
7.3	A self-organizing multi-layer neural network	177
7.3.1	Description and operation of the network	177
7.3.2	Weight correction for different measures of fuzziness	181
7.4	Learning rate for different measures of fuzziness	185
7.5	Application to object extraction	189
7.5.1	Simulation and results	190
7.6	Conclusions and discussion	198
8	Robustness analysis of neural network based segmentation techniques under component failure	200
8.1	Introduction	201
8.2	Modeling of failures and sampling in a neural network	202
8.2.1	Modeling of failures	202
8.2.2	Sampling	203
8.3	Application of failure model to ruggedness evaluation	205
8.4	Simulation and results	207
8.4.1	Way of simulation	207
8.4.2	Results	208

8.5	Conclusions and discussion	222
9	Conclusions and further scope of work	223
9.1	Conclusions	224
9.2	Further scope of work	227
A	Preliminaries of fuzzy set theory and artificial neural networks	230
A.1	Introduction	231
A.2	Theory of fuzzy sets	231
A.2.1	S and π functions	232
A.2.2	A few basic operations on fuzzy sets	232
A.2.3	Measures of fuzziness	233
A.3	Theory of neural networks	236
A.3.1	General properties of neural networks	238
A.3.2	Learning paradigm in neural networks	239
A.3.3	Hopfield's model	240
A.3.4	Kohonen's model	249
A.3.5	Multi-layer perceptron	251
	Bibliography	257
	List of papers by the author	284

List of Tables

2.1	Thresholds based on Algorithm 1	65
2.2	Thresholds based on Algorithm 1	65
2.3	Thresholds based on Algorithm 2	66
2.4	Thresholds based on Algorithm 2	66
2.5	Thresholds based on Algorithm 3	67
2.6	Thresholds based on Algorithm 3	67
2.7	Thresholds based on Algorithm 4	68
2.8	Thresholds based on Algorithm 4	68
2.9	Thresholds with some co-occurrence matrix based methods	73
3.1	Thresholds based on Algorithm 1	89
3.2	Thresholds based on Algorithm 2	93
3.3	Thresholds based on Algorithm 2	94
5.1	Classification accuracy of different approaches	145
8.1	Classification accuracy after node-damage	209

8.2	Classification accuracy after link-damage	210
8.3	Classification accuracy after node and link damage (nodes and links are treated as two separate components)	210
8.4	Classification accuracy after node and link damage (nodes and links are treated as a single component)	211
8.5	Classification accuracy after node-damage	211
8.6	Classification accuracy after link-damage	211
8.7	Classification accuracy after node and link damage (nodes and links are treated as two separate components)	212
8.8	Classification accuracy after node and link damage (nodes and links are treated as a single component)	212
8.9	Classification accuracy after node-damage	213
8.10	Classification accuracy after link-damage	213
8.11	Classification accuracy after node and link damage (nodes and links are treated as two separate components)	213
8.12	Classification accuracy after node and link damage (nodes and links are treated as a single component)	214

List of Figures

2.1	Quadrants of a co-occurrence matrix	56
2.2	Results of thresholding of <i>Biplane</i> image	69
2.3	Results of thresholding of <i>Lincoln</i> image	70
2.4	Results of thresholding of <i>Blurred Chromosome</i> image	71
2.5	Variation of fuzzy correlation with cross over points	72
3.1	A scene with two regions containing different spatial information	84
3.2	Results of thresholding of <i>Biplane</i> image	88
3.3	Results of thresholding of <i>Lincoln</i> image	89
3.4	Results of thresholding of <i>Blurred Chromosome</i> image	90
3.5	Results of thresholding of <i>Shu</i> image	95
3.6	Results of pixel classification	99
3.7	Results of pixel classification of noisy versions of <i>Biplane</i> image	100
4.1	Kohonen's model of neural network	105
4.2	Results of <i>Biplane</i> image	115

4.3	Results of <i>Lincoln</i> image	116
4.4	Results of <i>Noisy tank</i> image	117
4.5	Segmented <i>Noisy tank</i> image	117
4.6	Results of a noisy version ($\sigma = 3$) of the <i>Biplane</i> image	119
4.7	Results of a noisy version ($\sigma = 5$) of the <i>Biplane</i> image	120
4.8	Variation of total square error with threshold	121
4.9	Variation of correlation co-efficient with threshold	121
5.1	Neighborhood system	128
5.2	Topology of the neural network	128
5.3	Input output transfer function	130
5.4	Original synthetic image	135
5.5	Results of a noisy version with $SNR=4.8$	136
5.6	Results of a noisy version with $SNR=1.6$	137
5.7	Results of a noisy version with $SNR=1.0$	138
5.8	Results of <i>Noisy tank</i> image	139
5.9	Results of <i>Biplane</i> image (weights by apriori knowledge)	140
5.10	Results of <i>Biplane</i> image	141
5.11	Results of noisy <i>Biplane</i> image (weights by apriori knowledge)	142
5.12	Results of noisy <i>Biplane</i> image	143
6.1	Neighborhood and associated cliques	151

6.2	Electrical circuit equivalent of a neuron of the proposed model .	160
6.3	Results of a noisy version with $SNR=2.4$	165
6.4	Results of a noisy version with $SNR=1.2$	166
6.5	Results of a noisy version with $SNR=0.75$	167
6.6	Results of <i>Noisy tank</i> image	168
6.7	Results of noisy <i>Biplane</i> image	169
6.8	Results of a noisy version with $SNR=2.4$ for the method described in Chapter 5	170
6.9	Results of a noisy version with $SNR=1.2$ for the method described in Chapter 5	171
6.10	Results of a noisy version with $SNR=0.75$ for the method described in Chapter 5	172
7.1	Schematic representation of a self-organizing multi-layer neural network	178
7.2	Rate of learning with variation of status for different error measures	186
7.3	Results of a noisy version with $SNR=2.4$	191
7.4	Results of a noisy version with $SNR=1.2$	192
7.5	Results of a noisy version with $SNR=0.75$	193
7.6	Results of noisy <i>Biplane</i> image	194
7.7	Results of <i>Noisy tank</i> image	195
7.8	Change of average system error with time	196
7.9	Results of a noisy version with $SNR=1.2$ and N^1	197

8.1	Results of synthetic image by Method 1	215
8.2	Results of synthetic image by Method 2	216
8.3	Results of synthetic image by Method 3	217
8.4	Results of <i>Noisy tank</i> image by Method 1	218
8.5	Results of <i>Noisy tank</i> image by Method 2	219
8.6	Results of <i>Noisy tank</i> image by Method 3	220
8.7	Results of <i>Noisy tank</i> image by Method 4	221
A.1	Schematic representation of a biological neuron	237
A.2	Schematic representation of an electronic neuron	237
A.3	Hopfield's neural network model	243
A.4	Sigmoidal activation function	246
A.5	Kohonen's model of neural network	249
A.6	Schematic representation of the multi-layer perceptron	252
A.7	Activation function for the multi-layer perceptron	253

Chapter 1

Introduction and scope of the thesis

1.1 Introduction

During the last five decades or even more a large number of researchers are trying to design *intelligent systems* to perform tasks at which human beings are more efficient at present. One of the most important behavioral tasks in which human beings show their expertise is image analysis or recognition; where a large amount of pictorial data is processed in a very small amount of time (called real time). Widespread attempts have been made to develop intelligent systems (under different names, like *pattern recognition system*, *image understanding system*, *computer vision system* etc.) for pictorial pattern analysis and recognition. The importance of these subjects is due to their extensive use in a broad range of scientific and commercial application areas such as :

- *document processing* : recognition of printed and hand written characters;
- *biology and medicine* : cytology (blood cell counting, chromosome analysis, etc.), radiology, ECG analysis, EEG analysis and cancerous growth detection;
- *criminal identification* : finger print analysis, hand written character recognition and face recognition;
- *military applications* : target detection, missile guidance and airport detection;
- *industrial automation* : robotics, micrographic image analysis and non-destructive testing;
- *remote sensing* : land surface determination, earth surface classification, radio astronomy, meteorology and environment monitoring;
- *man-machine communication* : processing of visual information for image understanding and scene analysis.

In addition to the classical mathematical techniques (deterministic and probabilistic) new technologies, like *fuzzy set theory* and *artificial neural network*

models are suggested for designing such systems. The theory of fuzzy sets provides an approximate and yet effective means for describing the characteristics of a system which is too complex or ill-defined to admit of precise mathematical analysis. Fuzzy sets can model the human thinking process and behavior, and is reputed to handle (to a reasonable extent) uncertainties arising from deficiencies of information available from a situation, the deficiencies may result from incomplete, ill-defined, not fully reliable, vague and contradictory information.

Artificial neural networks are signal processing systems that try to emulate the human brain, i.e., the architecture and working principle of human nervous systems, by providing a mathematical model of combination of numerous neurons connected in a network. These models are reputed to have the following characteristics : adaptivity, speed, fault tolerance and optimality.

The aim of this thesis is to present some results of investigations, both theoretical and experimental, that demonstrate the effectiveness of some of the concepts of fuzzy set theory and artificial neural network models in formulating various image segmentation/object extraction methodologies for image analysis. The rest of this chapter is organized as follows. In section 1.2 preliminaries of image processing and analysis are described in short. A survey on existing image segmentation techniques is given in section 1.3. A discussion on the integration of the merits of fuzzy set theory and neural networks is made in section 1.4. Scope of the thesis is provided in section 1.5.

1.2 Preliminaries of image processing and analysis

An image (monochrome image) is a two dimensional function $f'(x, y)$; where (x, y) denotes a point in the two dimensional plane (image plane) and $f'(x, y)$ ($0 < f'(x, y) < \infty$) is called the gray value or gray level at that point. The gray value at (x, y) is proportional to the light intensity or brightness value at this point. A digital image $f(x, y)$ is obtained by discretizing an image $f'(x, y)$ in

both spatial domain and brightness values. Thus a digital image can be viewed as a two dimensional matrix whose row and column indices identify a point in the image and the value at this point shows the intensity value (gray value) at that point. Throughout this thesis an L level digital image X of size $M \times N$ will be represented as

$$X = [f(m, n)]_{M \times N} \quad (1.1)$$

where $f(m, n) \in G_L = \{1, 2, \dots, L\}$, the set of gray levels (having a maximum value of L); $m = 1, 2, \dots, M$; $n = 1, 2, \dots, N$.

Each element of an image matrix is called an *image element*, *picture element*, *pixel* or *pel*. From now on the terms *gray level image* or *image* and *pixel* will be used to refer to a digital image and its elements, respectively.

Image processing refers to operations that transform an input digital image into other image forms which are, in some desired sense, improved versions of the input image. *Image analysis*, on the other hand, deals with description of shapes, positions, colors, sizes and relationships among the different regions or parts in the image, and their recognition. The major image processing and analysis tasks are :

- digitization of input image,
- image quality improvement (e.g., restoration and enhancement),
- feature/primitives extraction (e.g., segmentation of different regions, edges or contours detection, image thinning),
- image coding and data compression for efficient storage and transmission,
- representation and description of image data for analysis and recognition.

A short description of these tasks is given in the following sections.

1.2.1 Digitization

Digitization is a process which converts the continuous brightness value (light intensity) and spatial coordinates into discrete components. The quality of the processed image depends, obviously, on the resolution of digitization both in gray levels and spatial coordinates. The higher the resolution in either scale, the better is the quality of the processed output. However, nothing can be gained by increasing the resolutions beyond the spatial and gray scale resolution capabilities of the receiver. For details one can see [1]-[4].

1.2.2 Image quality improvement

In practical situations, often it is noticed that the available image is not of the desired quality that is necessary or not suitable for some specific purpose. The quality of these degraded images can be improved either by *enhancement* or by *restoration*. In enhancement techniques some suitable transformations are applied on the input image so that the result is more suitable than the original for a particular purpose. Various methods have been suggested for this task both in spatial and frequency domain (see [1]-[4], [5]-[12]). The objective of restoration is the same as that of enhancement; but in this technique either the degradation function is known or can be estimated by modeling the degradation process in order to restore the information that has been lost in the image. (Note that in case of enhancement the information is not lost, but needs to be highlighted.) Thus some apriori knowledge about the degradation function is a must for restoration. Further information can be obtained from [1]-[4], [13, 14].

1.2.3 Primitive extraction

Segmentation

The process of partitioning a scene into connected non-overlapping meaningful regions (representing the naturally occurring components in an image) is known as image segmentation. This is one of the most important components of an

image analysis/computer vision system. Various approaches, like histogram thresholding, clustering and iterative pixel classification have been suggested for this task. Once the meaningful regions have been extracted, the relationship among various components can be established and their properties (features) can be computed. Since the present thesis is on image segmentation, a detail discussion on its characteristics, strategies involved and performance evaluation is provided in section 1.3 while reviewing the existing methodologies.

Edge detection

Another widely known process of region separation uses information of intensity change. If the gray levels are relatively consistent in each of the two adjacent extensive regions and if they change abruptly as the border between the regions is crossed, then such a border is called an edge. The higher the difference in intensity between two regions, the stronger is the edge intensity between them and the easier is their detection. Thus, the regions in a scene can be separated by extracting edges. A large number of techniques for edge extraction are available in the literature [1]-[4], [15]-[24]. Some of them are explained in section 1.3.1.D while describing the review on image segmentation techniques.

Thinning

Thinning or skeletonization is a process of transformation that extracts the core lines of images preserving their shape and topological information to a great extent. A considerable amount of data reduction could be obtained while retaining the essential properties of the original image by thinning. Thinning is also necessary for image representation and efficient handling of complex image analysis algorithms. For example, the thinned version (or the skeleton) can be used either as a prototype for efficient matching of the image, or for extracting its constituting arcs, lines, curves and branches representing the structural primitives of the image for its recognition. Details can be obtained in [2], [25]-[32].

1.2.4 Image data compression

Image data compression means the suppression or rejection of redundant information from the total information contained in the given image. Depending on the purpose and the context of application the meaning of redundancy varies. Data compression is needed to reduce the storage requirement and for efficient transmission. If the purpose is reduction of storage, then the compression technique should be such that a perfect reconstruction is possible. In image transmission applications, interest lies in techniques which achieve maximum reduction in the quantity of data to be transmitted subject to the constraint that a reasonable amount of fidelity is preserved. It is mainly achieved by image coding and image thinning/skeletonization. A good coverage of various aspects of image data compression can be found in [1]-[4], [33]-[37].

1.2.5 Image representation and description

Segmentation decomposes a picture into meaningful regions. Different geometrical properties such as size, shape, position, connectedness, border information etc. are then used to represent these regions. Picture description, on the other hand, specify the relationship among these parts. Different models (namely, relational, declarative and procedural) have been suggested for picture description. In the relational model structures such as graphs (in which nodes correspond to regions and arcs correspond to relation between parts) are used for image description. Declarative models put constraints on properties and relations among parts at the time of description. In procedural or syntactic model grammars are used to describe images. For more details one can refer [1, 2], [38]-[44].

1.3 Review of image segmentation techniques

Segmentation is an important step of a pattern recognition, image analysis and machine vision system [1]-[4], [17, 47]. For example, in a vision guided car assembly system the robot needs to pick up appropriate components from the bin. For this, segmentation followed by recognition is required. Segmentation [48, 49, 50] is a process of partitioning the image space into some non-intersecting meaningful regions such that each region is homogeneous and union of no two adjacent regions is homogeneous. Formally, it can be defined [39] as follows. If F is the set of all pixels and $P()$ is a uniformity (homogeneity) predicate defined on groups of connected pixels, then segmentation is a partitioning of the set F into a set of connected subsets or regions $(S_1, S_2, \dots, S_n)$ such that

$$\bigcup_{i=1}^n S_i = F \text{ with } S_i \cap S_j = \phi, \quad i \neq j. \quad (1.2)$$

The uniformity predicate $P(S_i) = \text{true}$ for all regions (S_i) and $P(S_i \cup S_j) = \text{false}$, when S_i is adjacent to S_j .

There are various ways in which one can attempt to extract the image segments. These can be categorized broadly into two groups, namely, classical approach and fuzzy mathematical approach. Under the classical approach we have segmentation techniques based on histogram thresholding, edge detection, iterative pixel classification, and semantic and syntactic approaches. Similarly, fuzzy mathematical approach also has techniques based on thresholding, edge detection, and iterative pixel classification. Besides these, a few attempts have also been made using neural network models for pixel classification and histogram thresholding. It may be mentioned here that there is no single method which can be considered good for all images; nor all methods are equally good for a particular type of image.

Though many algorithms are available for color image segmentation, the literature is not that rich as it is for gray level images. In this context it may be mentioned that the literature is quite rich on the methods of segmentation, but not many attempts have been made for the objective evaluation of segmented outputs. The following subsections present a critical review and summary of

some of the existing methods including color image segmentation and objective evaluation.

1.3.1 Classical methods

A. Gray level thresholding

Thresholding is one of the old, simple and popular techniques for image segmentation. Thresholding can be done based on global information (e.g., gray level histogram of the entire image) or local information (e.g., co-occurrence matrix) of the image. Taxt et al. [51] refer to the local and global information based techniques as contextual and non contextual methods, respectively. Under each of these schemes (contextual/non contextual) if only one threshold is used for the entire image then it is called global or bilevel thresholding. Here bilevel thresholding means the image is partitioned into two regions say, object (black) and background (white). On other hand, when the image is partitioned into several sub-regions and a threshold is determined for each of the sub-regions, it is referred to as local or multi-level thresholding [51]. In this case one tries to get a set of thresholds $\{T_1, T_2, \dots, T_k\}$ such that all pixels with $f(m, n) \in [T_i, T_{i+1}]$, $i = 1, 2, \dots, k - 1$; constitute the i th region type.

If the image is composed of regions with different gray level ranges, i.e., the regions are distinct, the histogram of the image usually shows different peaks, each corresponding to one region; and adjacent peaks are likely to be separated by a valley. For example, if the image has a distinct object on a background, the gray level histogram is likely to be bimodal with a deep valley. In this case, the bottom of the valley (T) is taken as the threshold for object background separation. In other words, gray levels less than or equal to T are taken as parts of the object (background) and others are considered to be its background (object). Therefore, when the histogram has a (or a set of) deep valley(s), selection of threshold(s) becomes easier because, it becomes a problem of detecting valleys. Various methods are available for this.

For example, Otsu [52] used a measure of class separability. He maximized the ratio of the between class variance to the total variance for obtaining thresh-

olds. Nakagawa and Rosenfeld [53] assumed that the object and the background populations are distributed normally with distinct means and standard deviations. Under this assumption they selected the threshold by minimizing the total mis-classification error. This method is computationally involved. Kittler and Illinworth [54], under the same assumption of normal mixture, suggested a computationally less involved method. The method optimizes a criterion function related to average pixel classification rate that finds out an approximate minimum error threshold.

Pun [55] assumed that an image is the outcome of an L symbol source. He maximized an upper bound of the total aposterior entropy of the partitioned image for the purpose of selecting the threshold. Kapur et al. [56], on the other hand, assumed two probability distributions, one for the object region and the other for the background region. They then, maximized the total entropy of the partitioned image in order to arrive at the threshold level. Wong and Sahoo [57] attempted to maximize the aposterior entropy of a partitioned image subject to a constraint on the uniformity measure of Levine and Nazif [58] and a shape measure. They maximized the aposterior entropy over $\min(s_1, s_2)$ and $\max(s_1, s_2)$ to get the threshold for segmentation; where s_1 and s_2 are the threshold levels at which the uniformity and the shape measure attain the maximum values, respectively. Pal and Pal [59] modeled the image as a mixture of two Poisson distributions and developed several parametric methods for segmentation. The assumption of the Poisson distribution has been justified based on the theory of image formation. The algorithms maximize either entropy or minimize the χ^2 statistic. Though these methods use only the histogram information, they are found to produce good results due to the incorporation of the image formation model.

All these methods have a common drawback, they take into account only the histogram information (ignoring the spatial details). As a result, the algorithm may fail to detect thresholds if these are not properly reflected as valleys in the histogram, which is normally the case. There are many thresholding schemes which use spatial information, instead of histogram information. For example, the busyness measure of Weszka and Rosenfeld [2, 60] is dependent on the

co-occurrence of adjacent pixels in an image. They minimized the busyness measure in order to arrive at the threshold for segmentation. Deravi and Pal [61] minimized the conditional probability of transition across the boundary between two regions. This method also uses the local information contained in the co-occurrence matrix of the image. However, finally all these methods threshold the histogram, but since they make use of the spatial details, they result in a more meaningful segmentation than the methods which use only the histogram information. Pal and Pal [62, 63] proposed measures of contrast between regions and homogeneity of regions using the brightness perception aspect of human psychovisual system, and applied them to segmentation. They also defined the higher order local entropy and conditional entropy of an image [63, 64] giving measures of homogeneity and contrast, respectively. These measures are finally applied to develop object extraction algorithms. A concept similar to the second order local entropy of Pal and Pal [64] has been used by Abutaleb [65] for segmentation. Gray value of a pixel and the average of its neighboring pixels have been used there for the computation of the co-occurrence matrix. As a result the boundary of the segmented object usually becomes blurred.

The philosophy behind gray level thresholding, *pixels with gray level $\leq T$ fall into one region and the remaining pixels belong to another region*, may not be true on many occasions, particularly, when the image is noisy or the background is uneven and illumination is poor and improper. In such cases the objects will still be lighter or darker than the background, but any fixed threshold level for the entire image will usually fail to separate the objects from the background. This leads one to the methods of adaptive thresholding and pixel classification (described in the next subsection). In adaptive thresholding [2, 53, 66, 67] normally the image is partitioned into several non-overlapping blocks of equal area and a threshold for each block is computed independently. Chow and Kaneko [66] used the (sub)histogram of each block to determine local thresholds. These local thresholds are then interpolated over the entire image to yield a threshold surface. They [66] used only gray level information. Yanowitz and Bruckstein [67] extended this idea to use combined edge and gray level information. They computed the gray level gradient magnitude from a smooth version of the im-

age. The gradient values have then been thresholded and thinned using a local maxima directed thinning process. Locations of these local gradient maxima are taken as boundary pixels between object and background. The corresponding gray levels in the image are taken as local thresholds. The sampled gray levels are then interpolated over the entire image to obtain the adaptive threshold surface.

B. Iterative pixel classification

i) Relaxation

Relaxation [2, 68, 69] is an iterative approach to segmentation in which classification decision about each pixel can be taken in parallel. Decisions made at neighboring points in the current iteration are then combined to make decision in the next iteration. There are two types of relaxation : probabilistic and fuzzy.

Suppose a set of n pixels $\{p_1, p_2, \dots, p_n\}$ are to be classified into m classes $\{C_1, C_2, \dots, C_m\}$. For the probabilistic relaxation we assume that for each pair of class assignments $p_i \in C_j$ and $p_h \in C_k$, there exists a quantitative measure of compatibility $C(i, j; h, k)$, i.e., the class assignment of pixels are interdependent. It is reasonable to assume that a positive value of $C(i, j; h, k)$ indicates the compatibility of $p_i \in C_j$ and $p_h \in C_k$, while a negative value represents incompatibility, and a zero *don't care* situation. The function C need not be symmetric. Let p_{ij} represent the probability that $p_i \in C_j$ and $p_h \in C_k$ with $0 \leq p_{ij} \leq 1$, and $\sum_j p_{ij} = 1, \forall i$. Intuitively, if p_{hk} is high and $C(i, j; h, k)$ is positive, we increase p_{ij} since it is compatible with the high probability event $p_h \in C_k$. Similarly, if p_{hk} is high and $C(i, j; h, k)$ is negative, we reduce p_{ij} as it is incompatible with $p_h \in C_k$. On the other hand, if p_{hk} is low or $C(i, j; h, k)$ is nearly zero, p_{ij} is not changed as either $p_h \in C_k$ has a low probability or is irrelevant to $p_i \in C_j$. Since this technique has been compared with two of the proposed algorithms in Chapters 5 and 6 some discussion on the updation rule of p_{ij} is made in Chapter 5.

In case of fuzzy relaxation the probability estimates p_{ij} 's are replaced by fuzzy class membership (explained in section 1.3.2) values $\mu_j(i)$'s. Unlike probabilis-

tic relaxation there is no restriction like $\sum_j \mu_j(i) = 1, \forall i$. This approach is applicable when the classes are not mutually exclusive or not exhaustive.

ii) Markov random field based approaches

There are many image segmentation methods which use the spatial interaction model like Markov Random Field (*MRF*) or Gibbs Random Field (*GRF*) [70]-[74] to model digital images. Geman and Geman [73] have proposed a hierarchical stochastic model for the original image and developed a restoration algorithm, based on stochastic relaxation and annealing, for computing the maximum a-posterior estimate of the original scene given a degraded realization. Because of the use of annealing, the restoration algorithm does not stop at local maxima but finds the global maximum of the a-posterior probability. It may be mentioned here that the probabilistic relaxation [2, 68] and stochastic relaxation, although share some common features like parallelism and locality, are quite distinct. Relaxation labeling is essentially a non stochastic (deterministic) process which allows movements only to states (configurations) of lower energy. On the other hand, in stochastic relaxation transition to a configuration which increases the energy (decreases the probability) is also allowed. In fact, if the new configuration decreases the energy the system transits to that state; whereas, if the new configuration increases the energy, the system accepts that state with a probability. This helps the system to avoid the local minima. Relaxation labeling usually gets stuck to local minimum. Moreover, in relaxation labeling there is nothing corresponding to an equilibrium state or even a joint probability law over the configurations.

Derin et al. [72] extended the $1 - D$ Bayes smoothing algorithm of Askar and Derin [75] to $2 - D$ to get the optimum Bayes estimate for the scene value at every pixel. In order to reduce the computational complexity of the algorithm, the scene is modeled as a special class of *MRF* model, called Markov mesh random fields which are characterized by causal transition distributions. The processing is done over a relatively narrow strips and estimates are obtained at the middle section of the strips. These pieces together form overlapping strips and yield a suboptimal estimate of the scene. Without parallel implementation these algorithms become computationally prohibitive. Derin and Elliott [74]

used a doubly stochastic hierarchical model for image data. At the top level a Gibbs distribution is used to characterize the clusters of the image pixels into regions with similar features. At the bottom level, the feature or textural properties of the region types are modeled by a second set of Gibbs distribution, one for each region type. The segmentation algorithms are derived by using the maximum a posterior probability (*MAP*) criterion. To reduce the computational overhead of the exact *MAP* estimate, they derived sub-optimal solutions through simplifying assumptions in the model. The problem of *MAP* estimation of a scene is described in detail in Chapter 6 where a neural network based computationally efficient *MAP* estimation method is suggested.

C. Surface based methods

This section mainly discusses a few selected techniques developed for range image segmentation [77]-[80]. Besl and Jain [78] have developed an image segmentation algorithm based on the assumption that the image data exhibits surface coherence, i.e., image data may be interpreted as noisy samples from a piecewise smooth surface function. Though, this method is probably most useful for range images the method can be used to segment any type of image that can be modeled as a noisy sampled version of a piecewise smooth graph surface. This method is based on the fact that the signs of Gaussian and mean curvatures yield a set of eight surface primitives : peak, pit, ridge, saddle ridge, valley, saddle valley, flat (planar) and minimal. These primitives possess some desirable invariant properties and can be used to decompose any arbitrary smooth surface. In other words, any arbitrary smooth surface can be decomposed into one of those eight possible surface types. These simple surfaces can be well approximated for the purpose of segmentation by bivariate polynomials of order ≤ 4 . The first stage of the algorithm creates a surface type labeled image based on the local information (using mean curvature and Gaussian curvature images). The second stage takes the original image and the surface type labeled image as input and performs an iterative region growing using the variable order surface fitting. In the variable order surface fitting, it has been tried to represent the points in a seed region by a planar surface. If this simple hypothesis of planar surface is found to be true then the seed region is grown on the planar surface fit. If the simple hypothesis fails, then

the next more complicated hypothesis of biquadratic surface fit is tried. If this is satisfied, the region is grown based on that form, otherwise the next complicated form is tried. The process is terminated when either the region growing has converged (same region obtained twice) or when all pre-selected hypotheses fail. In the later case, possibly a higher order surface should be tried.

Hoffman and Jain [77] have developed a method for segmentation and classification of range images. They have used a clustering algorithm to segment the image into surface patches. Different types of clustering algorithms including methods based on minimal spanning tree, mutual nearest neighbor, hierarchical clustering and square error clustering have been attempted. The square error clustering has been found to be the most successful method for range images. The feature set used contains the co-ordinate position (m, n) , the depth value $f(m, n)$ and the estimated unit surface normal vector. The unit surface normal vector is normal to the tangent plane at a point which is obtained by finding the best (in the least square sense) fitting plane over a neighborhood. In the second phase of the method this patches are classified as planar, convex or concave. In order to make the method of classification more effective they have combined three different methods, namely, *non parametric trend test for planarity*, *curvature planarity test*, and the *eigen value planarity test*. In the final stage, boundaries between adjacent surface patches are classified as crease or noncrease edge, and this information is then used to merge adjacent compatible patches to result in reasonable faces of the object. For this type of methods, the choice of the neighborhood to compute the local parameter is an important issue and no theoretical guideline has been provided for this.

Yokoya and Levine [80] used differential geometric technique like the method of Besl and Jain [78] for range image segmentation. They combined both region and edge based techniques. They approximated object surfaces using biquadratic polynomials. As in [78] signs of Gaussian and mean curvatures (curvature sign map) have been used to get the initial region based segmentation. Two edge maps are formed : one for the jump edge and the other for the roof edge. The jump edge magnitude is obtained by computing the maximum difference in depth between a point and its eight neighbors; while the roof edge

magnitude is computed as the maximum angular difference between adjacent unit surface normals. These two edge maps and the curvature sign map are then fused to form the final segmentation. This method too requires the selection of threshold levels for edge maps and the curvature sign map. Improper choice of these parameter values is likely to deteriorate the quality of the segmentation output.

Rimey and Cohen [79] formulated the problem as a *maximum likelihood* segmentation problem. Here also the objective is to divide the range image into windows, classify each window as a particular surface primitive, and group them into surface regions. Homogeneous windows are classified according to a generalized likelihood ratio test. The test uses information from adjacent windows and is computationally simple. Once each window has been classified, similar windows are merged using maximum likelihood clustering analysis.

D. Methods based on edge detection

Segmentation can also be obtained through detection of edges of various regions, which normally tries to locate the points of abrupt changes in gray level intensity values. Since edges are local feature, they are determined from local information. Various methods are available in the literature [1]-[4], [15]-[24], [81]-[84] for edge finding. Davis [19] classified edge detection techniques into two categories: sequential and parallel. In sequential techniques the decision whether a pixel is an edge pixel or not is dependent on the result of the detector at some previously examined pixels. On the other hand, in parallel methods the decision whether a point is an edge or not is taken based on the point under consideration and some of its neighboring points. As a result of this the operator can be applied to every point in the image simultaneously. The performance of a sequential edge detection method is dependent on the choice of an appropriate starting point and how the results of previous points influence the selection and result of the next point.

There are different types of differential operators such as Roberts gradient, Sobel gradient, Prewitt gradient and the Laplacian operator [1]-[4]. These difference operators respond to changes in gray level or average gray level. The gradient operators, not only respond to edges but also to isolated points. For

Prewitt's operator the response to the diagonal edge is weak, while for Sobel's operator it is not that weak as it gives greater weight to points lying close to the point under consideration. The above mentioned operators are called the first difference operators. Laplacian, on the other hand, is a second difference operator. The Laplacian operator is given by

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \quad (1.3)$$

The digital Laplacian being a second difference operator, has a zero response to linear ramps. It responds strongly to corners, lines, and isolated points. Thus for a noisy picture, unless it has a low contrast, the noise will produce higher Laplacian values than the edges. Moreover, the digital Laplacian is not orientation invariant.

A good edge detector, should be a filter with the following two features. First, it should be a differential operator, taking either a first or a second spatial derivative of the image. Second, it should be capable of being tuned to act at any desired scale, so that large filters can be used to detect blurred shadow edges, and small ones to detect sharply focused fine details. The second requirement is very useful as intensity changes occur at different scales in an image. According to Marr and Hildreth [16] the most satisfactory operator fulfilling these conditions is the Laplacian of Gaussian ($L-G$) operator. It is normally denoted by $\nabla^2 G$; where the Laplacian is given by

$$\nabla^2 = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (1.4)$$

and

$$G = e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (1.5)$$

is a two dimensional Gaussian distribution, with standard deviation σ . The Gaussian part of the $L-G$ operator blurs the image, wiping out all structures at scales much smaller than the σ of the Gaussian.

The Gaussian blurring function is preferred over others because it has the desirable property of being smooth and localized in both the spatial and the frequency domain. In order to find the intensity change at a given scale, Marr and Hildreth, first filtered the image with the $\nabla^2 G$ filter and then found the

zero crossings in the filtered image. The space described by the scale parameter σ and the zero-crossing curves is called the scale space.

The behavior of edges in the scale space produced by the $L-G$ operator has been studied by Lu and Jain [84]. In order to formulate rules for reasoning in the scale space they studied dislocation of edges, false edges, and merging of edges with nice mathematical framework.

Canny [22] gave a comprehensive set of precise goals for computation of edge points namely detection, localization and single response for an edge point. The operator due to Canny is a nonsymmetric (directional) one, which uses Gaussian smoothing before detection. It also uses multiple thresholds [23], the value of which may be estimated from the busyness measure [2] of the edgy image.

In case of a noise free image, the edge angle can be measured accurately, but in real life images, noise cannot be avoided and it is difficult to estimate the true edge angles. Kittler et al. [83] suggested three methods to improve the edge angle estimate obtained from Sobel's operator. All the three methods involve averaging of the outputs of the Sobel operator over a 3×3 window. One of the methods, which ignores the effect of the central pixel, at which the angle estimate is to be made, is found to produce the best result. They have justified this counter intuitive view also.

Haralick [21] attacked the problem of edge and region detection from a new angle. He assumed that the observed image is an ideal image with noise added. Each region in the image is a sloped plane. In order to determine the edge between two pixels, best fitted sloped planes over a neighborhood of each pixel are found. Edges are declared at locations having significant different planes on either side of them. The least square error procedure has been used to estimate the parameters of a sloped surface for a given neighborhood. An appropriate F -statistic has been used to test the significance of the difference of the estimated slope from a zero slope or the significance of the difference of the estimated slopes of adjacent neighbors.

An iterative algorithm has been developed by Gokmen and Li [85] using the regularization theory. The energy functional in the standard segmentation has been modified to spatially control the smoothness over the image in order to obtain the accurate location of edges. An algorithm for defining small, optimal kernel conditioned on the important aspects of the imaging process has been suggested by Reichenbach et al. [86] for edge detection. This algorithm takes into account the nature of the scene, the point spread function of the image gathering device, the effect of noise, and generates the kernel values which minimize the expected mean square error of the estimate of the scene characteristics.

All the edges produced by the above mentioned operators are, normally, not significant edges when viewed by human beings. Therefore, one needs to find out prominent (valid) edges from the output of the edge operators. Kundu and Pal [18] have suggested a method of thresholding to extract the prominent edges based on psychovisual phenomena. Haddon [24] also developed a technique to derive a threshold for any edge operator, based on the noise statistics of the image.

1.3.2 Fuzzy set theoretic methods

Fuzzy sets were introduced in 1965 by Lofti Zadeh [89] as a new way to represent vagueness in everyday life. They are generalizations of conventional (crisp) set theory. Conventional sets contain objects that satisfy precise properties required for membership. Fuzzy sets, on the other hand, contain objects that satisfy imprecisely defined properties to varying degrees. A fuzzy set A in a space of points $X = \{x\}$ is a class of events with a continuum of grades of membership and is characterized by a membership function $\mu_A(x)$ which associates with each point in X a real number in the interval $[0, 1]$ with the value of $\mu_A(x)$ at x representing the grade of membership of x in A . Formally, a fuzzy subset A of the universe X is defined as a collection of ordered pairs

$$A = \{(\mu_A(x), x), \forall x \in X\} \quad (1.6)$$

where the support of A is the subset of X and is defined as

$$S(A) = \{x \mid x \in X \ \& \ \mu_A(x) > 0\}. \quad (1.7)$$

This characteristic function $\mu_A(x)$, ($0 \leq \mu_A(x) \leq 1$) in fact, can be viewed as a weighting coefficient which reflects the ambiguity in a set, and as it approaches unity, the grade of membership of an event in A becomes higher. For example, $\mu_A(x) = 1$ indicates a strict containment of the event x in A . If on the other hand, x does not belong to A , $\mu_A(x) = 0$. Any intermediate value would represent the degree to which x could be a member of A . If $\mu_A(x) = 0.5$, then x is called a cross over point. Some aspects of fuzzy set theory including membership functions, basic operations [90]-[96] and measures of fuzziness [90], [97]-[101] are discussed in detail in Appendix A.

A. Image definition and ambiguities

A gray tone image possesses ambiguity within each pixel because of the possible multi-valued levels of brightness. If the gray levels are scaled to lie in the range $[0, 1]$, we can regard the gray level of a pixel as its degree of membership in the set of high-valued *bright pixels* - thus a gray tone image can be viewed as a fuzzy set. Regions, features, primitives, properties, and relations among them that are not crisply defined can similarly be regarded as fuzzy subsets of images.

An L level image X ($M \times N$) can therefore be considered as an array of fuzzy singletons, each having a value of membership denoting its degree of possessing some property (e.g., brightness, darkness, edginess, blurredness, texture etc.). In the notation of fuzzy sets one may therefore write that $X = \{\mu_X(x_{mn}) : m = 1, 2, \dots, M; n = 1, 2, \dots, N\}$; where $\mu_X(x_{mn})$ denotes the grade of possessing such a property μ by the (m, n) th pixel. This property μ of an image may be defined using global information or local information or both depending on the problem. Basic principles and operations of image processing and recognition in the light of fuzzy set theory are available in [93].

Incertitude in an image pattern may be explained in terms of grayness ambiguity or spatial (geometrical) ambiguity or both. Grayness ambiguity means *indefiniteness* in deciding whether a pixel is white or black. Spatial ambiguity refers to *indefiniteness* in the shape and geometry of a region within the

image. Grayness ambiguity measures as reflected by index of fuzziness [90] and entropy [97, 98, 99, 101] are provided in Chapter 7 (also in Appendix A) whereas the spatial ambiguity measures represented by fuzzy geometrical properties [102, 103] are described in Chapter 3.

B. Relevance of fuzzy set theoretic methods

Relevance of fuzzy set theoretic methods in pattern recognition problems has adequately been addressed in the literature [90]-[95], [104]-[108]. Fuzzy set theories are reputed to handle uncertainties [96, 105, 109], to a reasonable extent, arising from deficiencies of information available from a situation (the deficiency may result from incomplete, ill-defined, not fully reliable, vague and contradictory information). As mentioned before, the basic concepts of image analysis, e.g., the concept of an edge, or a boundary or a corner or a relation between regions do not lend themselves well to precise definition [5, 102, 106, 107, 110]. This realization motivated the researchers to develop fuzzy image processing and recognition systems [6, 92, 93, 94, 95, 105, 108, 111, 112, 113, 114].

Conventional approaches to image analysis and recognition [1, 2, 17, 102] as mentioned in section 1.2 consist of segmenting the image into meaningful regions, extracting their edges and skeletons, computing various features/properties (e.g., area, perimeter, centroid etc.) and primitives (e.g., line, corner, curve etc.) of and relationships among the regions, and finally, developing decision rules/grammars for describing, interpreting and/or classifying the image and its sub-regions. In a conventional system each of these operations involves crisp decisions (i.e., yes or no, black or white, 0 or 1) to make regions, features, primitives, properties, relations and interpretations crisp.

Since the regions in an image are not always crisply defined, uncertainty can arise within every phase of the aforesaid tasks. Any decision made at a particular level will have an impact on all higher level activities. An image recognition (or computer vision) system should have sufficient provision for representing and manipulating the uncertainties involved at every processing stage; i.e., in defining image regions, features, matching, and relations among them, so that the system retains as much of the *information content* of the data as possible.

If this is done, the ultimate output (result) of the system will possess minimal uncertainty (and unlike conventional systems, it may not be biased or affected as much by lower level decision components).

For example, consider the problem of object extraction from a scene. Now, the question is *how can one define exactly the target or object region in a scene when its boundary is ill-defined?* Any hard thresholding made for the extraction of the object will propagate the associated uncertainty to subsequent stages (e.g., thinning, skeleton extraction, primitive selection, etc.) and this might, in turn, affect future analysis and recognition. Thus, it is convenient, natural and appropriate to avoid committing ourselves to a specific (hard) decision (e.g., segmentation/ thresholding, edge detection and skeletonization), by allowing the segments or skeletons or contours to be fuzzy subsets of the image, the subsets being characterized by the possibility (degree) to which each pixel belongs to them. Prewitt [5] first suggested that the results of image segmentation should be fuzzy subsets, rather than ordinary subsets.

We shall describe here in brief a few methods of fuzzy segmentation (both gray level thresholding and pixel classification) and edge detection using global and/or local information of an image.

C. Fuzzy thresholding

Different histogram thresholding techniques in providing both fuzzy and non-fuzzy versions of segmented output by minimizing the grayness ambiguity and geometrical ambiguity of an image have been described by Pal et al. in [111, 115, 116]. These algorithms use different 'S' type membership functions (described in Chapter 2 and also in Appendix A) to define fuzzy *object regions* and then select the one which is associated with the minimum (optimum) value of the aforesaid ambiguity measures. The optimum membership function thus obtained enhances the object from background and denotes the membership values of the pixels for the fuzzy *object region*. Note that the cross-over point (the level for which the membership value is 0.5) of the optimum membership function may be considered as a threshold for crisp segmentation. Its extension to multi-thresholding has also been made. The mathematical framework of the algorithms including the selection of 'S' functions, its band width and bounds

has been established by Murthy and Pal [117]. Pal and Pal [118] introduced a measure called higher order entropy of a fuzzy set and applied it in a similar way to object extraction problem using an adaptive membership function.

The problem of determining the appropriate membership function in image processing drew the attention of many researchers. Reconsider the problem of gray level thresholding using 'S' function. If there is a difference in opinion in defining an 'S' function (i.e., instead of a single membership function we have a set of monotonically nondecreasing functions), the concept of spectral fuzzy sets [119] can be used to provide soft decisions (a set of thresholds along with their certainty values) by giving due respect to all the opinions. In making such a decision, the algorithm minimizes differences in opinions in addition to the ambiguity measure; thereby managing the uncertainty. The bounds for 'S'-type function has also been defined recently based on the properties of fuzzy correlation [120] so that any function lying within the bounds would give satisfactory performance. It, therefore demonstrates the flexibility of fuzzy algorithms. Xie and Bedrosian [121] have also made attempt in determining membership functions for gray level thresholding. The concept of fuzzy thresholding is described in detail in Chapters 2 and 3 where correlation between membership functions and fuzzy geometrical properties have been used for developing new segmentation algorithms.

D. Fuzzy clustering

The fuzzy *c*-means (*FCM*) clustering algorithm [92] has also been used to image segmentation [112, 123]. The fuzzy *c*-means algorithm uses an iterative optimization of an objective function based on a weighted similarity measure between the pixels in the image and each of *c* cluster centers. A local extremum of this objective function indicates an optimal clustering of the input data. The objective function that is minimized is given by

$$W_m(U, V) = \sum_{k=1}^n \sum_{i=1}^c (\mu_{ik})^m (d_{ik})^2 \quad (1.8)$$

where μ_{ik} is the fuzzy membership value of the *k*th pixel in the *i*th cluster, d_{ik} is any inner product induced norm metric, *m* controls the nature of clustering with hard clustering at *m* = 1 and increasingly fuzzier clustering at higher

values of m , n = number of pixels, V is the set of c cluster centers and U is the fuzzy c -partition of the image.

Trivedi and Bezdek [124] proposed a fuzzy set theoretic image segmentation algorithm for aerial images. The method is based on region growing principles using a pyramidal data structure. The algorithm is hierarchical in nature. Segmentation of the image at a particular processing level is done by the *FCM* algorithm. In a multi-level segmentation experiment, the regions at i th level are considered homogeneous when image elements have cluster membership values greater than a prescribed threshold. If the homogeneity test fails, the regions are split to form the next level regions which are again subjected to the *FCM* algorithm. This algorithm is a region splitting algorithm, but the acceptance of a region is determined by a fuzzy membership values to different regions.

Some recent results on image segmentation using the fuzzy c -means algorithm and fuzzy integral [125] have been reported in [113, 123, 126, 127]. Hall et al. [128] segmented magnetic resonance brain images using the unsupervised *FCM* and also by a supervised neural network, a dynamic multi-layer perceptron trained with the cascade correlation learning algorithm. The different aspects of both approaches and their utility for the diagnostic process have been discussed. However, the computational complexity of *FCM* algorithms is too high to apply it for real time applications. To get rid of this problem Cannon et al. [123] suggested an approximate version of the algorithm. One of the advantages in using fuzzy clustering algorithms is that one can dynamically select the appropriate number of clusters depending on the strength of memberships across clusters [123].

Backer [129] developed a very general clustering strategy which has been applied to different types of data including images. The set of samples S is first partitioned into c (number of classes) disjoint sets as an initial guess of the desired partition. Then a membership function is assigned to those initial clusters according to some *point to point subset affinity* mechanism for all points in S . In fact, he suggested a number of affinity mechanisms based on the distance concept, the neighborhood concept, and the probabilistic concept. Updating of the partitions (re-partition, reclassification) is then done under the guidance of

some criterion function which characterizes the partition. Three different types of criterion functions, based on measures of fuzziness, inter fuzzy set distance, and measure of fuzzy similarity have been considered there. If changes occur in the earlier step, the process of assigning membership function and updating is repeated otherwise, the algorithm terminates.

We emphasize here that these developments are mainly based on the applications of fuzzy operators, properties and mathematics. Segmentation based on the theory of approximate reasoning (i.e., based on *if-then* rules) should constitute another field of research in the near future.

E. Fuzzy edge detection

Pal and King [111, 130] used a non-symmetrical membership function G to get fuzzy property plane from the intensity plane. The function G is defined as

$$G(f(m, n)) = (1 + |\hat{f} - f(m, n)|/F_d)^{-F_e} \quad (1.9)$$

where $\hat{f}$ is a reference level, F_e and F_d are the exponential and denominational fuzzifiers, respectively. If $\hat{f} = f_{max}$, the maximum gray level, then G approximates the standard 'S' function [89] of Zadeh and when $\hat{f} =$ some other level, $0 < \hat{f} < f_{max}$, it approximates the standard ' π ' function [89]. (Both the 'S' and ' π ' functions are described in Appendix A.) The G functions under the above cases are denoted by G_S and G_π , respectively. They used this G_S and G_π functions in conjunction with a contrast intensification operator INT (described in Appendix A) to intensify the contrast in the image. Note that the non-fuzzy thresholds obtained automatically from fuzzy segmentation techniques can be used in defining G_S and G_π . Finally, an inverse transformation is applied to get the enhanced spatial domain image. Edges of this enhanced image can then easily be found with any spatial domain technique.

Edge detection operators based on *max* and *min* operations are available in [130, 131]. Pal and Pal [63] used the second order entropy of a fuzzy set, defined by an adaptive membership function over a neighborhood of a pixel (m, n) , as a measure of edginess at (m, n) . The use of an adaptive membership function makes the edge detection algorithm robust. The framework of the algorithm is quite general and works with any measure of ambiguity (fuzziness).

1.3.3 Neural network based methods

For any image analysis/vision system, one desires to achieve robustness of the system with respect to random noise and failure of components. Moreover, a system can probably be made artificially intelligent if it is able to emulate some aspects of human information processing system. Another important requirement is to have the output in real time. Artificial neural network (commonly known as neural networks) [134]-[165] based approaches are attempts to achieve these goals. A neural network can formally be defined as *massively parallel interconnected network of simple (usually adaptive) processing elements which are intended to interact with the objects of the real world in the same way as biological systems do.*

Neural networks (NN) are designated by the network topology, connection strength between pairs of neurons (called weights), node characteristics and the status updating rules. Node characteristics mainly specify the primitive types of operations it can perform, like summing the weighted inputs coming to it and then amplifying it or doing some fuzzy aggregation operations. The updating rules may be for weights and/or states of the processing elements (neurons). Normally an objective function is defined which represents the complete status of the network and the set of minima of it corresponds to the set of stable states of the network. The neurons operate asynchronously (status of any neuron can be updated at random times independent of the other) or synchronously (parallelly) thereby providing output in real time. The architecture of the network depends on the goal one is trying to achieve. The massive connections usually make the the system robust while the parallel processing enables the system to produce output in real time. Since there are interactions among the neurons the collective computational property inherently reduces the computational task and makes the system fault tolerant. (For more details on preliminaries of neural network theory and some of the commonly used neural network architectures see Appendix A.)

Major areas in which the neural networks have been applied in order to exploit its high computational power and to make robust decisions are : supervised

classification [166, 167], expert system design [168, 169], complex optimization problems (like traveling salesman problem [139, 170, 171] and minimum cut problem [153]), analog to digital conversion [172, 173], clustering [174, 175], text classification [176], cloud classification [183], pixel classification [184], histogram thresholding [185], image compression [186, 187, 188], image segmentation [189, 190], image restoration [196, 197], object recognition [198, 199] and scene labeling [200].

A few preliminary attempts [184, 185, 189, 190, 201, 202, 203, 204, 205, 206] have been made to develop segmentation algorithms using artificial neural network models. A brief description of them is given below.

Several texture segmentation algorithms based on deterministic and stochastic relaxation principles and their implementation on neural networks are suggested by Manjunath et al. [203]. The segmentation problem is posed as an optimization task and two different optimality criteria (namely the maximum aposterior probability estimate and the maximization of posterior marginal distribution) are considered for its solution. The texture labels are modeled by a Gibbs Markov random field [73, 74]. A neural network architecture having k layers (k =number of textures present in the image) and $M \times N$ neurons per layer (for an $M \times N$ image) is used. The dynamics of the network is same as that of Hopfield's discrete model [137]. An energy function is designed in such a way that its local minima correspond to optimal segmented versions. In the converged state, the set of ON neurons in different layers correspond to different textures.

An image segmentation system which uses a multi-layer perceptron as a central building block is described in [184]. The complete system consists of two parts. First module is called feature extractor and the second one connectionist classifier. The feature extractor basically defines pixel descriptors which capture the distinguishing features of objects and background (may be in gray levels, texture or both). Different features such as local maxima, gradient, median, variance are used. A set of important features are then selected. These feature values are then used as input to the input layer of the feed-forward multi-layer perceptron classifier. Labeled pixels (pixels with known classes) are

used for learning the network parameters (weights and node biases) using back-propagation learning. After the network has learned (i.e., weights have settled) test images are used to measure the performance of classification. Demonstrations are given on 'real images' of combustion chamber, textures and printed circuit boards.

In [185] Babaguchi and others used a three-layer perceptron architecture for image binarization. Binarization is basically a task of converting a gray level image to a bilevel image. In particular they applied it to document (news paper clipping, technical papers etc.) binarization. The input to the network is the gray level histogram (mapped on to $[0,1]$) and the output is the corresponding threshold. Number of units in the input layer is equal to the maximum number of gray levels available, and number of units in the output layer is two. Number of units in the hidden layer is variable. A set of images (with known thresholds) is used to adjust the parameters (weights and biases) of the feed forward multi-layer back-propagating network. The known thresholds are given in different forms (e.g., a single threshold, a number of thresholds with varying degree in an interval). The algorithm has been demonstrated on a set of images.

Silverman [201, 202] suggested a method for using both back-propagation neural networks and competitive learning networks in segmentation of ultrasonic images. A four layer back propagation neural network is used for learning the operation of a *gradient operator* using the gradient image as teaching signal and median image as input for a number of images. After the learning is complete, if an unknown input image is presented the system provides a gradient version as output. He also used a competitive learning network in image segmentation using a single image as input. An extension to 3-dimensional image segmentation is also attempted using competitive learning scheme.

Shah [204] formulated the problem of edge detection in the context of an energy minimizing model. The method is capable of eliminating weak boundaries and small regions. Cortes et al. [205] proposed a neural network to detect potential edges in different orientations. The performance of the system has been investigated through simulation studies using simulated annealing and mean field annealing. In [190] image segmentation problem has been formulated

as a constraint satisfaction problem (*CSP*), and a class of constraint satisfaction neural network (*CSNN*) is proposed. A *CSNN* consists of a set of objects, a set of labels, a collection of constraint relations and a topological constraint describing the neighborhood relationships among various objects. The *CSNN* is viewed as a collection of interconnected neurons. The architecture is chosen in such a way that it represents constraints in the *CSP*.

A mapping method that makes the Hopfield neural network perform the probabilistic relaxation is suggested in [207]. In such a network the neurons/nodes are used to represent various possible hypotheses and pieces of evidence in the relaxation process; and the links embody the relationship among them. Experimental results demonstrating the feasibility of the proposed mapping process by performing a pixel classification task are also given.

At present a large number of researchers are concentrating to make fusion of the concepts of fuzzy set theory and neural network technologies in order to exploit their merits for designing more efficient pattern recognition and computer vision systems. A discussion on this issue is made in section 1.4.

1.3.4 Segmentation of color images

Color is a very important perceptual phenomenon related to human response to different wavelengths in the visible electromagnetic spectrum [208]-[211]. The image is usually described by the distribution of three color components R (red), B (blue) and G (green). Color image is often also represented by three psychological qualities *hue*, *saturation* and *intensity*. These color features can be calculated from the R , B & G by either a linear or a nonlinear transformation.

Ohta et al. [209] attempted to find a set of effective color features by systematic experiments in region segmentation. They applied an Ohlander type segmentation algorithm [210] for the experiment. At every step of segmenting a region, calculation of the new color features is done for the pixels in that regions by Karhunen Loave transform of R , B & G data. Based on extensive experiments, it has been found that three orthogonal color features $I1 = (R + B + G)/3$,

$I_2 = (R - B)/2$ or $(B - R)/2$ and $I_3 = (2G - R - B)/4$ constitute an effective set of features for segmentation.

Spectrum analysis is one of the old techniques of color image segmentation in which prior knowledge about object colors is used to classify pixels. However, in many real life applications prior knowledge about the colors of the object may be difficult to gather. Under this situation clustering techniques can be used. Ohta et al. [210] instead of using the R , B & G color coordinate directly, used I_1 , I_2 and I_3 . Lim and Lee [212] developed a two stage color image segmentation technique based on thresholding and fuzzy c -means (FCM) [92] methods. This method [212] can be viewed as a coarse to fine technique which tries to reduce the computational overhead of FCM . The method is similar to the iterative algorithm proposed by Huntsberger et al. [112] except that it uses the scale space filter for finding the number of clusters. The coarse segmentation attempts to segment using thresholding and then the FCM algorithm is used to classify the pixels which have not yet been assigned to any class in the coarse segmentation phase. Though the method is claimed to find the number of classes automatically, it does have some subjective choices. For example, in the coarse segmentation phase if the number of pixels in a class exceeds a pre-specified number, then only it is taken as a valid class.

1.3.5 Objective evaluation of segmentation results

We have already discussed several methods of image segmentation. It is known that no method is equally good for all images and all methods are not good for a particular type of images. Here an important problem remains to be discussed is how to make a quantitative evaluation of segmentation results. Such a quantitative measure would be quite useful for image analysis/vision applications where automatic decisions are required. This also will help to justify the utility of an algorithm. Unfortunately, human being is the best judge to evaluate the output of any segmentation algorithm. However, some attempts have already been made for the quantitative evaluation.

Levine and Nazif [213] used a two dimensional distance measure that quantifies

the difference between two segmented images, one proposed by human being and the other by an algorithm. Later on, they [58] defined another set of performance parameters such as region uniformity, region contrast, line contrast etc. These measures have also been used for the quantitative evaluation of segmentation algorithms. Lim and Lee [212] attempted to do this by computing the probability of error between the manually segmented image and the segmentation result. Pal and Bhandari [214] used the higher order local entropy as an index to measure the quality of the output. They also suggested the use of symmetric divergence between two probability distributions, one for the output generated by an algorithm and the other for the manually segmented image. The correlation measure [215] between the original image and the segmented one has also been used for the purpose of quantitative evaluation. We have already mentioned that human being is the ultimate judge to make an evaluation of the result. However, one can use a vector of such measures for objective evaluation. For example, if for some segmented image, the correlation, uniformity and entropy are high and divergence is low then one can consider the output to be good.

1.4 Integration of the theories of fuzzy sets and neural networks

Fuzzy set theory provides an approximate but effective and more flexible means of describing the behaviors of the system which are too complex or too ill-defined to admit of precise mathematical analysis by classical methods and tools. Its effectiveness in handling real life problems (mainly in environments where the information is ill-defined, defective, incomplete or uncertain) is adequately demonstrated in [90]-[95], [104]-[117]. The approach has some drawbacks. The choice of membership function to represent a fuzzy set is normally heuristic. Since the operations involved in fuzzy logic based system are set operations, it becomes computationally prohibitive. Though the approach tries to model the human thought process in a decision making system, it has no relation with architecture of the human neural information processing system,

nor does it take into consideration the information storage technique of human beings.

Human intelligence and discriminating power are mainly attributed to the massively connected network of biological neurons in the human brain. Recently attempts have been made to emulate electronically the architecture and information representation scheme of human neural network under the name *artificial neural network* models. The collective computational abilities of the densely interconnected nodes or processors may provide a material technique, at least to a great extent, for solving highly complex real life problems in a manner a human being does. The technique is highly robust (with respect to random noise and failure of components) and parallelly implementable. Impressive demonstrations of the technique have been shown in various problems of *image processing* [183]-[200], *pattern recognition* [105, 149, 166, 167, 174, 175, 176, 221, 222, 223] and *computer vision* [224]-[229].

We see that the fuzzy set theoretic models try to mimic human reasoning and the capability of handling uncertainty, whereas the artificial neural network models attempt to emulate the architecture and information representation schemes of human brain. Integration of the merits of these two technologies, therefore promises to provide, to a great extent, more intelligent systems to handle real life processes.

A large number of researchers are now concentrating on exploiting these modern concepts in the field of pattern recognition and machine vision. The fusion is mainly tried out in two different ways [105, 230]. The first way is to incorporate the concepts of fuzziness into an artificial neural network framework, i.e., to build *fuzzy neural networks*. For example, the target output of the neural network during training can be fuzzy label vectors [231]. In this case the network itself is functioning as a fuzzy classifier [167]. Another way to incorporate fuzziness into the standard neural network is by altering the integration/transfer function at each node so that they perform some sort of fuzzy aggregation (i.e., fuzzy union, intersection etc.) [232]. A different way to incorporate fuzziness into a pre-existing neural network frame work is through the input data, which can be fuzzified in one or several ways. Attempts are

also made in [194] to use fuzziness measures of a fuzzy set for modeling error of neural networks. More over, there are suggestions in the literature about making individual neurons fuzzy [233, 234].

The second fusion methodology is to use neural networks for a variety of computational tasks within the framework of a pre-existing fuzzy model. In this category, for example, attempts are made to have membership function representation by neural networks [235] and implementation of fuzzy logic operations with neural networks [236, 237, 238]. A great deal of effort is also given to derive neural network driven optimal decision rules for fuzzy controllers [239]. For further references on this issue one can refer [149, 221, 222, 223], [240]-[259].

Note that these attempts of integration are mainly in the field of control theory and pattern recognition. No literature on image segmentation is yet available to the knowledge of the author.

1.5 Scope of the thesis

The objective of this thesis is to present some results of investigations, both theoretical and experimental, that demonstrate the effectiveness of some of the concepts of fuzzy set theory and neural network models as tools in image segmentation/object extraction problem. Initially, a set of fuzzy set theoretic thresholding and pixel classification algorithms using some new measures on image correlation and fuzzy image geometry has been described along with the experimental results. Then some neural network based algorithms for extracting objects from noisy environments are presented using self-organizing feature map, auto-associative memory model and a multi-layer neural network. An attempt is made to incorporate some measures of fuzziness in a neural network framework in this context. Robustness of these algorithms with component failure has also been studied using a stochastic model. The performance of the proposed methods has been compared with those of several related techniques. The results of investigations are summarized below on the basis of chapter headings.

1.5.1 Gray level thresholding using fuzzy correlation [122]

A concept of correlation between two properties (fuzzy representations) of an image is introduced in Chapter 2 based on the definition of correlation between two membership functions [120]. Some new measures called *transitional correlation* and *within-class correlation* are then defined which involve information on co-occurrence of pixels. A set of algorithms for image thresholding (both fuzzy and non fuzzy) has been formulated based on these measures. It has been proved that the criterion of *maximizing correlation* is able to detect valleys in a histogram.

A relation between the fuzzy correlation and the quadratic index of fuzziness is theoretically established and experimentally verified. The effectiveness of the algorithms is illustrated on images having different types of histograms, and compared with algorithms based on fuzziness-minimization [111, 115] and co-occurrence matrix [62, 64].

1.5.2 Gray level thresholding and pixel classification using fuzzy geometry [273,275,276]

Chapter 3 consists of two parts. In the first part, some new fuzzy geometrical properties namely, length, breadth, index of area coverage (*IOAC*), major axis, minor axis, center of gravity and density are described with illustration. The concepts of Rosenfeld's *adjacency* and *degree of adjacency* [102] have been critically examined and their new versions are defined. The second part consists of defining various algorithms based on these properties for providing both fuzzy and nonfuzzy segmentation outputs through gray level thresholding and pixel classification. Gray level thresholding is based on optimization of the measures *IOAC* and degree of adjacency. Adjacency measure is considered for pixel classification.

Incorporation of co-occurrence matrix and/or local histogram (over rows and

columns) in computing the geometrical properties makes their conceptual realization easier and computation faster. The effectiveness of the algorithms has been demonstrated on various types of images. As expected, pixel classification technique is seen to be more suitable for noisy images than gray level thresholding. The *IOAC* minimization is seen to provide better results for non-compact objects over compactness minimization [116].

1.5.3 Kohonen type self-organizing neural network for object extraction [180,192]

The object extraction problem is viewed as a self-organizing task and is solved by employing a modified version of Kohonen's neural network model in Chapter 4. A single neuron (node) is assigned to each pixel and is connected to its nearest neighbors in order to implement the concept of self-organizing feature map. Both global and local information have been used as input features. Statistical criteria (such as least square error and correlation maximization) are used for obtaining the optimal output. The proof for the convergence of the algorithms is given. The algorithms are found to work well even for noisy input.

1.5.4 Hopfield type neural network for object background classification [181,193]

In Chapter 5 the task of image segmentation is formulated as an optimization problem and solved by a Hopfield type network. An objective function has been formulated for object background separation which is similar to the expression of energy of a Hopfield type network. A modified version (architecture wise) of Hopfield's model is then proposed where a single neuron (processor) is assigned, as in the case of Chapter 4, to each pixel and is connected to its nearest neighbors. The weights and input biases are given in such a way that the network self-organizes to form compact clusters.

Both the discrete and the continuous dynamics of the network are used for this purpose. Guidelines for initialization of weights for different types of images are also provided. Performance of the proposed methods has been compared with that of the probabilistic relaxation technique. This model is found to be more suitable for noisy images compared to that in Chapter 4.

1.5.5 Gibbs distribution and Hopfield type neural network for object extraction [182,191]

The maximum a posterior probability (*MAP*) estimate of a scene from a noise corrupted realization (modeled by Gibbs random field) is found to be computationally exponential in nature [74]. A modified version (both architecture wise and neuron characteristic wise) of Hopfield's model has been designed for solving this problem in Chapter 6. Here also, a single neuron is assigned to every pixel and it is connected only to all of its nearest neighbors. The energy function of the network is designed in such a way that its minimum value corresponds to the *MAP* estimate of the scene. A possible hardware realization of the neurons is then suggested.

The technique is implemented on a set of noisy images and is found to be highly robust and more noise immune than the method described in Chapter 5. A comparative study with the probabilistic relaxation algorithm is also given.

1.5.6 Multi-layer neural network with fuzziness measures for object extraction [194]

An attempt is made in Chapter 7 to integrate the concept of fuzzy sets and neural network models by incorporating various fuzziness measures in a multi-layer network for object extraction. Since a feed forward multi-layer perceptron (*MLP*) with back-propagation of error requires a set of labeled input patterns (training samples), as such it can not be used for image segmentation, particularly when only one image is available. (However, if images to be processed

are of similar nature, one can use a set of known images for learning and then use the network for processing of other images.) Therefore, besides describing a method of integrating the concept of fuzzy sets and neural networks, the present work also demonstrates a way of using a multi-layer neural network for performing (unsupervised) self-organizing task in image processing, in general, and object extraction in particular.

A *self-organizing multi-layer neural network* architecture suitable for image processing operations is designed. The architecture is also a feed forward one with back propagation of error; but unlike conventional *MLP* it does not require any supervised learning. Each neuron is connected to the corresponding neuron in the previous layer and its neighbors. Another structural difference from the conventional *MLP* is that there exists a feed-back path from the output to the input layer. The status of neurons in the output layer is described as a *fuzzy set*. A fuzziness measure of this set is used as a measure of error in the system (instability of the network).

Learning rates for various measures of fuzziness (e.g., index of fuzziness and entropy) have been theoretically and experimentally studied. An application of this network in extracting objects from noisy scenes is also demonstrated. The results are seen to be comparable to those of the methods described in Chapters 4-6.

1.5.7 Robustness analysis of neural network based segmentation techniques under component failure [195]

Neural network based systems are believed to be robust with respect to both noise and component failure. The robustness of the proposed neural network based segmentation algorithms has been studied with respect to noise in Chapters 4-7. In Chapter 8 the robustness of these algorithms with respect to component failure (damaging of nodes/links) has been investigated. For this purpose, we have formulated, first of all, a theory for investigating the robustness of

neural network based information processing systems in general, and then applied it on the segmentation algorithms. The damaging process is modeled as a *Poisson process*. In order to choose the instants or moments of damaging, a statistical sampling technique is used. The nodes/links to be damaged are determined randomly.

The performance of the object extraction algorithms is evaluated in terms of percentage of pixels correctly classified under different noisy environments and different degrees of damage. The deterioration in the output is seen to be very small even when a large number of nodes/links are damaged; thereby demonstrating the ruggedness (with respect to component failure) of the systems.

1.5.8 Conclusions and further scope of work

The concluding remarks with further scope of research are presented in Chapter 9.

Chapter 2

Gray level thresholding using fuzzy correlation

2.1 Introduction

We described in brief the different approaches (namely classical, fuzzy set theoretic and neural network based) to image segmentation or object extraction in Chapter 1. Since the regions in an image are not always crisply defined, it is convenient, natural and appropriate to avoid committing ourselves to a specific (hard) thresholding. This can be achieved by allowing the segments to be fuzzy subsets of the image space, the subsets being characterized by the possibility (degree) to which each pixel belongs to them. The present chapter provides a set of new measures based on correlation between membership functions [120] and a few algorithms based on them, for multi-level fuzzy thresholding of an image.

A concept of correlation between two fuzzy properties (called fuzzy correlation) of an image is first of all introduced. The properties may be brightness, edginess, texture, color, blurredness etc. As a special case, correlation between a fuzzy property and its nearest two tone property is then computed. It is to be mentioned here that correlation defined in [120] did neither consider the number of supports of a set, nor the dependencies (order) of their occurrence. Both the factors have been taken into account while implementing it on an image. In the second part, some new measures called *transitional correlation* and *within class correlation* measures have been defined which involve information of co-occurrence of pixels. All these measures have been maximized for obtaining optimal fuzzy and nonfuzzy segmented output. A theoretical proof shows that *maximization of fuzzy correlation* can be considered as a criterion of detecting valleys in a histogram. Finally, a relation between fuzzy correlation and quadratic index of fuzziness [90] has been established. The effectiveness of the algorithms (with crisp outputs) is demonstrated on images having different types of histogram, and compared with a fuzziness-minimization [115] and some co-occurrence matrix based techniques [62, 64].

2.2 Correlation between two membership functions [120]

In real life phenomena we come across many characteristics and attributes which are similar in nature, e.g., tall and very tall, glamorous and beautiful etc. representing two fuzzy sets. The distinguishing factor between the fuzzy membership functions of tall and very tall is the degree of tallness; whereas, glamorous and beautiful are two distinct characteristics, but has correlation. If the value of one membership function is quite high the other one cannot be very low. Correlation provides a measure of such relationship between two membership functions representing fuzzy sets.

Let Ω be a closed interval in R . Let $\mu_1 : \Omega \longrightarrow [0, 1]$ and $\mu_2 : \Omega \longrightarrow [0, 1]$ be two continuous fuzzy membership functions. Let the correlation between the fuzzy membership functions μ_1 and μ_2 (defined on the same domain) be represented by $C(\mu_1, \mu_2)$. Then the mathematical formula for $C(\mu_1, \mu_2)$ as defined in [120] is

$$C(\mu_1, \mu_2) = \begin{cases} 1 - \frac{4}{X_1 + X_2} \int \{\mu_1(x) - \mu_2(x)\}^2 dx & \text{if } X_1 + X_2 \neq 0 \\ 1 & \text{if } X_1 + X_2 = 0 \end{cases} \quad (2.1)$$

where $X_1 = \int \{2\mu_1(x) - 1\}^2 dx$ and $X_2 = \int \{2\mu_2(x) - 1\}^2 dx$.

Now $C(\mu_1, \mu_2)$ gives a measure of relation between the natures of μ_1 and μ_2 , i.e., with change of x what happens to μ_1 and μ_2 . $C(\mu_1, \mu_2)$ satisfies a number of properties. Some of them are :

- If for higher values of $\mu_1(x)$, $\mu_2(x)$ takes higher values and the converse is also true then $C(\mu_1, \mu_2)$ must be very high.
- If with increase of x both μ_1 and μ_2 increase then $C(\mu_1, \mu_2) > 0$.
- If with increase of x , μ_1 increases and μ_2 decreases or vice versa then $C(\mu_1, \mu_2) < 0$.
- $C(\mu_1, \mu_1) = 1$.

- $C(\mu_1, 1 - \mu_1) = -1$.
- $C(\mu_1, \mu_2) = C(\mu_2, \mu_1)$.
- $-1 \leq C(\mu_1, \mu_2) \leq 1$.
- $C(\mu_1, \mu_2) = -C(1 - \mu_1, \mu_2)$.
- $C(\mu_1, \mu_2) = C(1 - \mu_1, 1 - \mu_2)$.

Correlation between two continuous membership functions is defined in equation (2.1). If the functions are discrete in nature (as applicable to digital image), the expression takes the form [120]

$$C(\mu_1, \mu_2) = \begin{cases} 1 - \frac{4}{X_1 + X_2} \sum_x \{\mu_1(x) - \mu_2(x)\}^2 & \text{if } X_1 + X_2 \neq 0 \\ 1 & \text{if } X_1 + X_2 = 0 \end{cases} \quad (2.2)$$

where $X_1 = \sum_x \{2\mu_1(x) - 1\}^2$ and $X_2 = \sum_x \{2\mu_2(x) - 1\}^2$.

For the sake of convenience, we will be using the first part of equation (2.2) while referring it in the following discussion.

2.3 Correlation between two fuzzy representations (properties) of an image

An L level image $X(M \times N)$ can be considered as an array of fuzzy singletons, each having a membership value denoting its degree of possessing some property (e.g., brightness, darkness, edginess, blurredness, texture etc.). In the notation of fuzzy sets one may therefore write that

$$X = \{\mu_X(x_{mn}) : m = 1, 2, \dots, M; n = 1, 2, \dots, N\};$$

where $\mu_X(x_{mn})$ denotes the grade of possessing such a property μ by the (m, n) th pixel. Let μ_1 and μ_2 denote two such properties of X .

It may be noted that the expressions of correlation (equations 2.1 and 2.2) do not consider the number of supports in the set. In an image a particular gray level occurs many times. In such cases correlation between two representations characterized by μ_1 and μ_2 may be defined as follows.

Case 1: Correlation in terms of pixel intensity

Let $h(i)$ be the frequency of occurrence of the gray level i . Let f and g be two functions such that

$$\begin{aligned} f(i) &= \mu_1(i) \times h(i) \\ g(i) &= \mu_2(i) \times h(i) \end{aligned} \quad (2.3)$$

where $\mu(i)$ represents the membership value of possessing some property μ by the i th gray level.

Now the correlation between f and g will be given by the expression

$$\begin{aligned} C(f, g) &= 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^L \{f(i) - g(i)\}^2 \\ &= 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^L \{\mu_1(i) - \mu_2(i)\}^2 \times h(i)^2 \end{aligned} \quad (2.4)$$

where $X_1 = \sum_{i=1}^L \{2f(i) - h(i)\}^2$ and $X_2 = \sum_{i=1}^L \{2g(i) - h(i)\}^2$.

Case 2: Correlation in terms of pixel position

The correlation between μ_1 and μ_2 may also be expressed as

$$C(\mu_1, \mu_2) = 1 - \frac{4}{X_1 + X_2} \sum_m \sum_n \{\mu_1(m, n) - \mu_2(m, n)\}^2 \quad (2.5)$$

with $X_1 = \sum_m \sum_n \{2\mu_1(m, n) - 1\}^2$ and $X_2 = \sum_m \sum_n \{2\mu_2(m, n) - 1\}^2$ where $\mu_1(m, n)$ and $\mu_2(m, n)$ denote the degree of possessing the property μ_1 and μ_2 respectively by the (m, n) th pixel.

If the (m, n) th pixel has intensity i , then $\mu_1(m, n) = \mu_1(i)$ for a particular choice of i ($i = 1, 2, \dots, L$). In that case summing over all the pixels is nothing but

summing over the gray levels after multiplying it by the number of occurrences. Hence we can write

$$C(\mu_1, \mu_2) = 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^L \{\mu_1(i) - \mu_2(i)\}^2 h(i) \quad (2.6)$$

with $X_1 = \sum_{i=1}^L \{2\mu_1(i) - 1\}^2 h(i)$ and $X_2 = \sum_{i=1}^L \{2\mu_2(i) - 1\}^2 h(i)$.

Note that the expression in equation (2.4) differs from that in equation (2.6) by a factor of $h(i)$ only.

2.3.1 Correlation between a fuzzy representation of an image and its nearest two tone version

Let μ_2 be the nearest two tone version of μ_1 such that

$$\mu_2(x) = \begin{cases} 0 & \text{if } 0 \leq \mu_1(x) \leq 0.5 \\ 1 & \text{otherwise.} \end{cases} \quad (2.7)$$

Let μ_1 denote a fuzzy *bright image* plane of X having cross over point at s , say, and be dependent only on gray level. Then μ_2 represents its closest two tone version thresholded at s . In this case (from equation 2.4)

$$\begin{aligned} C(f, g) &= 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^s \{(\mu_1(i) - 0)h(i)\}^2 + \sum_{i=s+1}^L \{(\mu_1(i) - 1)h(i)\}^2 \\ &= 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^s \{\mu_1(i)h(i)\}^2 + \sum_{i=s+1}^L \{(1 - \mu_1(i))h(i)\}^2 \\ &= C(X_f, \bar{X}) \end{aligned} \quad (2.8)$$

where X_f is the fuzzy representation (*bright image* plane) of the image X and $\bar{X}$ is its closest two tone version with $X_1 = \sum_{i=1}^L \{(2\mu_1(i) - 1)h(i)\}^2$ and

$$\begin{aligned} X_2 &= \sum_{i=1}^L \{(2\mu_2(i) - 1)h(i)\}^2 \\ &= \sum_{i=1}^L h(i)^2 = \text{constant; [since } \mu_2 = 0 \text{ or } 1]. \end{aligned} \quad (2.9)$$

If the membership value is expressed in terms of pixel position, similar expression can also be obtained with the exception that instead of i , the summation will be taken over (m, n) . In such cases we get (from equation 2.6)

$$C(X_f, \bar{X}) = 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^s \{[\mu_1(i)]^2 h(i)\} + \sum_{i=s+1}^L \{[1 - \mu_1(i)]^2 h(i)\} \quad (2.10)$$

$$\text{with } X_1 = \sum_{i=1}^L [2\mu_1(i) - 1]^2 h(i)$$

$$\begin{aligned} \text{and } X_2 &= \sum_{i=1}^L [2\mu_2(i) - 1]^2 h(i) \\ &= \sum_{i=1}^L h(i) = M \times N = \text{constant.} \end{aligned}$$

In section 2.4 we will show the way of using equation (2.8) and equation (2.10) as a criterion for optimum segmentation of an image.

Proposition 1 : *The fuzzy correlation $C(X_f, \bar{X})$ between a fuzzy representation (X_f) of an image (X) and the nearest two tone version $(\bar{X})$ satisfies the relation $0 \leq C(X_f, \bar{X}) \leq 1$.*

Proof :

Let s be the cross over point of the membership function μ_1 [i.e., $\mu_1(i) \leq 0.5$ if $i \leq s$ and $\mu_1(i) > 0.5$ if $i > s$] and μ_2 be as defined in equation (2.7).

Then in the range $[1, s]$, $\mu_1(i) \leq 0.5$ and hence $\{\mu_1(i)\}^2 \leq 0.25$; and in the range $[s+1, L]$, $\{1 - \mu_1(i)\} \leq 0.5$ and hence $\{1 - \mu_1(i)\}^2 \leq 0.25$.

$$\begin{aligned} \text{Let } T &= 4 \left[\sum_{i=1}^s \{\mu_1(i)h(i)\}^2 + \sum_{i=s+1}^L \{(1 - \mu_1(i))h(i)\}^2 \right] \\ &\leq 4 \left[\sum_{i=1}^s 0.25 h(i)^2 + \sum_{i=s+1}^L 0.25 h(i)^2 \right] \\ &\leq \sum_{i=1}^s h(i)^2 + \sum_{i=s+1}^L h(i)^2 \\ &\leq \sum_{i=1}^L h(i)^2 = X_2 \quad (\text{from equation 2.9}). \end{aligned}$$

Thus, $T \leq X_2$, i.e., $\frac{T}{X_2} \leq 1$.

Now $X_1 = \sum_{i=1}^L \{(2\mu_1(i) - 1)h(i)\}^2$ is a positive quantity. Hence $\frac{T}{X_1 + X_2} \leq 1$; and thus from equation (2.8) we get $C(X_f, \bar{X}) \geq 1 - 1 = 0$.

From above it is evident that $T \geq 0$, i.e., $\frac{T}{X_1 + X_2} \geq 0$. Thus from equation (2.8) $C(X_f, \bar{X}) \leq 1 - 0 = 1$.

$$\text{Hence } \boxed{0 \leq C(X_f, \bar{X}) \leq 1}$$

♣

Note 1 : In case of membership functions expressed in terms of pixel position, Proposition 1 can also be shown to be valid.

2.4 Histogram thresholding by maximization of fuzzy correlation

The criteria regarding the selection of membership functions along with the window size in image processing problems have recently been reported by Murthy and Pal [260]. The criteria involve symmetry in ambiguity around the cross over point and is bounded by the properties of fuzzy correlation [120]. Zadeh's 'S' function (as defined below) satisfies the aforesaid criteria.

Let us construct, say, a fuzzy subset *bright image* characterized by a membership function μ_1 using the standard 'S' function of Zadeh [89] defined as

$$\begin{aligned} \mu_1(x) = S(x; a, b, c) &= 0 && \text{if } x \leq a \\ &= 2 \{(x - a)/(c - a)\}^2 && \text{if } a \leq x \leq b \\ &= 1 - 2 \{(x - c)/(c - a)\}^2 && \text{if } b \leq x \leq c \\ &= 1 && \text{if } c \leq x \end{aligned} \quad (2.11)$$

where $b = \frac{a+c}{2}$ is the crossover point of the membership function μ_1 .

From the properties of correlation we notice that if two functions μ_1 and μ_2 are very close then $C(\mu_1, \mu_2)$ is very high whereas, $C(\mu_1, \mu_2)$ is least when $\mu_2 = 1 - \mu_1$. Since μ_2 is the nearest two tone version of μ_1 , $C(X_f, \bar{X})$ gives a measure of closeness of the two images X_f (a fuzzy bright image plane of an

image X) and $\bar{X}$ (the corresponding two tone version).

Now if we vary the cross over point (keeping the window $w = c - a$ of the membership function μ_1 fixed), both X_f and $\bar{X}$ will change and we will get different values of $C(X_f, \bar{X})$. Among them the image X_f for which $C(X_f, \bar{X})$ is maximum can be viewed as the optimum fuzzy segmented version of the image X (this is optimum in the sense that for any other selection of cross over point of μ_1 , the value of $C(X_f, \bar{X})$ will be lower). The corresponding $\bar{X}$ can be taken as the non-fuzzy (crisp) segmented output.

For a multi-modal image, the algorithm will result in a set of maxima corresponding to different boundaries. Of them the global one corresponds to the boundary between object and background. The concept is summarized below in algorithmic form.

Algorithm 1

- Step 1 : Construct the fuzzy membership planes $X_f = \{(\mu_1(x), x), \forall x\}$ and $\bar{X} = \{(\mu_2(x), x), \forall x\}$ using equations (2.11) and (2.7) respectively.
- Step 2 : Find the correlation coefficient $C(X_f, \bar{X})$ using equation (2.8) for a fixed threshold 'b', $1 \leq b \leq L$.
- Step 3 : Vary 'b' and select those 'b's for which $C(X_f, \bar{X})$ has local maxima.

Among the local maxima, let the global one correspond to X_f^g and $\bar{X}^g$. X_f^g then can be viewed as a fuzzy segmented version of the image X such that $\mu_1(x)$ denotes the degree of possessing brightness property (or the degree of belonging to object, say) of the pixel intensity x . On the other hand, $\bar{X}^g$ denotes the non-fuzzy (crisp) segmented output.

Proposition 2 : *If the cross over point of the membership function μ_1 is moved along the histogram, then $C(\mu_1, \mu_2)$ will have local maxima at the valleys of the histogram.*

To prove this let us have the following assumptions :

- i : histogram h is continuous;

- ii : $h(a) = h(c)$, $a < c$ and h has local maxima (peaks) at points a and c ;
- iii : at $y_0 = \frac{a+c}{2}$, h has a local minimum (valley);
- iv : h is symmetric in $[a, c]$ about y_0 ;
- v : h is convex in $[a, c]$.

Let $\mu_1 : [0, w] \longrightarrow [0, 1]$. Thus we get μ_1 (putting $a = 0$ and $c = w$ in equation 2.11) as

$$\begin{aligned}
 \mu_1(x) &= S(x, w) \\
 &= 0 && \text{if } x \leq 0 \\
 &= 2\left(\frac{x}{w}\right)^2 && \text{if } 0 \leq x \leq \frac{w}{2} \\
 &= 1 - 2\left(\frac{x-w}{w}\right)^2 && \text{if } \frac{w}{2} \leq x \leq w \\
 &= 1 && \text{if } w \leq x.
 \end{aligned} \tag{2.12}$$

From the definition of μ_1 we notice that

A : μ_1 is continuous, $\mu_1(0) = 0$ and $\mu_1(w) = 1$;

B : μ_1 is monotonic non decreasing; and

C : $\mu_1(x) = 1 - \mu_1(w - x)$, $\forall x \in [0, w]$.

To be mentioned here that Murthy and Pal [260] has established that a valley of a histogram will be detected only if one chooses $w < (c - a)$. Thus we will choose $w < (c - a)$.

To prove, $C(\mu_1, \mu_2)$ has a local maximum at any point y_0 , it suffices to prove (from equation 2.6) that $N = \int \{\mu_1(x) - \mu_2(x)\}^2 h(x) dx$ has a local minimum at y_0 (we will refer it as **Part A**) and $D = X_1 + X_2$ has a local maximum at the same point y_0 (we will refer it as **Part B**).

It has already been shown in equation (2.10) that X_2 is constant. Note that outside the range of w , $\mu_1(x) = \mu_2(x)$. Hence the numerator of equation (2.6) is equivalent to

$$N\left(\frac{w}{2}\right) = \int_0^w \{\mu_1(x) - \mu_2(x)\}^2 h(x) dx \tag{2.13}$$

with $\frac{w}{2}$ as the cross over point of μ_1 .

Before going into the details of the proof let us have a few definitions and lemmas.

Let us define the functions $g(x) = \{2\mu_1(x) - 1\}^2$;

$g_1(x - \epsilon) = g(x)$, $\forall x \in [0, w]$; and

$g_2(x) = g(x)$, $\forall x \in [0, w]$.

Thus $g_2(x) = g_1(x - \epsilon)$.

Lemma 1 : $g(x)$ is symmetric about the point $\frac{w}{2}$, $\forall x \in [0, w]$.

Proof :

$$\begin{aligned} g(x) &= \{2\mu_1(x) - 1\}^2 \\ &= \{2(1 - \mu_1(w - x)) - 1\}^2 \\ &= \{2\mu_1(w - x) - 1\}^2 \\ &= g(w - x) \end{aligned}$$

Hence, $g(x) = g(w - x)$. Thus $g(x)$ is symmetric about $= \frac{x+w-x}{2} = \frac{w}{2}$.

Lemma 2 : $g(x)$ is convex in $[0, w]$.

Proof :

$$g(x) = \{2\mu_1(x) - 1\}^2.$$

Hence

$$g'(x) = 2 \{2\mu_1(x) - 1\} 2\mu'_1(x).$$

Now from equation (2.12)

$$\begin{aligned} \mu'_1(x) &= 0 \quad \text{if } x \leq 0 \\ &= 4 \frac{x}{w^2} \quad \text{if } 0 \leq x \leq \frac{w}{2} \\ &= -4 \frac{x-w}{w^2} \quad \text{if } \frac{w}{2} \leq x \leq w \\ &= 0 \quad \text{if } w \leq x. \end{aligned}$$

For $x \in [0, \frac{w}{2}]$, $\mu_1(x) \leq 0.5$.

Thus, $\{2\mu_1(x) - 1\} \leq 0$ and $\mu'_1(x) = 4 \frac{x}{w^2} \geq 0$.

Hence $g'(x) \leq 0$. The equality holds only for $x = \frac{w}{2}$.

For $x \in [\frac{w}{2}, w]$, $0.5 \leq \mu_1(x) \leq 1.0$.

Thus, $\{2\mu_1(x) - 1\} \geq 0$ and $\mu'_1(x) = -4\frac{x-w}{w^2} \geq 0$.

Hence $g'(x) \geq 0$ with the equality at $x = \frac{w}{2}$.

Thus, $g(x)$ is monotonically decreasing in $[0, \frac{w}{2}]$ and monotonically increasing in $[\frac{w}{2}, w]$. Hence $g(x)$ is convex.

Lemma 3 : $g_2(x) \geq g_1(x)$, $\forall x \in [0, \frac{w}{2} - \frac{\epsilon}{2}]$; the equality is at $x = \frac{w}{2} - \frac{\epsilon}{2}$.

Proof :

From the definition of $g_1(x)$ and $g_2(x)$ we can write, $g_2(x) = g_1(x - \epsilon)$; and $g_1(x)$ is monotone decreasing in $[-\epsilon, \frac{w}{2} - \epsilon]$, i.e., in $[0, \frac{w}{2} - \epsilon]$.

Now

$$\begin{aligned} g_2(x) - g_1(x) &= g_1(x - \epsilon) - g_1(x) > 0, \forall x \in [0, \frac{w}{2} - \epsilon] \\ &[\text{since } g_1(x) \text{ is monotone decreasing in } [0, \frac{w}{2} - \epsilon]]. \end{aligned}$$

For $x \in [\frac{w}{2} - \epsilon, \frac{w}{2} - \frac{\epsilon}{2}]$ $g_1(x)$ is increasing and $g_2(x)$ is decreasing. Also $g_2(\frac{w}{2} - \epsilon) > g_1(\frac{w}{2} - \epsilon)$. Thus the difference $g_2(x) - g_1(x)$ gradually decreases.

Again,

$$\begin{aligned} g_2\left(\frac{w}{2} - \frac{\epsilon}{2}\right) &= g_2\left(\frac{w}{2} + \frac{\epsilon}{2}\right) \quad [\text{since } g_2 \text{ is symmetric around } \frac{w}{2}] \\ &= g_1\left(\frac{w}{2} + \frac{\epsilon}{2} - \epsilon\right) \\ &= g_1\left(\frac{w}{2} - \frac{\epsilon}{2}\right). \end{aligned}$$

Hence, $g_2(x) \geq g_1(x)$ for $x \in [\frac{w}{2} - \epsilon, \frac{w}{2} - \frac{\epsilon}{2}]$ with the equality at $x = \frac{w}{2} - \frac{\epsilon}{2}$.

Lemma 4 : $g_1(x) \geq g_2(x)$, $\forall x \in [\frac{w}{2} - \frac{\epsilon}{2}, w - \epsilon]$.

Proof : Similar as in Lemma 3.

Lemma 5 : Let $d(x) = |g_2(x) - g_1(x)|$, $\forall x \in [0, w - \epsilon]$. Then $d(x)$ is symmetric about $(\frac{w}{2} - \frac{\epsilon}{2})$.

Proof :

$g_1(x)$ is defined in $[-\epsilon, w - \epsilon]$ and is symmetric about $\frac{w}{2} - \epsilon$. $g_2(x)$ is defined in $[0, w]$ and is symmetric about $\frac{w}{2}$. Thus,

$$\begin{aligned} g_2(x) &= g_1(x - \epsilon) \\ &= g_1(w - x - \epsilon). \end{aligned}$$

$$\begin{aligned} g_1(x) &= g_2(x + \epsilon) \\ &= g_2(w - x - \epsilon). \end{aligned}$$

Now,

$$\begin{aligned} d(x) &= |g_2(x) - g_1(x)|, \quad \forall x \in [0, w - \epsilon] \\ &= |g_1(w - x - \epsilon) - g_2(w - x - \epsilon)| \\ &= d(w - \epsilon - x). \end{aligned}$$

Thus $d(x)$ is symmetric about $\frac{w - \epsilon}{2} = \frac{w}{2} - \frac{\epsilon}{2}$.

Proof of Part A

Suppose $\delta = \frac{1}{2}(c - a - w)$.

Let $0 < \epsilon < \delta$ and $y = y_0 - \epsilon$ be the crossover point of the function μ_1 defined over $[y_0 - \epsilon - \frac{w}{2}, y_0 - \epsilon + \frac{w}{2}]$. Thus from equation (2.13) we get

$$\begin{aligned} N(y) &= \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \epsilon - \frac{w}{2} + x) dx + \int_{\frac{w}{2}}^w \{1 - \mu_1(x)\}^2 h(y_0 - \epsilon - \frac{w}{2} + x) dx \\ &= I_1 + I_2, \quad (\text{say}). \end{aligned}$$

Now

$$I_2 = \int_{\frac{w}{2}}^w \{1 - \mu_1(x)\}^2 h(y_0 - \frac{w}{2} - \epsilon + x) dx.$$

Substituting $x = w - z$

$$\begin{aligned} I_2 &= - \int_{\frac{w}{2}}^0 \{1 - \mu_1(w - z)\}^2 h(y_0 - \frac{w}{2} - \epsilon + w - z) dz \\ &= \int_0^{\frac{w}{2}} \mu_1(z)^2 h(y_0 + \frac{w}{2} - \epsilon - z) dz \quad [\text{from the properties of } \mu_1] \\ &= \int_0^{\frac{w}{2}} \mu_1(z)^2 h(y_0 - \frac{w}{2} + \epsilon + z) dz \quad [\text{as } h(x) \text{ is symmetric}]. \end{aligned}$$

Thus

$$\begin{aligned}
N(y) &= I_1 + I_2 \\
&= \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} - \epsilon + x) dx \\
&\quad + \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} + \epsilon + x) dx \\
&= 2 \left\{ \frac{1}{2} \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} - \epsilon + x) dx \right. \\
&\quad \left. + \frac{1}{2} \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} + \epsilon + x) dx \right\} \\
&\geq 2 \left\{ \int_0^{\frac{w}{2}} \mu_1(x)^2 h \left\{ \frac{1}{2} (y_0 - \frac{w}{2} - \epsilon + x + y_0 - \frac{w}{2} + \epsilon + x) \right\} \right\} dx \\
&\quad \text{(since } h(x) \text{ is convex in } [0, w]) \\
&= 2 \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} + x) dx.
\end{aligned}$$

Now

$$N(y_0) = \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} + x) dx + \int_{\frac{w}{2}}^w \{1 - \mu_1(x)\}^2 h(y_0 - \frac{w}{2} + x) dx.$$

In this case μ_1 is defined over $(y_0 - \frac{w}{2})$ to $(y_0 + \frac{w}{2})$.

As in the previous case the expression can be written as

$$N(y_0) = 2 \int_0^{\frac{w}{2}} \mu_1(x)^2 h(y_0 - \frac{w}{2} + x) dx.$$

Thus $N(y_0) \leq N(y)$.

For $y = y_0 + \epsilon$, $[0 < \epsilon < \delta]$, a similar proof holds. Hence we can infer

$$N(y_0) \leq N(y), \quad \forall y \in [y_0 - \delta, y_0 + \delta].$$

So, $\int \{\mu_1(x) - \mu_2(x)\}^2 h(x) dx$ will be locally minimum at the valley points.

Proof of Part B

We are to prove that $D = X_1 + X_2$ has a local maximum at y_0 . Since X_2 is constant, it is sufficient to prove that $X_1 = \int \{2\mu_1(x) - 1\}^2 h(x) dx$ has a local maximum at y_0 .

Let $y = y_0 - \epsilon$ as in **Part A**. Then

$$\begin{aligned} X_1(y) &= \int \{2\mu_1(x) - 1\}^2 h(x) dx \\ &= \int_{-\epsilon}^w \{2\mu_1(x) - 1\}^2 h(x) dx + C_1. \\ &\quad (C_1 \text{ is a constant as } \mu_1(x) = 0 \text{ or } 1 \text{ outside the range } [-\epsilon, w]) \\ &= \int_{-\epsilon}^0 g_1(x) h(x) dx + \int_0^{\frac{w}{2}-\epsilon} g_1(x) h(x) dx \\ &\quad + \int_{\frac{w}{2}-\epsilon}^{w-\epsilon} g_1(x) h(x) dx + \int_{w-\epsilon}^w h(x) dx + C_1. \end{aligned}$$

Similarly,

$$X_1(y_0) = \int_{-\epsilon}^0 h(x) dx + \int_0^{\frac{w}{2}} g_2(x) h(x) dx + \int_{\frac{w}{2}}^w g_2(x) h(x) dx + C_1.$$

Now,

$$\begin{aligned} X_1(y_0) - X_1(y) &= \int_{-\epsilon}^0 \{1 - g_1(x)\} h(x) dx + \int_0^{\frac{w}{2}-\epsilon} \{g_2(x) - g_1(x)\} h(x) dx \\ &\quad + \int_{\frac{w}{2}-\epsilon}^{w-\epsilon} \{g_2(x) - g_1(x)\} h(x) dx + \int_{w-\epsilon}^w \{g_2(x) - 1\} h(x) dx \\ &= I_1 + I_2 + I_3 + I_4, \quad (\text{say}). \end{aligned}$$

$$\begin{aligned} I_4 &= \int_{w-\epsilon}^w \{g_2(x) - 1\} h(x) dx \\ &= \int_0^{\epsilon} \{g_2(x) - 1\} h(x) dx \quad [\text{from symmetry}]. \end{aligned}$$

Hence,

$$I_1 + I_4 = \int_{-\epsilon}^0 \{1 - g_1(x)\} h(x) dx - \int_0^{\epsilon} \{1 - g_2(x)\} h(x) dx.$$

From the definition of g_1 and g_2 we have

$$g_2(x) = g_1(x - \epsilon),$$

and from the convexity of the histogram

$$h(x) \geq h(y), \quad x \in [-\epsilon, 0] \text{ and } y \in [0, \epsilon].$$

Thus

$$\int_{-\epsilon}^0 \{1 - g_1(x)\} h(x) dx \geq \int_0^{\epsilon} \{1 - g_2(x)\} h(x) dx \implies I_1 + I_4 > 0.$$

$$\begin{aligned}
I_2 + I_3 &= \int_0^{\frac{w}{2} - \frac{\epsilon}{2}} \{g_2(x) - g_1(x)\} h(x) dx + \int_{\frac{w}{2} - \frac{\epsilon}{2}}^{w - \epsilon} \{g_2(x) - g_1(x)\} h(x) dx \\
&= \int_0^{\frac{w}{2} - \frac{\epsilon}{2}} \{g_2(x) - g_1(x)\} h(x) dx - \int_{\frac{w}{2} - \frac{\epsilon}{2}}^{w - \epsilon} \{g_1(x) - g_2(x)\} h(x) dx.
\end{aligned}$$

As $|g_2 - g_1|$ is symmetric about $\frac{w}{2} - \frac{\epsilon}{2}$ and $h(\frac{w}{2} - \frac{\epsilon}{2} - x) > h(\frac{w}{2} - \frac{\epsilon}{2} + x)$,

$$\int_0^{\frac{w}{2} - \frac{\epsilon}{2}} \{g_2(x) - g_1(x)\} h(x) dx > \int_{\frac{w}{2} - \frac{\epsilon}{2}}^{w - \epsilon} \{g_1(x) - g_2(x)\} h(x) dx.$$

Hence $I_2 + I_3 > 0$. Thus $I_1 + I_2 + I_3 + I_4 > 0$ and hence $X_1(y_0) \geq X_1(y)$. This shows that X_1 has local maximum at y_0 . Since X_2 is constant, $X_1 + X_2$ also has local maximum at y_0 . This proves **Part B**.

From the proof of **Part A** we get $\int \{\mu_1(x) - \mu_2(x)\}^2 h(x) dx$ will be locally minimum at the valley points; and that of **Part B** $X_1 + X_2$ will be locally maximum at the same valley points. This shows that $C(\mu_1, \mu_2)$ will have local maxima at the valley points.

Hence the proof. ♣

2.5 Correlation measure using local information

In the previous section, the membership values were computed based on global information i.e., individual gray levels. Equation (2.11) was used in this context to represent a fuzzy subset *bright image*. In this section, we are going to explain how the local information in X can be incorporated in computing correlation. For example, the properties edginess, blurredness, texture involve local (spatial) information of the image for their computation.

Let μ_1 and μ_2 be two such properties of the image. The correlation between two properties can be expressed by the following ways :

Case 1 : In terms of pixel intensity

Let $X(M \times N)$ be an L level image and $C(i, j)$ the frequency of occurrence of the gray level i followed by j , i.e., C is the co-occurrence matrix [64]. Let f

and g be two functions such that

$$\begin{aligned} f(i, j) &= \mu_1(i, j)C(i, j) \\ g(i, j) &= \mu_2(i, j)C(i, j) \end{aligned} \quad (2.14)$$

where $\mu(i, j)$ represents the grade of possessing some property by the gray level i followed by j in a specific fashion. Then

$$\begin{aligned} C(f, g) &= 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^L \sum_{j=1}^L \{f(i, j) - g(i, j)\}^2 \\ &= 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^L \sum_{j=1}^L \{[\mu_1(i, j) - \mu_2(i, j)] C(i, j)\}^2 \end{aligned} \quad (2.15)$$

$$\text{where } X_1 = \sum_{i=1}^L \sum_{j=1}^L \{2f(i, j) - C(i, j)\}^2$$

$$\text{and } X_2 = \sum_{i=1}^L \sum_{j=1}^L \{2g(i, j) - C(i, j)\}^2.$$

Case 2: In terms of pixel position

Let $\mu_1(m, n)$ and $\mu_2(m, n)$ denote the grade of possessing two local properties μ_1 and μ_2 (respectively) by the (m, n) th pixel. Then

$$C(\mu_1, \mu_2) = 1 - \frac{4}{X_1 + X_2} \sum_m \sum_n \{\mu_1(m, n) - \mu_2(m, n)\}^2 \quad (2.16)$$

$$\text{with } X_1 = \sum_m \sum_n \{2\mu_1(m, n) - 1\}^2$$

$$\text{and } X_2 = \sum_m \sum_n \{2\mu_2(m, n) - 1\}^2.$$

Let the local properties possessed by the pixels depend on the neighboring gray values of them. If the (m, n) th pixel has intensity i and one of its neighboring intensity is j , then $\mu_1(m, n) = \mu_1(i, j)$; equation (2.16) then takes the form

$$C(\mu_1, \mu_2) = 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^L \sum_{j=1}^L \{\mu_1(i, j) - \mu_2(i, j)\}^2 C(i, j) \quad (2.17)$$

$$\text{with } X_1 = \sum_{i=1}^L \sum_{j=1}^L \{2\mu_1(i, j) - 1\}^2 C(i, j)$$

$$\text{and } X_2 = \sum_{i=1}^L \sum_{j=1}^L \{2\mu_2(i, j) - 1\}^2 C(i, j).$$

Note that, as in the case of $C(f, g)$ and $C(\mu_1, \mu_2)$ in section 2.3, the equation (2.17) differs from equation (2.15) by a factor of $C(i, j)$.

2.5.1 Correlation between two properties of a segmented image

Suppose an image has two distinct regions. Let the background consists of the levels $[1, s]$ and the object $[s + 1, L]$. Then the co-occurrence matrix C [Figure 2.1] can be divided into 4 quadrants C_1 , C_2 , C_3 and C_4 where

- entries in C_1 or C_3 give a reflection of the number of occurrences of background pixels followed by background pixels;
- entries in C_2 or C_4 give a reflection of the number of occurrences of background pixels followed by object pixels;
- entries in C_3 or C_1 give a reflection of the number of occurrences of object pixels followed by object pixels;
- entries in C_4 or C_2 give a reflection of the number of occurrences of object pixels followed by background pixels.

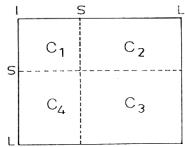


Figure 2.1: Four quadrants of the co-occurrence matrix of an image

Correlation between any two properties of X computed over the entries in individual quadrants C_1 , C_3 , C_2 and C_4 may be termed as background correlation

$[C(f,g)_B]$, object correlation $[C(f,g)_O]$, transitional (background to object) correlation $[C(f,g)_{B/O}]$, and transitional (object to background) $[C(f,g)_{O/B}]$ correlation, respectively. They may be computed from the following equations.

$$C(f,g)_B = 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^s \sum_{j=1}^s \{[\mu_1(i,j) - \mu_2(i,j)] C(i,j)\}^2 \quad (2.18)$$

$$\text{with } X_1 = \sum_{i=1}^s \sum_{j=1}^s \{[2\mu_1(i,j) - 1] C(i,j)\}^2$$

$$\text{and } X_2 = \sum_{i=1}^s \sum_{j=1}^s \{[2\mu_2(i,j) - 1] C(i,j)\}^2.$$

$$C(f,g)_O = 1 - \frac{4}{X_1 + X_2} \sum_{i=s+1}^L \sum_{j=s+1}^L \{[\mu_1(i,j) - \mu_2(i,j)] C(i,j)\}^2 \quad (2.19)$$

$$\text{with } X_1 = \sum_{i=s+1}^L \sum_{j=s+1}^L \{[2\mu_1(i,j) - 1] C(i,j)\}^2$$

$$\text{and } X_2 = \sum_{i=s+1}^L \sum_{j=s+1}^L \{[2\mu_2(i,j) - 1] C(i,j)\}^2.$$

$$C(f,g)_{B/O} = 1 - \frac{4}{X_1 + X_2} \sum_{i=1}^s \sum_{j=s+1}^L \{[\mu_1(i,j) - \mu_2(i,j)] C(i,j)\}^2 \quad (2.20)$$

$$\text{with } X_1 = \sum_{i=1}^s \sum_{j=s+1}^L \{[2\mu_1(i,j) - 1] C(i,j)\}^2$$

$$\text{and } X_2 = \sum_{i=1}^s \sum_{j=s+1}^L \{[2\mu_2(i,j) - 1] C(i,j)\}^2.$$

$$C(f,g)_{O/B} = 1 - \frac{4}{X_1 + X_2} \sum_{i=s+1}^L \sum_{j=1}^s \{[\mu_1(i,j) - \mu_2(i,j)] C(i,j)\}^2 \quad (2.21)$$

$$\text{with } X_1 = \sum_{i=s+1}^L \sum_{j=1}^s \{[2\mu_1(i,j) - 1] C(i,j)\}^2$$

$$\text{and } X_2 = \sum_{i=s+1}^L \sum_{j=1}^s \{[2\mu_2(i,j) - 1] C(i,j)\}^2.$$

Note that $C(f,g)_O + C(f,g)_B$ gives the total *within-class* (object-background) correlation and $C(f,g)_{O/B} + C(f,g)_{B/O}$ gives the total *transitional* correlation. It can also be noticed that the sum of the above four quantities (equations 2.18-2.21) gives the same expression as that in equation (2.15).

Similar definitions can also be provided in terms of pixel coordinates.

Let us consider μ_2 as defined in equation (2.7). Then equations (2.18)-(2.21) provide the correlation between a gray tone image and the corresponding two tone version using local information (computed over different quadrants of the co-occurrence matrix).

Note 2: Proposition 1 is valid when we use equation (2.15) as the expression of correlation.

Proof:

Let μ_1 , μ_2 and s be chosen as in **Proposition 1**. Then for the entries of the quadrants C_1 , C_2 and C_4 , $\mu_1(i, j) \leq 0.5$ and hence $\{\mu_1(i, j)\}^2 \leq 0.25$ and for the entries of the quadrant C_3 , $\{1 - \mu_1(i, j)\} \leq 0.5$ and hence $\{1 - \mu_1(i, j)\}^2 \leq 0.25$.

$$\begin{aligned} \text{Let, } T &= 4 \left[\sum_{i=1}^L \sum_{j=1}^s \{\mu_1(i, j)C(i, j)\}^2 + \sum_{i=1}^s \sum_{j=s+1}^L \{\mu_1(i, j)C(i, j)\}^2 \right. \\ &\quad \left. + \sum_{i=s+1}^L \sum_{j=s+1}^L \{[1 - \mu_1(i, j)]C(i, j)\}^2 \right] \\ &\leq 4 \left[\sum_{i=1}^L \sum_{j=1}^s \{0.5 \times C(i, j)\}^2 + \sum_{i=1}^s \sum_{j=s+1}^L \{0.5 \times C(i, j)\}^2 \right. \\ &\quad \left. + \sum_{i=s+1}^L \sum_{j=s+1}^L \{0.5 \times C(i, j)\}^2 \right] \\ &\leq \sum_{i=1}^L \sum_{j=1}^L C(i, j)^2 = X_2. \end{aligned}$$

Thus, $T \leq X_2$, i.e., $\frac{T}{X_2} \leq 1$.

Now $X_1 = \sum_{i=1}^L \sum_{j=1}^L \{[2\mu_1(i, j) - 1]C(i, j)\}^2$ is a positive quantity.

Hence $\frac{T}{X_1 + X_2} \leq 1$.

So, from equation (2.15) we get $C(X_f, \bar{X}) \geq 1 - 1 = 0$.

From above it is evident that $T \geq 0$, i.e., $\frac{T}{X_1 + X_2} \geq 0$.

Again from equation (2.15) we can write $C(X_f, \bar{X}) \leq 1 - 0 = 1$.

$$\text{Hence } \boxed{0 \leq C(X_f, \bar{X}) \leq 1}.$$



The proposition is also valid for the expressions of correlation in equations (2.18)-(2.21).

2.6 Histogram thresholding by maximization of fuzzy correlation with local information

2.6.1 Membership function selection

Equation (2.11) is suitable for one dimensional input only (i.e., $x \in R$) and it considers only the global information of X . To take the local information into account, the function has been modified considering $\mathbf{x} \in R^n$. From equation (2.11) when $a \leq x \leq b$, and $x \in R$, we can write (putting $w = c - a$)

$$\begin{aligned} S(x, b, w) &= 2 \{(x - a)/w\}^2 \\ &= 2 \{(x - b + w/2)/w\}^2 \\ &= 2 \{1/2 + (x - b)/w\}^2 \\ &= 2 \{1/2 - (b - x)/w\}^2. \end{aligned}$$

When $\mathbf{x} \in R^n$ the above expression can be written in the form

$$S(\mathbf{x}, \mathbf{b}, w) = 2(1/2 - \|\mathbf{b} - \mathbf{x}\|/w)^2 \quad \text{if } \|\mathbf{a}\| \leq \|\mathbf{x}\| \leq \|\mathbf{b}\|$$

where $\|\mathbf{b} - \mathbf{x}\|$ is the Euclidean distance between $\mathbf{b}$ and $\mathbf{x}$, and w is the Euclidean distance between $\mathbf{c}$ and $\mathbf{a}$, i.e., $w = \|\mathbf{c} - \mathbf{a}\|$.

Similarly, one can find

$$S(\mathbf{x}, \mathbf{b}, w) = 1 - 2(1/2 - \|\mathbf{x} - \mathbf{b}\|/w)^2 \quad \text{if } \|\mathbf{b}\| \leq \|\mathbf{x}\| \leq \|\mathbf{c}\|.$$

Therefore, when $\mathbf{x} \in R^n$

$$\begin{aligned}
 S(\mathbf{x}, \mathbf{b}, w) &= 0 && \text{if } \|\mathbf{x}\| \leq \|\mathbf{a}\| \\
 &= 2(1/2 - \|\mathbf{b} - \mathbf{x}\|/w)^2 && \text{if } \|\mathbf{a}\| \leq \|\mathbf{x}\| \leq \|\mathbf{b}\| \\
 &= 1 - 2(1/2 - \|\mathbf{x} - \mathbf{b}\|/w)^2 && \text{if } \|\mathbf{b}\| \leq \|\mathbf{x}\| \leq \|\mathbf{c}\| \\
 &= 1 && \text{if } \|\mathbf{c}\| \leq \|\mathbf{x}\|
 \end{aligned} \tag{2.22}$$

with ' $\mathbf{b}$ ' as the crossover point and $w = \|\mathbf{c} - \mathbf{a}\|$.

2.6.2 Histogram thresholding

It has been shown in *Note 2* that $C(X_f, \bar{X}) \geq 0$. So maximization of $C(X_f, \bar{X})$ is equivalent to minimization of (from equation 2.15)

$$T = \frac{4}{X_1 + X_2} \sum_{i=1}^L \sum_{j=1}^L \{[\mu_1(i, j) - \mu_2(i, j)] C(i, j)\}^2.$$

Now X_2 is constant and $(X_1 + X_2)$ is very large. The rate of change of $(X_1 + X_2)$ is very less. So, the above expression decreases with decrease of $|\mu_1(i, j) - \mu_2(i, j)|$, and so does the corresponding $C(i, j)$. When the threshold corresponds to the appropriate boundary between two regions the value of $C(i, j)$ for $i \leq s, j \leq s$ or $i > s, j > s$ (i.e., a pixel from one region followed by another pixel from the same region) would be more, whereas the value of $C(i, j)$ for $i \leq s, j > s$ or $i > s, j \leq s$ (i.e., a pixel from one region followed by a pixel from another region) will be less. So the number of co-occurrences of levels, $C(i, j)$, with low value of $|\mu_1(i, j) - \mu_2(i, j)|$ will be more, and with higher value of $|\mu_1(i, j) - \mu_2(i, j)|$ will be less, thus contributing less toward T .

Hence the minimization of T , i.e., maximization of $C(X_f, \bar{X})$ may be viewed as a criterion for detecting the thresholds between regions of X .

Although maximization of equation (2.15) has been shown above to be a criterion of thresholding, one can also establish

$$C(X_f, \bar{X})_O + C(X_f, X)_B \tag{2.23}$$

$$C(X_f, \bar{X})_{O/B} + C(X_f, \bar{X})_{B/O} \quad (2.24)$$

as a criterion for thresholding. The algorithm for their computation is same as that of **Algorithm 1**.

The algorithm using equation (2.15) as the thresholding criterion will be referred to as **Algorithm 2**. Similarly, equations (2.23) and (2.24) correspond to **Algorithm 3** and **Algorithm 4**, respectively.

If the global maximum value of correlation obtained is $C(X_f^g, \bar{X}^g)$ then X_f^g and $\bar{X}^g$ can be taken as a fuzzy and nonfuzzy segmented version of the image respectively.

Use of local information is expected to yield the boundary between regions even when the corresponding valley is not present in the histogram of the image.

2.7 (Dis)Similarity between fuzzy correlation and quadratic index of fuzziness

The quadratic index of fuzziness of a fuzzy set A having n supports is defined as [90] (for details see Appendix A.2.3)

$$\nu_q(A) = \frac{2}{\sqrt{n}} \sqrt{\left[\sum_{i=1}^n \{ \mu_A(x_i) - \mu_{\bar{A}}(x_i) \}^2 \right]}. \quad (2.25)$$

Such a measure of an image $X(M \times N)$ (considering the image as a fuzzy set, and thus $n = M \times N$) reflects the average amount of ambiguity (fuzziness) present in it by measuring the distance (quadratic) between its fuzzy property plane μ_X and the nearest two level property $\mu_{\bar{X}}$; in other words, the distance between the gray tone image and its nearest two tone version. Its mathematical expression is [93]

$$\nu_q(X) = \frac{2}{\sqrt{MN}} \sqrt{\sum_{m=1}^M \sum_{n=1}^N \{ \mu_X(x_{mn}) - \mu_{\bar{X}}(x_{mn}) \}^2}$$

$$= \frac{2}{\sqrt{MN}} \sqrt{\sum_{i=1}^L \{\mu_X(i) - \mu_{\bar{X}}(i)\}^2 h(i)}, \quad (2.26)$$

where $\mu_X(x_{mn})$ (or $\mu_X(i)$) represents the grade of possessing some property μ by the (m, n) th pixel (i th gray level), and $\mu_{\bar{X}}(x_{mn})$ (or $\mu_{\bar{X}}(i)$) denotes the nearest two tone version of $\mu_X(x_{mn})$ (or $\mu_X(i)$).

If we consider spatial information in the membership function, then the expression of $\nu_q(X)$ takes the form

$$\begin{aligned} \nu_q(X) &= \frac{2}{\sqrt{MN}} \sqrt{\sum_{m=1}^M \sum_{n=1}^N \{\mu(m, n) - \bar{\mu}(m, n)\}^2} \\ &= \frac{2}{\sqrt{MN}} \sqrt{\sum_{i=1}^L \sum_{j=1}^L \{\mu(i, j) - \bar{\mu}(i, j)\}^2 C(i, j)}. \end{aligned} \quad (2.27)$$

The quadratic index of fuzziness of an image can also be computed considering the various quadrants C_1 , C_2 , C_3 and C_4 of the co-occurrence matrix.

In this section we will establish a relation between $\nu_q(X)$ and $C(X_f, \bar{X})$.

Let s be the cross over point of the membership function $\mu_X(i)$, $i = 1, 2, \dots, L$, i.e., $\mu_X(i) \leq 0.5$ for $i \leq s$ and $\mu_X(i) \geq 0.5$ for $i \geq s$. From equation (2.26)

$$\begin{aligned} \nu_q(X) &= \frac{2}{\sqrt{MN}} \sqrt{\left[\sum_{i=1}^s \mu_X(i)^2 h(i) + \sum_{i=s+1}^L [1 - \mu_X(i)]^2 h(i) \right]} \\ \text{i.e., } \nu_q(X)^2 &= \frac{4}{MN} \left[\sum_{i=1}^s \mu_X(i)^2 h(i) + \sum_{i=s+1}^L [1 - \mu_X(i)]^2 h(i) \right] \\ \text{or, } MN \times \nu_q(X)^2 &= 4 \left[\sum_{i=1}^s \mu_X(i)^2 h(i) + \sum_{i=s+1}^L [1 - \mu_X(i)]^2 h(i) \right]. \end{aligned}$$

Now from equation (2.10) we get

$$\begin{aligned} 1 - C(X_f, \bar{X}) &= \frac{4}{X_1 + X_2} \left\{ \sum_{i=1}^s \mu_X(i)^2 h(i) + \sum_{i=s+1}^L [1 - \mu_X(i)]^2 h(i) \right\} \\ &= \frac{MN}{X_1 + X_2} \nu_q(X)^2 \end{aligned}$$

$$= \frac{X_2}{X_1 + X_2} \nu_q(X)^2 \quad [\text{since } X_2 = MN]$$

$$\text{or, } C(X_f, \bar{X}) = 1 - \frac{X_2}{X_1 + X_2} \nu_q(X)^2.$$

$$\text{Now } \frac{X_2}{X_1 + X_2} \leq 1.$$

$$\text{Therefore } C(X_f, \bar{X}) \geq 1 - \nu_q(X)^2. \quad \clubsuit$$

Note : If the window size w of the membership function is very small, then

$$\frac{X_2}{X_1 + X_2} \approx 1/2 \text{ as } X_1 \approx MN.$$

$$\text{Hence } \nu_q(X)^2 \approx 2\{1 - C(X_f, \bar{X})\}.$$

In the limiting case as $w \rightarrow 0$ (i.e., μ_1 becomes a step function), $C(X_f, \bar{X}) \rightarrow 1$ and $\nu_q(X) \rightarrow 0$.

From the above discussions it is evident that the thresholds obtained by minimization of $\{1 - C(X_f, \bar{X})\}$, i.e., maximization of $C(X_f, \bar{X})$ will be more or less the same as those obtained by minimization of ν_q . This has been experimentally verified and is shown in Tables 2.1-2.8.

Similar relation can also be established using local information in the expressions for the membership functions and $C(i, j)$.

2.8 Implementation and results

The proposed algorithms have been implemented on three different images, namely, a *Biplane* [Figure 2.2(a)], *Lincoln* [Figure 2.3(a)] and a *Blurred Chromosome* [Figure 2.4(a)]. The *Biplane* and *Lincoln* images have size 64×64 , level 32 and black object on white background, whereas the *Blurred Chromosome* image has size 64×64 , level 64 and white object on black background. The corresponding histograms are shown in Figures 2.2(b), 2.3(b) and 2.4(b).

For the computation of correlation and quadratic index of fuzziness (with local information) the membership value for a particular combination of gray values (i, j) of the co-occurrence matrix was assigned as follows :

- For a particular threshold s , any combination of pixel intensities s was assigned a value of 0.5, since (s, s) is the most ambiguous point, i.e., the boundary.
- If one object pixel is followed by another object pixel (i.e., the entries of quadrant C_3 or C_1), then its degree of belonging to object (membership value) is greater than 0.5. The membership value increases with increase in pixel intensity.
- For quadrants C_2 and C_4 where one object pixel is followed by one background pixel or vice versa, the membership value is less than or equal to 0.5, depending on the deviation from the boundary point (s, s) .
- If one background pixel is followed by another background pixel (i.e., for the entries in C_1 or C_3), then its degree of belonging to object (membership value) is less than 0.5. The membership value decreases with decrease of pixel intensity.

The thresholds obtained for different window sizes by the proposed algorithms are given in Tables 2.1-2.8. (Thresholds with superscript * denote global optima.) Tables 2.1 and 2.2 show the thresholds obtained by Algorithm 1 which considers only global information. Tables 2.3-2.8 contain the thresholds obtained by Algorithms 2, 3 and 4. The thresholds obtained by the minimization of the quadratic index of fuzziness of an image (with both global and local information) are also included in the tables for a comparative study. It is to be noticed here that the thresholds obtained by the index of fuzziness minimization, as expected, are more or less the same as those obtained by correlation maximization. Note that for algorithms involving local information the window sizes are larger.

For visual inspection, non-fuzzy segmented versions of the images [corresponding to the global maxima of $C(X_f, \bar{X})$] for a fixed window size ($w = 10$ for

Table 2.1: Thresholds based on Algorithm 1

Window size	<i>Biplane image</i>		<i>Lincoln image</i>	
	Correlation Maximization (Algorithm 1)	Fuzziness Minimization	Correlation Maximization (Algorithm 1)	Fuzziness Minimization
6	12* 15 23 31	12* 15 22 31	7 11* 14 18 25 31	7 11* 18 25 31
8	12* 23 31	12* 23 31	11* 18 25 31	11* 18 25 31
10	12* 22 31	12* 22 31	10* 18 25 31	10* 18 25 31
12	13* 22 31	13* 22 31	10* 18 25 31	10* 18 25 31

Table 2.2: Thresholds based on Algorithm 1

Window size	<i>Blurred Chromosome image</i>	
	Correlation Maximization (Algorithm 1)	Fuzziness Minimization
6	16 19 30 36 40* 43 46	16 19 30 36 40* 43 46
8	19 30 36 40* 43 46	19 30 36 40* 43 46
10	19 30 40* 43 46	19 30 40* 43 46
12	19 30 40* 46	19 30 40* 43 46

Table 2.3: Thresholds based on Algorithm 2

Window size	<i>Biplane image</i>		<i>Lincoln image</i>	
	Correlation Maximization (Algorithm 2)	Fuzziness Minimization	Correlation Maximization (Algorithm 2)	Fuzziness Minimization
12	13* 23 30	13* 15 23 31	10* 14 18 25 30	10* 18 25 31
16	13* 22 30	13* 22 31	10* 18 25 30	10* 18 25 31
20	13* 31	13* 31	9* 18 25 30	10* 18 25 31
24	14* 31	14*	9* 17 25 30	10* 17 25

Table 2.4: Thresholds based on Algorithm 2

Window size	<i>Blurred Chromosome image</i>	
	Correlation Maximization (Algorithm 2)	Fuzziness Minimization
12	18* 22 56	18* 22 56
16	18* 21 56	18* 21 56
20	21* 56	21* 56
24	21* 56	21* 56

Table 2.5: Thresholds based on Algorithm 3

Window size	<i>Biplane image</i>		<i>Lincoln image</i>	
	Correlation Maximization (Algorithm 3)	Fuzziness Minimization	Correlation Maximization (Algorithm 3)	Fuzziness Minimization
12	13* 23 30	13* 15 23 31	10* 14 18 25 30	9* 14 18 25 31
16	13* 22 30	13* 31	9* 14 18 25 30	9* 18 25 31
20	13* 31	13* 31	9* 18 25 30	9* 18 25 31
24	14* 31	13* 31	9* 17 25 30	9* 17 26 31

Table 2.6: Thresholds based on Algorithm 3

Window size	<i>Blurred Chromosome image</i>	
	Correlation Maximization (Algorithm 3)	Fuzziness Minimization
12	19 31 37 41* 46	19 31 37 41* 46
16	20 31 40* 46	20 31 40* 46
20	20 31 40*	20 31 40*
24	20 40*	20 40*

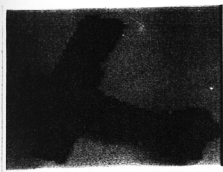
Table 2.7: Thresholds based on Algorithm 4

Window size	<i>Biplane image</i>		<i>Lincoln image</i>	
	Correlation Maximization (Algorithm 4)	Fuzziness Minimization	Correlation Maximization (Algorithm 4)	Fuzziness Minimization
12	11* 14 19 22 30	11* 14 22 30	6 10* 13 18 24 31	6 10* 13 18 24 31
16	11* 14 22 30	11* 14 22 30	6 10* 13 18 24	6 10* 13 18 31
20	11* 14 22 30	11* 14 22	6 10* 13 18 24	6 10* 13 18 31
24	11* 14 22 30	11* 14 22	6 10* 13 18 24	6 10* 13 18 31

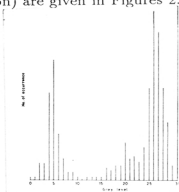
Table 2.8: Thresholds based on Algorithm 4

Window size	<i>Blurred Chromosome image</i>	
	Correlation Maximization (Algorithm 4)	Fuzziness Minimization
12	17 30 35 38* 42 45 58	17 30 35 38* 42 45
16	17 30 35 38* 42 45 58	17 30 35 38* 42
20	17 30 35 38* 42 45 58	17 30 35 38* 42
24	17 30 35 38* 42 45 58	17 30 35 38* 42

global information, $w = 16$ for local information) are given in Figures 2.2-2.4.



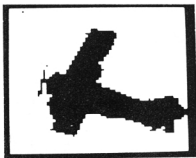
(a)



(b)



(c)



(d)

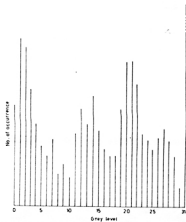
Figure 2.2: *Biplane* image : a) input, b) histogram, c) threshold=12, d) threshold=13

The histogram of the image of *Biplane* has a sharp valley in the gray level range 11-16, a weak valley in the range of 21-26 and a spurious valley at 31. From the results (thresholds) obtained by the proposed algorithms, it is evident that for all window sizes most of the valleys are detected. For smaller window sizes, more valleys are detected in the range 11-16 which is in agreement with the earlier findings of Murthy and Pal [260]. The valley (12 or 13) corresponding to the maximum (global) correlation is seen to be the appropriate threshold for object background separation.

The histogram of *Lincoln* image has four sharp valleys and one or two weak val-



(a)

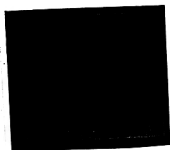


(b)

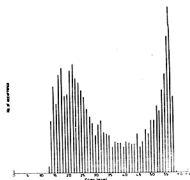


(c)

Figure 2.3: *Lincoln* image : a) input, b) histogram, c) threshold=10



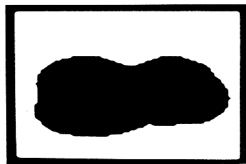
(a)



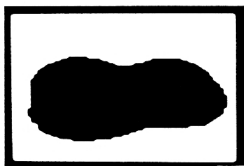
(b)



(c)



(d)



(e)

Figure 2.4: *Blurred Chromosome* image : a) input, b) histogram, c) threshold=18, d) threshold=38, e) threshold=40

leys, whereas the histogram of *Blurred Chromosome* image has many (including one or two sharp) valleys over the entire histogram. Here also it is noticed that for smaller window sizes more valleys are detected and for higher window sizes less valleys are detected. The global threshold (10 or 11) obtained for object background separation of the *Lincoln* image is seen to be appropriate. For the *Blurred Chromosome* image, the global thresholds obtained do not seem to be much appropriate. This will be evident from the results given in the next chapter.

A curve showing the variation of fuzzy correlation with cross over point (using global information) for a fixed window size (6) of the image of *Lincoln* is given in Figure 2.5. From the figure it is evident that the nature of the graph is complementary to the *sharpened* version of the histogram.

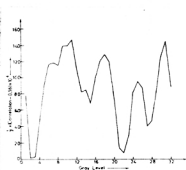


Figure 2.5: Variation of fuzzy correlation with cross over points (using global information) for the image of *Lincoln* and a window size of 6

Besides comparing with the fuzziness-minimization based method, the proposed algorithms have also been compared with three co-occurrence matrix based classical techniques, which use second order local entropy, second order conditional entropy [64], and the product of contrast between regions and homogeneity within regions [62] as objective functions. The basic concept of these algorithms is to determine a set of threshold values for which these objective measures have local maxima. The global thresholds obtained for object background classification by these methods are shown in Table 2.9. It is seen that

the global thresholds provided by the proposed algorithms involving *transitional correlation* (Algorithm 4) and *within class correlation* (Algorithm 3) agree well with those obtained by the contrast-homogeneity measure and the conditional entropy. The conditional entropy provides a measure of between-class contrast by taking the information on the submatrices C_2 and C_4 . The local entropy, on the other hand, takes care of within class information using the submatrices C_1 and C_3 , and provides inferior results.

Table 2.9: Thresholds with some co-occurrence matrix based methods

Images	Contrast-homogeneity based method	Second-order local entropy based method	Second-order conditional entropy based method
<i>Biplane</i>	11	23	13
<i>Lincoln</i>	10	18	11
<i>Blurred Chromosome</i>	40	34	42

A comparative study of the thresholds obtained with the proposed local information based correlation measures (Algorithms 2, 3 and 4) show that the transitional correlation (using the submatrices C_2 and C_4) is most effective in determining the boundaries between regions in an image. This is in agreement with the earlier findings in [64] which has been mentioned in the previous paragraph. This further establishes that intersets distance (contrast between regions) is more important than within-class distance (homogeneity) in separating regions.

Furthermore, it is evident from the results that incorporation of local information makes it more effective in detecting boundaries between regions. For instance, let us consider the thresholds of *Lincoln* image detected by the *transitional correlation* (Table 2.7). Here the boundaries (valleys of the histogram) at gray levels 6 and 13 are detected as local maxima for almost all window sizes. This is not the case with the global-information based algorithm (for most of the window sizes). Similar conclusion can be made from the results of *Blurred Chromosome* image (Table 2.8).

2.9 Conclusions and discussion

An attempt has been made to apply the concept of correlation between two fuzzy membership functions in image segmentation (and thereby object background classification) problems. In this connection various fuzzy correlation measures (*transitional correlation* and *within-class correlation*) have been proposed and four algorithms are formulated involving both global and local information. Mathematical framework of the algorithms showing that the maximization of fuzzy correlation provides a criterion of detecting valleys in a histogram has also been given.

It is to be mentioned here that the concept of correlation between two membership functions as introduced in [120] neither included the number of supports in a set nor considered their order of occurrence. In the present work, both the number of occurrences of levels and their spatial relation have been taken into account (in defining the correlation between two properties of an image). A relation between fuzzy correlation and quadratic index of fuzziness of an image has also been established.

Incorporation of spatial information, as expected, makes the algorithms more efficient in detecting boundaries. *Transitional correlation* is seen to provide better results. It has also been noticed that with the variation of window size the changes of threshold values are insignificant. This makes the algorithms flexible.

It may be noted that the proposed fuzzy correlation measures involve information on both histogram and co-occurrence matrix of an image for gray level thresholding; and reflect the gray level ambiguity present in the image. No geometrical concept of image subset (regions) has been used to extract the segments. In the next chapter we shall be defining a set of fuzzy geometrical parameters/properties (which reflect spatial ambiguities present in an image) to extract segmented outputs using both gray level thresholding and pixel classification.

Chapter 3

**Gray level thresholding and
pixel classification using fuzzy
geometry**

3.1 Introduction

In pattern recognition and image analysis we often want to measure geometrical properties of regions in an image that are not crisply defined. Many of the standard geometrical properties of and relationship among regions can be generalized to fuzzy subsets. There have been a great deal of work in this regard by Rosenfeld [102, 103], [261]-[264] who has made these generalizations and has extended the concept of digital picture geometry to fuzzy subsets. Such an extension, called fuzzy geometry, includes topological concept of connectedness, surroundedness, convexity, area, perimeter, compactness, height, width, diameter, extent, elongatedness, adjacency and degree of adjacency. After generalizing the standard geometric properties, it becomes possible to use these definitions to construct image description without committing to a specific segmentation of an image. These measures have also been found to reflect the spatial (geometrical) ambiguity of an image and thus seem to be useful in providing soft decision for image description and analysis. Recently fuzzy medial axis has been defined in [265, 266] in this context using the concept of fuzzy disk. In [267, 268] attempts are also made to represent spatial relationship between fuzzy subsets of an image. Further references can be found in [269]-[272].

The work presented in this chapter consists of two parts. In the first part some new fuzzy geometrical properties namely, length, breadth, index of area coverage (*IOAC*), major axis, minor axis, center of gravity and density along with their definitions and illustrations are provided. The concepts of Rosenfeld's *adjacency* and *degree of adjacency* [102] are critically examined and their new versions defined. The second part consists of defining various algorithms based on these properties for providing both fuzzy and nonfuzzy segmented outputs through gray level thresholding and pixel classification.

The measures involving spatial information have been evaluated either in terms of co-occurrence matrix or in terms of local histogram computed over each row and each column. This in turn makes their conceptual realization easier and computation faster.

Standard 'S' function (equation 2.11, pp. 46) and its complement are used for extracting the *bright image* plane $\mu(X)$ from an image X for its object extraction. The parameters *IOAC* and degree of adjacency are optimized for gray level thresholding. Adjacency measure and membership value of individual pixels computed over $\mu(X)$ are considered for pixel classification. The effectiveness of the algorithms has been demonstrated on various input images having wide range of histograms and compared with some of the recently developed algorithms [64, 116].

3.2 Existing fuzzy geometrical properties

Let $\mu(X)$ denote a fuzzy representation of an $M \times N$ image X , i.e., a mapping μ from X into $[0, 1]$ representing a fuzzy subset of X . For convenience, we shall use μ only to denote $\mu(X)$ in this chapter.

The various geometrical properties of μ as defined by Rosenfeld [102] are given below with illustration.

Area : The area of a fuzzy subset μ is defined as

$$a(\mu) = \int \mu \quad (3.1)$$

where the integration is taken over a region outside which $\mu = 0$. For μ being piece-wise constant (in case of digital image) the area is

$$a(\mu) = \sum \mu, \quad (3.2)$$

the summation being considered over a region outside which $\mu = 0$. The area is therefore the weighted sum of the regions on which μ has constant value weighted by these values.

Example 1 : Let μ be of the form

$$\begin{pmatrix} 0.2 & 0.4 & 0.3 \\ 0.2 & 0.7 & 0.6 \\ 0.6 & 0.5 & 0.6 \end{pmatrix}.$$

Then area $a(\mu) = 4.1$.

Perimeter : If μ is piece-wise constant, the perimeter of μ is defined as

$$p(\mu) = \sum_{i,j,k} |\mu(i) - \mu(j)| \times |A(i, j, k)|. \quad (3.3)$$

This is just the weighted sum of the lengths of the arcs $A(i, j, k)$ along which the regions having μ values $\mu(i)$ and $\mu(j)$ meet, weighted by the absolute difference of these values. In case of an image if we consider the pixels as the piece-wise constant regions, and the common arc length for adjacent pixels as unity then the perimeter of an image is defined by

$$p(\mu) = \sum_{i,j} |\mu(i) - \mu(j)| \quad (3.4)$$

where $\mu(i)$ and $\mu(j)$ are the membership values of two adjacent pixels. For the fuzzy subset μ of *Example 1*, perimeter $p(\mu) = 2.3$.

Compactness : The compactness of a fuzzy set μ having area $a(\mu)$ and perimeter $p(\mu)$ is defined as

$$comp(\mu) = \frac{a(\mu)}{p^2(\mu)}. \quad (3.5)$$

Physically, compactness means the fraction of maximum area (that can be encircled by the perimeter) actually occupied by the fuzzy region/concept represented by μ . In the nonfuzzy case, the value of compactness is maximum for a circle and this value is $\frac{1}{4\pi}$. In case of a fuzzy disc, where the membership value is only dependent on its distance from the center, the compactness value is $\geq \frac{1}{4\pi}$ [102]. Of all possible fuzzy discs compactness is therefore minimum for its crisp version. For the fuzzy subset μ of *Example 1*, $comp(\mu) = 0.775$.

Height and Width : The height $h(\mu)$ and width $w(\mu)$ of a fuzzy set μ are defined as

$$h(\mu) = \int \max_x \{\mu(x, y)\} dy \quad (3.6)$$

and

$$w(\mu) = \int \max_y \{\mu(x, y)\} dx \quad (3.7)$$

where the integration is taken over a region outside which $\mu(x, y) = 0$. For a digital picture the definitions take the form

$$h(\mu) = \sum_y \max_x \{\mu(x, y)\} \quad (3.8)$$

and

$$w(\mu) = \sum_x \max_y \{\mu(x, y)\}. \quad (3.9)$$

So, height (width) of a digital picture is the sum of the maximum membership values of each row (column). For the fuzzy subset μ of *Example 1*, the height is $h(\mu) = 0.4 + 0.7 + 0.6 = 1.7$ and the width is $w(\mu) = 0.6 + 0.7 + 0.6 = 1.9$.

Adjacency : Let μ and ν be two piece-wise constant fuzzy subsets of a set S . Then we can partition S into a finite number of bounded regions $B(i)$, meeting pair-wise along arcs, on each of which both μ and ν are constant, say having values $\mu(i)$ and $\nu(j)$. Let μ and ν be disjoint; then on each $B(i)$, either $\mu = 0$ or $\nu = 0$. Let $A(i, j, k)$ be the k th arc along which $B(i)$ and $B(j)$ meet, and let this arc length be $|A(i, j, k)|$. Then the adjacency between μ and ν is defined as

$$a(\mu, \nu) = \sum_{i,j,k} \mu(i) \times \nu(j) \times |A(i, j, k)|. \quad (3.10)$$

Evidently, if μ and ν are crisp, $a(\mu, \nu)$ is just the total length of their common border. In the fuzzy case, each arc length is weighted by the product $\mu(i) \times \nu(j)$. $a(\mu, \nu)$ is 1 only when both $\mu(i)$ and $\nu(j)$ are 1 (full adjacency), and it is small when either of them is small (i.e., either of them is weakly present). In case of a digital picture, the adjacency of a pixel to one of its neighbors is the product of its membership value with that of its corresponding neighbor. For the center pixel of *Example 1*, adjacency is 0.42 with its right neighbor.

Degree of Adjacency : Intuitively, two regions S and T in a digital picture are somewhat adjacent if some of the border pixels of S touch (nearly) some of the border pixels of T ; the degree of adjacency depends on how nearly they touch and along how much of their length they do so. Let $BP(S)$ be the set of border pixels of S . Take any border pixel $p \in BP(S)$. Let $d(p)$ be the shortest distance of this pixel from the border of T . Then the degree of adjacency

between S and T is defined as

$$a(S, T) = \sum_{p \in BP(S)} \frac{1}{1 + d(p)}. \quad (3.11)$$

3.3 Some new fuzzy geometrical properties

In this section we are going to introduce some new fuzzy geometric properties of image subsets.

Length : The length of a fuzzy set μ may be defined as

$$l(\mu) = \max_x \left\{ \int \mu(x, y) dy \right\} \quad (3.12)$$

where the integration is taken over the region outside which $\mu(x, y) = 0$. In case of a digital picture the expression takes the form

$$l(\mu) = \max_x \left\{ \sum_y \mu(x, y) \right\}. \quad (3.13)$$

Breadth : The breadth of a fuzzy set μ may be defined as

$$b(\mu) = \max_y \left\{ \int \mu(x, y) dx \right\} \quad (3.14)$$

where the integration is taken over the region outside which $\mu(x, y) = 0$. For a digital image

$$b(\mu) = \max_y \left\{ \sum_x \mu(x, y) \right\}. \quad (3.15)$$

The length (breath) of an image fuzzy subset gives its longest expansion in the y direction (x direction). If μ is crisp, $\mu(x, y) = 0$ or 1 ; then length (breadth) is the maximum number of pixels in a column (row). Comparing equation (3.13) with (3.8) and (3.15) with (3.9) we notice that the length (breadth) takes the summation of the entries in a column (row) first and then maximizes over different columns (rows), whereas the height (width) maximizes the entries in a column (row) and then sums over different columns (rows). For the fuzzy subset μ in *Example 1*, $l(\mu) = 1.6$ and $b(\mu) = 1.7$.

Index of Area Coverage (IOAC) : The index of area coverage of a fuzzy set may be defined as

$$IOAC(\mu) = \frac{a(\mu)}{l(\mu) \times b(\mu)}. \quad (3.16)$$

In the nonfuzzy case, the *IOAC* has a value of 1 for a rectangle (placed along the axes of measurement). For a circle this value is $\pi r^2 / (2r \times 2r) = \pi/4$. *IOAC* of a fuzzy image subset represents the fraction (which may be improper also) of the maximum area (that can be covered by the length and breadth of the image) actually occupied by the image.

For the fuzzy subset μ of *Example 1*, the maximum area that can be covered by its length and breadth is $1.6 \times 1.7 = 2.72$, whereas the actual area is 4.1, so the $IOAC = 4.1/2.72 = 1.51$. Note the difference between *IOAC* and *comp*(μ). Again, note the following relationships

$$\begin{aligned} l(\mu)/h(\mu) &\leq 1, \\ b(\mu)/w(\mu) &\leq 1. \end{aligned} \quad (3.17)$$

Major Axis : Find the length of the fuzzy image subset. Now rotate the axes through an angle θ , ($0^\circ \leq \theta \leq 90^\circ$). The angle for which length is maximum is said to be the angle of inclination of the object (with the vertical). The corresponding axis along which the length is maximum is said to be the major axis. The length along the major axis denotes the expansion of the object.

Minor Axis : The axis perpendicular to major axis, for which breadth is maximum is defined as the minor axis of the object.

Center of Gravity : The center of gravity (*C.G.*) of a fuzzy image subset can be defined in various ways. Two such definitions are given here.

- *C.G.* of a fuzzy image subset can be defined as the point of intersection of the major and the minor axes.
- Take any pixel as the center. Take a neighborhood of radius r . Find the energy (area) of the circle. Now shift the center of the circle over all the pixels of the image. The center for which the energy is maximum

is defined as the *C.G.* . If there is any tie then increase the radius and obtain the *C.G.* .

For the fuzzy subset μ of *Example 1*, length $l(\mu) = 1.6$ and breadth $b(\mu) = 1.7$. Now if we rotate the object by 45° then its length is $l(\mu) = 0.6 + 0.7 + 0.6 = 1.9$

Hence the fuzzy object is inclined at an angel of 45° with the vertical axis. So by the major axis of this image subset we mean the axis inclined at an angle of 45° with the vertical. Similarly, the minor axis of this image subset is inclined at an angle of 45° with horizontal. Trivially, the *C.G.* of this image subset passes through the pixel having membership 0.7.

Density : The density of a fuzzy set μ having N supports may be defined as

$$d(\mu) = \frac{1}{N} \int \mu = \frac{1}{N} a(\mu). \quad (3.18)$$

For a digital image (having N pixels) characterized by μ , its density is

$$d(\mu) = \frac{1}{N} \sum_{i=1}^N \mu(i). \quad (3.19)$$

The maximum value of density is 1 and this value occurs only for a non-fuzzy case. Density can be used for finding the *C.G.* of an image. If we break the image into different regions of same size then the region having the maximum density may be regarded to contain the *C.G.* .

Adjacency : The definition of adjacency given in equation (3.10) does not fully agree with our intuition. Let us take two piece-wise constant regions μ and ν . We say μ and ν are more adjacent if the difference of the membership values of μ and ν are less (whatever be their actual values). For example, let us take the membership values of the regions μ and ν as 0.2 and 0.25 respectively in the first case, and 0.8 and 0.85 in the second case. Since the difference of membership values is the same in both the cases, their adjacency is also expected to be same (assuming the length of their common border to be the same). But according to equation (3.10) this value is 0.05 in the first case whereas 0.68 in the second case. One may, therefore, defined adjacency as

$$a(\mu, \nu) = \sum_{i,j,k} \frac{|A(i, j, k)|}{1 + |\mu(i) - \nu(j)|}. \quad (3.20)$$

When $\mu(i) = \nu(j)$ and the length of the common boundary between the regions is unity the adjacency value is maximum ($=1$). The lesser the difference between μ and ν , the more is the adjacency and vice versa.

Degree of Adjacency : The definition of the degree of adjacency (equation 3.11) gives a measure of physical adjacency of two regions S and T . It does not say anything about their gray values. In practice, it is reasonable to determine adjacency between two regions by their both physical and gray level distances. So the degree of adjacency can be defined as

$$a(S, T) = \sum_{p \in BP(S)} \frac{1}{1 + |\mu(p) - \nu(q)|} \times \frac{1}{1 + d(p)}. \quad (3.21)$$

Here $d(p)$ is the shortest distance between p and q , q is a border pixel of T and p is a border pixel of S . The other symbols are having their same meaning as in the previous discussion. $a(S, T)$ denotes the degree to which the regions S and T are adjacent. The degree of adjacency of two regions is maximum ($=1$) only when they are physically adjacent, i.e., $d(p) = 0$ and their membership values are also equal; i.e., $\mu(p) = \nu(q)$. If two regions are physically adjacent (i.e., $d(p) = 0, \forall p$) then their degree of adjacency is determined only by the difference of their membership values. Similarly, if the membership values of two regions are equal their degree of adjacency is determined by their physical distance only.

It is to be noted that when $d(p) = 0, \forall p$ (i.e., when S is surrounded by T or vice versa), then $|A(i, j, k)| = 1$ (because p and q are adjacent) and equations (3.21) and (3.20) become identical.

9GKPSTVPNGDC78
 9GJKSTVTVJCA88
 96HJOVUMTRPHA8
 9AJNSUVMKGGA978
 9GHKMSTVRPHCA8
 A549HKOTTSNKGC8
 AFGKLPSUTPMJGF8
 AFSHSTVUJHCA998
 ADEGKOVSSJDA988
 AAHKRTMJFA89876
 ABJOSTTVUSKFGA8
 AAGHKUVRMHFC968 (a)

9GKPSTVPNGDC78
 9GJKSTVTVJCA88
 96HJOVUMTRPHA8
 9AJNSUVMKGGA978
 9GHKMSTVRPHCA8
 A549HKOTTSNKGC8
 AFGKLPSUTPMJGF8
 AFSHSTVUJHCA998
 ADEGKOVSSJDA988
 AAHKRTMJFA89876
 ABJOSTTVUSKFGA8
 AAGHKUVRMHFC968 (b)

9GKPSTVPNGDC78
 9GJKSTVTVJCA88
 96HJOVUMTRPHA8
 9AJNSUVMKGGA978
 9HKPTSTMKFDCA8
 A549HKOTTSNKGC8
 AFGKLPSUTPMJGF8
 AFSHSTVUJHCA998
 ADEGKOVSSJDA988
 AAHKRTMJFA89876
 ABJOSTTVUSKFGA8
 AAGHKUVRMHFC968 (c)

Figure 3.1: a) A scene with two regions; b) same as in (a), the relative positions of the regions are different; c) same as in (b), the gray levels of the border pixels of the first region are different

Example 2 : Let us consider a scene having two regions only (Figure 3.1). The gray levels of the scene are coded as $A = 10, B = 11, \dots, V = 32$. The gray levels of the regions in Figure 3.1(a) and Figure 3.1(b) are identical but the regions in Figure 3.1(b) are closer to each other compared to Figure 3.1(a). So the degree of adjacency of the regions in Figure 3.1(b) should be more than that in Figure 3.1(a). In Figures 3.1(b) and 3.1(c) the relative positions of the regions are the same, but the difference of gray levels of the border pixels in Figure 3.1(b) is more than that in Figure 3.1(c). So from intuition we would expect that the degree of adjacency of the regions in Figure 3.1(c) is more than that in Figure 3.1(b).

The $a(S, T)$ values computed with equation (3.21) are 0.062, 0.143 and 0.189 for Figures 3.1(a), 3.1(b) and 3.1(c), respectively. So the results agree well with our intuition.

In the following sections we will describe how these fuzzy geometrical concepts can be incorporated in formulating various algorithms for gray level thresholding and pixel classification. For developing the algorithms we will be concentrating only on the parameters length, breadth, compactness, *IOAC*, adjacency and degree of adjacency.

3.4 Object background classification by degree of adjacency minimization

Equation (3.21) shows that, the higher the contrast (difference in membership values or gray levels) between two regions and greater the distance between them, the lower is their degree of adjacency. Therefore, a lower value of degree of adjacency implies that the segments (clusters) formed are more valid (separable) considering their both gray levels and physical distances. For this reason we will use minimization of degree of adjacency as a criterion for extracting segments of an image.

3.4.1 Formulation of algorithm (*Algorithm 1*)

Let us consider an L level digital image X of size $M \times N$ having gray values from $l_{\min}$ to $l_{\max}$. Let the object pixels have gray values $\leq s$ and background pixels $> s$, where $l_{\min} \leq s \leq l_{\max}$. In other words, we consider the regions object and background with respect to gray levels only. In such a case the degree of adjacency between the regions (object and background) is defined as (from equation 3.21)

$$a(O, B) = \sum_{i,j} \frac{1}{1 + |i - j|} \times \frac{1}{1 + d} \quad (3.22)$$

where i and j represent the gray values of an object pixel and a background pixel respectively such that j is a neighbor of i , and d is the physical distance of the pixel having gray level j from the pixel having gray level i .

Since we are considering only the border pixels (the transition between object and background) for computing the degree of adjacency between two regions (with respect to gray levels only), the physical distance of a border pixel of object region from that of background region is zero. So putting $d = 0$ in equation (3.22) we get

$$a(O, B) = \sum_{i,j} \frac{1}{1 + |i - j|} \quad (3.23)$$

where j is a neighbor of i , and if i comes from object then j comes from background. The information regarding i and j can be obtained from the co-occurrence matrix C of X as described in Chapter 2 (section 2.5.1, pp. 56).

From the co-occurrence matrix (Figure 2.1, pp. 56) we see that the entries in C_1 and C_3 contribute nothing to the degree of adjacency measure (equation 3.23); only those of C_2 and C_4 will contribute. Equation (3.23) thus takes the form (considering j as one of the four neighbors of i)

$$a(O, B) = \sum_{(i,j), i \leq s, j > s} \frac{C[i, j]}{1 + |i - j|} + \sum_{(i,j), i > s, j \leq s} \frac{C[i, j]}{1 + |i - j|} \quad (3.24)$$

where $C[i, j]$ represent the number of occurrences of the gray level i followed by gray level j .

From equation (3.24) it is evident that the value of $a(O, B)$ is less when the difference of i and j is more and the corresponding $C[i, j]$ is less. In an image with increase in $|i - j|$, $C[i, j]$ usually decreases and the elements in C thus correspond mainly to a low value of $|i - j|$. When s corresponds to the appropriate boundary (threshold) between two regions, there will be a minimum number of pixel intensities having low $C[i, j]$ ($i \leq s, j > s$, or $i > s, j \leq s$) value; thus contributing least towards $a(O, B)$. This optimum (minimum) value of $a(O, B)$ would be greater for any other choice of s . The concept may be written in algorithmic form as follows.

Step 1 : Take a threshold s , between l_{min} and l_{max} .

Step 2 : Calculate $a(O, B)$ using equation (3.24).

Step 3 : Iterate *Steps 1* and *2* for different values of s , select that s as an optimum threshold for which $a(O, B)$ is globally minimum.

3.4.2 Implementation and results

Three images are considered here namely, *Biplane*, *Lincoln*, and *Blurred Chromosome* [Figures 2.2(a), 2.3(a) and 2.4(a)]. The thresholds¹ obtained by the algorithms are shown in Table 3.1. Some of the thresholded (global) images are shown in Figures 3.2(c), 3.3(c) and 3.4(d) as a typical illustration. Note that the Figures 3.2-3.4 also contain some other segmented versions which are obtained by Algorithm 2 described later in this chapter. All the results are put together for the convenience of readers.

The results agree well with those of the recently developed algorithms based on co-occurrence matrix [62, 64] (which have been mentioned in Table 2.9, pp. 73) and those of the previous chapter (Tables 2.1-2.8).

An experiment is also done by including a measure called *total internal scatter* (based on C_1 and C_3 of Figure 2.1, pp. 56) within object and background

$$S(T) = S(O) + S(B) \quad (3.25)$$

¹Superscript * denotes global optimum

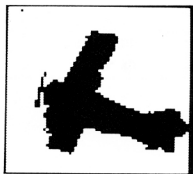
see Figure 2.2(a)

see Figure 2.2(b)

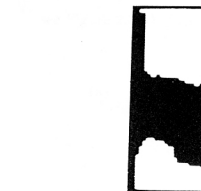
(a)



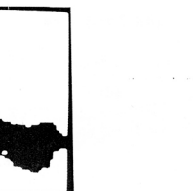
(b)



(c)



(d)



(e)

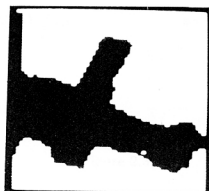


Figure 3.2: *Biplane* image : a) input, b) histogram, c) threshold=6, d) threshold=14, e) threshold=24

Table 3.1: Thresholds based on Algorithm 1

Images	Degree of adjacency	Degree of adjacency + Internal scattering
<i>Biplane</i>	11* 14 22	11 14* 22 30
<i>Lincoln</i>	3 5 10* 13 18 24	6 10* 13 16 18 24
<i>Blurred</i>	19 22 30 38	19 22 30
<i>Chromosome</i>	40* 43 45 53	38 40* 43

see Figure 2.3(a)

(a)

see Figure 2.3(b)

(b)

see Figure 2.3(c)

(c)

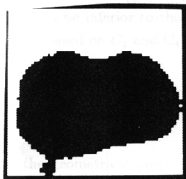
Figure 3.3: *Lincoln* image : a) input, b) histogram, c) threshold=10

see Figure 2.4(a)

see Figure 2.4(b)

(a)

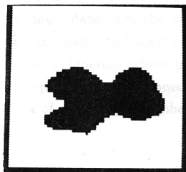
(b)



(c)

see Figure 2.4(e)

(d)



(e)



(f)

Figure 3.4: *Blurred Chromosome* image : a) input, b) histogram, c) threshold=32, d) threshold=40, e) threshold=54, f) threshold=56

where

$$S(O) = \sum_{(i,j), i,j \leq s} |i - j|, \quad (3.26)$$

and

$$S(B) = \sum_{(i,j), i,j > s} |i - j| \quad (3.27)$$

as an additional criterion, but it did not seem to change much (as seen from Table 3.1) the performance as obtained by equation (3.24) alone. This conforms well to the earlier findings (e.g., see [64] and the results of Chapter 2) where the second order local entropy and *within-class correlation* (based on C_1 and C_3) were found to be inferior to those of the conditional entropy and *transitional correlation* (based on C_2 and C_4).

3.5 Object extraction by optimizing IOAC

Most of the geometrical properties (of an image fuzzy subset) such as area, perimeter, length, breadth etc. are dependent not only on the shape of the image, but also on the membership values. If the object pixels have higher gray levels we can select the standard 'S' function (equation 2.11, pp. 46) to extract a *bright image* plane. On the other hand, when the object pixels have lower gray levels compared to background pixels, we would select a function complementary to 'S' function which may be referred as 'Z' function to represent a *dark image* plane. (As mentioned in section 2.4, pp. 46, these functions satisfy the criteria of selecting a membership function for image processing problems [260].)

3.5.1 Formulation of algorithm (*Algorithm 2*)

It has been noticed in section 3.3 that for crisp sets the value of index of area coverage (*IOAC*) is maximum for a rectangle. Again, of all possible fuzzy rectangles *IOAC* is minimum for its crisp version. For this reason, we will use minimization of *IOAC* as a criterion for image segmentation. Note that

a similar algorithm based on minimizing compactness measure was developed by Pal and Rosenfeld [116]. It has also been included here to compare its performance with that of the *IOAC* measure.

Criteria of fuzzy thresholding : Suppose we use the 'S' function (equation 2.11, pp. 46) for obtaining the *bright image* plane $\mu(X)$ of an image X . Then for a particular cross over point selected at, say, $b = s$, the pixels having gray levels $> s$ will have μ values > 0.5 and those having gray levels $< s$ will have μ values < 0.5 . This implies allocation of the gray levels into two regions. The term $IOAC(\mu)$ ($comp(\mu)$) then reflects the average amount of ambiguity in the geometry (i.e., in spatial domain) of X . Therefore, modification of the cross over point will result in different $\mu(X)$ planes (and hence different segmented versions), with varying amount of *IOAC* (compactness). The $\mu(X)$ plane having minimum *IOAC* (compactness) value can be regarded as an optimum fuzzy segmented version of X . This is optimum in the sense that for any other selection of crossover point b , the value of *IOAC* (compactness) will be greater. For obtaining the non-fuzzy output one may take the cross over point (which is considered to be the maximum ambiguous level) as the threshold between object and background. For images having multiple regions, one would have a set of such optimum $\mu(X)$ planes.

Faster method of computation : From Algorithm 2 it appears that one needs to scan an L level image L times (corresponding to L cross over points of the membership function) for computing the parameters for detecting its thresholds. The time of computation can be reduced significantly by scanning it only once for computing its co-occurrence matrix, row histogram and column histogram, and by computing $\mu(i), i = 1, 2, \dots, L$, every time with the membership function of a particular cross-over point.

Let $h(i), i = 1, 2, \dots, L$, be the number of occurrences of the level i , $C[i, j], i = 1, 2, \dots, L, j = 1, 2, \dots, L$, the (i, j) th entry of the co-occurrence matrix and $\mu(i), i = 1, 2, \dots, L$, the membership vector for a fixed cross over point of an L level image X . Then compute

$$a(\mu) = \sum_{i=1}^L \mu(i) \times h(i), \quad (3.28)$$

$$p(\mu) = \sum_{i=1}^L \sum_{j=1}^L |\mu(i) - \mu(j)| \times C[i, j]. \quad (3.29)$$

Let $R[m, l]$ represent the number of occurrences of the gray level l in the m th row of the image and $C[n, l]$ represent the number of occurrences of the gray level l in the n th column of the image. Then

$$l(\mu) = \max_n \sum_{l=1}^L \mu(l) \times C[n, l], \quad (3.30)$$

$$b(\mu) = \max_m \sum_{l=1}^L \mu(l) \times R[m, l]. \quad (3.31)$$

3.5.2 Implementation and results

Algorithm 2 has been implemented on the images of *Biplane* and *Lincoln* [Figures 2.2(a) and 2.3(a)] having black object and white background. Here $l_{min} = 1$ and $l_{max} = 32$. The different minima obtained by Algorithm 2 (*IOAC* and compactness measures) for different window sizes are shown in Table 3.2.

Table 3.2: Thresholds based on Algorithm 2

Window size	Images			
	<i>Lincoln</i>		<i>Biplane</i>	
	Compactness	IOAC	Compactness	IOAC
8	10*	11* 23	5* 26	11 19 24*
10	10*	11* 23	5* 26	12 24*
12	10*	11* 23	5* 24	13 24*
14	9*	11* 23	6* 23	13 24*
16	9*	11*	6* 21	14 24*

The algorithm has also been implemented on two other images [Figures 2.4(a) and 3.5(a)] (viz., a *Blurred Chromosome* and a hand written characters *Shu*) with white object and black background. For the *Blurred Chromosome* image $l_{min} = 13$ and $l_{max} = 60$, whereas for the *Shu* image, $l_{min} = 1$ and $l_{max} =$

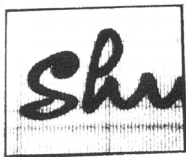
32. The different thresholds obtained by minimization of *IOAC* (compactness) measure for different window sizes are shown in Table 3.3. Some of the crisp segmented versions of the *Shu* image are shown in Figure 3.5 as a typical illustration. For other images some typical thresholded versions are given in Figures 3.2-3.4 [e.g., 3.2(c), 3.2(d), 3.3(c), 3.4(c), 3.4(e), 3.4(f)].

Table 3.3: Thresholds based on Algorithm 2

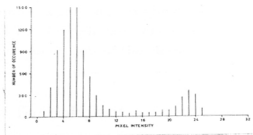
Window size	Images			
	<i>Shu</i>		<i>Blurred Chromosome</i>	
	Compactness	IOAC	Compactness	IOAC
8	8* 24	8 12*	33 56*	30* 53
10	8* 24	9 12*	33 56*	30* 51
12	8* 23	9 13*	55*	31* 49
14	9* 23	13*	54*	32* 46
16	10* 22	14*	52*	34*

The results (Tables 3.2-3.3) show that the thresholds (global) obtained by compactness measure, as expected, are very much worse for the *Shu* image than those of the *IOAC* measure. This is because of the fact that the former attempts to make a circular approximation of an object for its extraction. As a result, some of the background portions get treated as object, thus failing to remove background noise. For the *Biplane* image, neither compactness nor *IOAC* has been able to provide a global threshold appropriate for its extraction. However, it is interesting to note that the appropriate thresholds 11-14 of the *Biplane* image came out as local minima only for the *IOAC* measure.

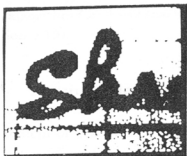
For a wide range of window sizes, the variation in global thresholds is seen to be insignificant. This corroborates the theoretical findings of Murthy and Pal [260] and the experimental results in Chapter 2 and establishes further the flexibility of the fuzzy set theoretic approach. Like Algorithm 1 the overall performance of Algorithm 2 (considering the crisp versions) is also comparable to that of the algorithms based on local information and entropy measure [64], and those described in the previous chapter.



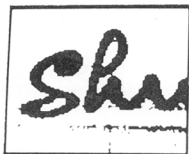
(a)



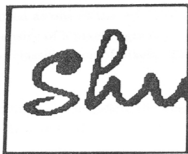
(b)



(c)



(d)



(e)

Figure 3.5: *Shu* image : a) input, b) histogram, c) threshold=8, d) threshold=10, e) threshold=14

It is to be noted here that the *IOAC* measure is dependent, to some extent, on the orientation of the object. If the object is along the horizontal and/or vertical direction, the measure will give good results. The result may be different if the object is inclined at an angle with the axes. In such cases one can rotate the axes and place them along the major and minor axes of the object.

3.6 Pixel classification for object extraction

In last two sections, algorithms for gray level thresholding have been described for object/background classification. Here we will be explaining a pixel classification algorithm for object/background separation based on adjacency measure (equation 3.10).

3.6.1 Formulation of algorithm (*Algorithm 3*)

The problem of pixel classification can be viewed as assigning the individual pixels into different types of classes depending upon the various features or properties possessed by the pixels with respect to those classes. An attempt is made here in classifying the pixels of a digital picture into two classes, namely object and background. The adjacency (or the degree of belonging) defined in equation (3.10) has been used as one of the features. In other words, the physical constraint that the intensity of a pixel is greatly influenced by its neighbors, has been exploited here for classification of pixels. The role of adjacency in this context is explained below.

Let $\mu(x, y)$ be the membership value denoting the degree of brightness of a pixel $p(x, y)$ or the degree of belonging to the object region, say. The adjacency of a

pixel to its neighbors can be defined as :

$$\begin{aligned}
 a(x, y) &= \mu(x, y) \{ \mu(x-1, y) + \mu(x+1, y) + \mu(x, y-1) + \mu(x, y+1) \} / 4 \\
 &\quad \text{(considering 4 neighbors)} \\
 &= \mu(x, y) \{ (\sum_{j=-1}^1 \sum_{i=-1}^1 \mu(x-i, y-j)) - \mu(x, y) \} / 8. \\
 &\quad \text{(considering 8 neighbors)}
 \end{aligned} \tag{3.32}$$

If the higher gray levels represent the object region, then $a(x, y)$ is high if both the pixel $p(x, y)$ and its neighbors belong to the object region, and it is low if both the pixel $p(x, y)$ and its neighbors belong to the background. $a(x, y)$ is medium if the pixel and its neighbors belong to object and background, respectively or vice versa (i.e., the region corresponding to edge or boundary of the object). So $a(x, y)$ can be regarded as the degree of belonging of the pixel $p(x, y)$ together with its neighbors to the object region.

For classification of a pixel into object and background we consider $\mu(x, y)$ and $a(x, y)$ as two features. Keeping consistency in features, a pixel pattern can be represented as

$$\vec{p}(x, y) = [a(x, y), \mu^2(x, y)]' \tag{3.33}$$

in a two dimensional feature space.

The pixel classification procedure can be thought of as a two class minimum distance classifier [45, 46] where the initial seed points or the representatives $\vec{S}(1)$ and $\vec{S}(2)$ of the two classes $C(1)$ and $C(2)$ (object and background) can be considered to be the two modes of the bivariate frequency distribution of the pattern vector $\vec{p}(x, y)$. One may also consider $[0, 0]$ and $[1, 1]$ as the initial seed points, since these are the extreme points in the feature space belonging to object (background) and background (object). The algorithm developed using this concept is given below.

Algorithm 3

Step 1 : Choose the membership function ('S' or 1-'S' as mentioned in section 3.5), number of neighbors (4 or 8) for a pixel, and assign brightness/darkness membership value $\mu(x, y)$ to the pixel $p(x, y)$, $\forall (x, y)$.

Step 2 : Compute $a(x, y)$ using equation (3.32).

Step 3 : Compute $d(i) = ||\vec{p}(x, y) - \vec{S}(i)||$, $i = 1, 2$,
the distance of an unknown pixel vector $\vec{p}(x, y)$ from the i th seed point/
representative vector of the i th class.

Step 4 : Assign the pixel $p(x, y)$ to $C(1)$ if $d(1) < d(2)$ and to $C(2)$ if $d(2) < d(1)$. Decide arbitrarily if $d(1) = d(2)$. (In such a case decision may also be taken by considering the number of neighbors of $p(x, y)$ belonging to object and background).

Step 5 : If $\vec{S}(1)$ and $\vec{S}(2)$ are chosen as the two modes, then *STOP*. If $\vec{S}(1)$ and $\vec{S}(2)$ are chosen as the two extreme points $[0, 0]$ and $[1, 1]$ then replace the seed points by the class means, and go to *Step 6*.

Step 6 : Iterate *Steps 2* to *5* until the difference between the seed points of two successive iterations is less than a pre-specified small positive quantity (ϵ).

3.6.2 Implementation and results

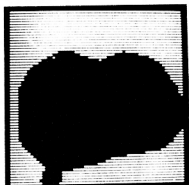
Algorithm 3 has been implemented on the images of *Biplane* [Figure 2.2(a)], *Lincoln* [Figure 2.3(a)] and *Blurred Chromosome* [Figure 2.4(a)]. The results showing satisfactory partition are depicted in Figure 3.6. Here the initial seed points were taken as the extreme points $[0, 0]$ and $[1, 1]$ and $\epsilon = 0.001$. The basic advantage of Algorithm 3 is that it is not noise sensitive because, by taking the local information from the neighbors through $a(x, y)$, it attempts to reduce the effect of noise on classification. To demonstrate this, the algorithm has also been tested by adding noise $N \sim (0, \sigma^2)$ to the image of *Biplane*. The noisy input images ($\sigma=3$ and 5) are shown in Figures 3.7(a) and 3.7(c). The corresponding partitioned images are shown in Figures 3.7(b) and 3.7(d). The gray level thresholded images obtained by Algorithm 1 for these images are also included here [Figures 3.7(e)-3.7(f)] for comparison.



(a)



(b)

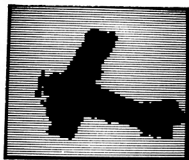


(c)

Figure 3.6: Object region extracted by pixel classification of : a) *Biplane* image
b) *Lincoln* image and c) *Blurred Chromosome* image



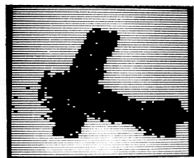
(a)



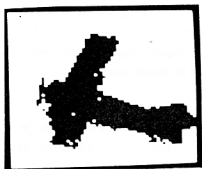
(b)



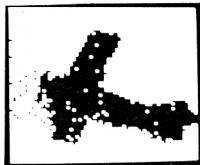
(c)



(d)



(e)



(f)

Figure 3.7: Object region extracted by pixel classification from noisy *Biplane* images : a) input; b) result by Algorithm 3 of input in (a); c) input; d) result by Algorithm 3 of input in (c); e) result by Algorithm 1 of input in (a); f) result by Algorithm 1 of input in (c)

3.7 Conclusions and discussion

An attempt has been made here to demonstrate the way of implementing the concept of fuzzy geometry in image segmentation problems. Some new measures have also been defined in this context. Various algorithms based on different geometrical measures are developed and are used to determine thresholds for object/background classification (both fuzzy and non-fuzzy).

Among the three algorithms the first two algorithms describe gray level thresholding, whereas the third one deals with a pixel classification technique. Algorithm 1 optimizes the degree of adjacency between two regions (with respect to gray levels only) to find out the thresholds. This measure takes spatial information into account and is influenced by the quadrants C_2 and C_4 (Figure 2.1, pp. 56) of the co-occurrence matrix. The algorithm is computationally fast and can be applied to any kind of image.

Algorithm 2 optimizes *IOAC* of the fuzzy object region in an image. The algorithm is based on local information and takes the shape of the object into consideration. Compactness optimization is expected to work better for circular compact objects, whereas the *IOAC* optimization gives good results for rectangular noncompact objects. The choice of window size does not seem to be very critical as long as it is restricted by the bound functions of Murthy and Pal [277]. This supports the theoretical findings in [260] and establishes the flexibility of the algorithms. Since *IOAC* does not require the calculation of co-occurrence of gray level, it takes less time compared to compactness calculation.

The third algorithm describes another way of object extraction (region growing). The algorithm uses both local and global information. The process is iterative, but converges very fast. Compared to the other two algorithms it is computationally less efficient but is more noise immune.

The crisp outputs of the proposed algorithms are found to agree well with those of the conventional techniques recently developed using the co-occurrence matrix [62, 64, 116]. The above mentioned geometrical properties have recently

been applied for extracting fuzzy medial axis of an image [266] and for motion analysis for video data abstraction [278].

Applications of major and minor axes, center of gravity and density are not considered here. Their definitions are introductory and may be further investigated.

It is to be mentioned here that the concept of fuzzy sets is more cumbersome to represent, and their properties are somewhat more expensive to compute. But as the computer memory and computing power are becoming cheaper, (and also new parallelly implementable computational technologies like artificial neural network models [132]-[165], genetic algorithms [279]-[281] are emerging with great promise) the fuzzy geometric approach to image analysis and description seems to deserve more serious considerations.

In the following chapters we will be presenting some results of investigations both theoretical and experimental that demonstrate the effectiveness of some of the concepts of artificial neural networks in formulating various parallelly implementable and robust segmentation algorithms.

Chapter 4

Kohonen type self-organizing neural network for object extraction

4.1 Introduction

An introduction to neural networks is given in Chapter 1. (Different aspects of neural networks along with the details of various models and learning rules are given in Appendix A.) Some of the applications of collective computational abilities of neural network models in image segmentation and clustering problems in order to exploit their high computational power and to make robust decisions have been described in section 1.3.3 (pp. 26).

In Chapters 4-7 we have proposed various methodologies for object extraction using modified versions of Kohonen's model, Hopfield's model and a multi-layer neural network where the problem of object extraction from a scene is viewed as a self-organizing task, or an optimization problem, or an unsupervised classification.

The work described in this chapter is an attempt to use Kohonen's concept [136, 282] of self-organizing feature map for object extraction from an image. Here, the object extraction problem is viewed as a self-organizing task and is solved by employing a modified version of Kohonen's model. The term *self-organization* refers to the ability to learn the structure of the input data even when there is no available information about it. Self-organization is a basic principle of sensory paths of the human visual system. In the feature mapping algorithm of Kohonen the same input is given to all the neurons to perform self-supervised classification. The present algorithm, on the other hand, considers the number of neurons equal to the number of pixels in an input image and the input differs from neuron to neuron.

The proposed model uses both global and local information as input features. Statistical criteria such as least square error and correlation maximization are used for obtaining the optimal output. Proof for the convergence of the algorithm is also given. The technique is found to work well even for noisy input.

It is to be mentioned here that the work of Chua and Yang [150, 151] also considered different input for different neurons, but the working principle (weight adjustment/learning) and the transfer functions are different. The principle

used in the present algorithm is more general as compared to that in [150, 151], because they assumed the output state of the neurons as -1 or +1, whereas the present case the output may be continuous.

4.2 Kohonen's model of self-organizing feature map

Kohonen's self-organizing network consists of a single layer of neurons as shown in Figure 4.1. The neurons are highly interconnected within the layer as well as with the outside world.

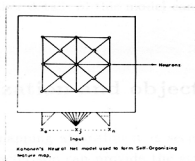


Figure 4.1: Kohonen's model of neural network

Let the signal

$$\vec{U} = \{u_1, u_2, \dots, u_n\}' \in R^n \quad (4.1)$$

be connected as input to all the processing units. Let the length of the input vector be fixed (to some constant length) and the processing unit (neuron) i have the weights (chosen as random numbers)

$$\vec{W}_i = \{w_{i1}, w_{i2}, \dots, w_{in}\} \quad (4.2)$$

where, w_{ij} is the weight of the i th unit from j th component of the input. The i th unit then forms the discriminant function

$$\eta_i = \sum_{j=1}^n w_{ij} u_j = \vec{W}_i \cdot \vec{U} \quad (4.3)$$

where, η_i is the output of that unit. A discrimination mechanism will then detect the maximum of η_i . Let

$$\eta_k = \max_i \{\eta_i\}. \quad (4.4)$$

For the k th unit and all of its neighbors (within a specified radius of r , say) the weight updating rule

$$\vec{W}_k(t+1) = \frac{\vec{W}_k(t) + \alpha \cdot \vec{U}(t)}{\|\vec{W}_k(t) + \alpha \cdot \vec{U}(t)\|} \quad (4.5)$$

is applied. Here the variables have been labeled by a discrete time index t . α ($0 < \alpha < 1$) is a gain parameter that decreases with time and the denominator is the length of the weight vector (which is used only for normalization of the weight vector). Further details of this model can be found in Appendix A.

4.3 Self-organization and object extraction

From Kohonen's feature mapping algorithm it is seen that self-organization and feature mapping characteristic can provide the structural information of the input data set, without having any apriori knowledge of it. In the context of pattern recognition and image processing, this can be viewed as clustering and/or segmenting an image space into meaningful regions, say object and background. In the following subsection, we shall explain how the concept of self-organization can be exploited for object extraction from a gray tone image.

4.3.1 Image processing in the notion of neural networks

An L level image $X(M \times N)$ can be represented by a neural network $Y(M \times N)$, having $M \times N$ neurons arranged in M rows and N columns (each neuron corresponds to a pixel). Let the neuron in the i th row and j th column be represented by $y(i, j)$ and let it take input from the global/local information of the (i, j) th pixel $x(i, j)$ having gray level intensity x .

In feature mapping algorithm, the maximum length of the input and weight vectors is fixed. Let the maximum length for each component of the input vector be unity. To keep the value of each component of the input ≤ 1 , let us apply a mapping

$$f : [l_{min}, l_{max}] \longrightarrow [0, 1].$$

Here l_{max} and l_{min} are the highest and the lowest gray levels available in the image. The components of the weight vectors are chosen as random numbers in $[0, 1]$.

The output (activation level) of a neuron $y(i, j)$ is a function of both the input and the weight vector. The function may be linear or non-linear depending on the problem at hand. The output status of neuron $y(i, j)$ governs the processed output. For example, consider the case of extracting edges (or regions i.e., segmentation). Here, the output value of a neuron determines to what extent the pixel $x(i, j)$ is a member of the *edge pixel* subset (or a region). Similarly, for image enhancement, it reflects the enhanced value of pixel intensity. (In this context we mention that the output status values of the neurons may be considered to form a fuzzy set.)

4.3.2 Formulation of the object extraction algorithm

Let us consider the case of object background classification in an image. Here the input to a neuron $y(i, j)$ is considered to be an N -dimensional vector

$$\vec{U}(i, j) = \{u_1(i, j), u_2(i, j), \dots, u_n(i, j)\}'.$$

The weight (to a neuron) is also a vector quantity of the same dimension

$$\vec{W}(i, j) = \{w_1(i, j), w_2(i, j), \dots, w_n(i, j)\}.$$

Then compute the dot product as

$$d(i, j)_t = \vec{W}(i, j)_t \cdot \vec{U}(i, j)_t = \sum_{k=1}^n w_k(i, j) \cdot u_k(i, j) \quad (4.6)$$

where the subscript 't' is used to denote the time instant.

Only those neurons for which $d(i, j)_t > T$ (T is assumed to be a threshold) are allowed to modify their weights along with their neighbors (within a specified radius). Consideration of a set of neighbors enables one to grow the region by including those elements which might have been dropped out because of the randomness of weights or presence of noise. The weight updating procedure is the same as in equation (4.5). The value of α (equation 4.5) decreases with time. In the present case, it is chosen to be inversely related to the iteration number.

The output (just to check convergence) of the network at time instant ' t ' is calculated as

$$O_t = \sum_{i,j} d(i, j)_t \quad (4.7)$$

with

$$d(i, j)_t > T.$$

The procedure of updation of the weights is continued until

$$|O_{t+1} - O_t| < \delta \quad (4.8)$$

where δ is a pre-assigned very small positive quantity. At this stage the network is said to have converged (settled). After the network has converged, the pixels $x(i, j)$ (along with their neighbors) with the constraint $d(i, j) > T$, may be considered to constitute a region, say, object.

It may be mentioned here that the value of $d(i, j)$ denotes the degree of belonging of the pixel $x(i, j)$ to the object (background). If $d(i, j) = 1$, the pixel fully belongs to the object. Thus the extracted region may be considered to denote a fuzzy subset *object*. About the stability of the network the following proposition can be made.

Proposition : The network converges for all values of T .

Proof :

To prove

$$|O_{t+1} - O_t| < \delta$$

where

$$O_t = \sum_{i,j} d(i,j)_t$$

with

$$d(i,j)_t > T,$$

T is a threshold, and δ is a pre-assigned small positive quantity.

Here $d(i,j)_t = \vec{W}(i,j)_t \cdot \vec{U}(i,j)_t$. From equation (4.5) we have

$$\vec{W}(i,j)_{t+1} = \frac{\vec{W}(i,j)_t + \alpha \vec{U}(i,j)_t}{\|\vec{W}(i,j)_t + \alpha \vec{U}(i,j)_t\|}. \quad (4.9)$$

From above it is evident that $\|\vec{W}(i,j)_{t+1}\| = 1, \forall t$. It is also given that α decreases as t increases and for the present case $\vec{U}(i,j)$ is constant with respect to time. So, as $t \rightarrow \infty, \alpha \rightarrow 0$. Hence

$$\lim_{t \rightarrow \infty} \vec{W}(i,j)_{t+1} = \frac{\vec{W}(i,j)_t}{\|\vec{W}(i,j)_t\|} = \vec{W}(i,j)_t, \text{ as } \|\vec{W}(i,j)_t\| = 1.$$

So, with time the updating of the weights converges. Since $\vec{U}$ is not a function of time

$$d(i,j)_t = \vec{W}(i,j)_t \cdot \vec{U}(i,j)$$

also converges as $t \rightarrow \infty$.

Hence

$$O_t = \sum_{i,j} d(i,j)_t$$

will become constant as $t \rightarrow \infty$. Thus

$$|O_{t+1} - O_t| < \delta.$$

Since this is independent of T , the network converges for all T .

Hence the proof.



4.3.3 Selection of optimal threshold

In the above subsection, a threshold T in the $\vec{W} \cdot \vec{U}$ plane was assumed to decide whether to update (or not to do so) the weights of a neuron; i.e., the

threshold T divides the neuron set into two groups (thereby the image space into two regions, say object and background). The weights of only one group are updated. For a particular threshold the updating procedure continues until it converges. The threshold is then varied. Any one of the following two statistical criteria can be used for selecting the optimal threshold.

Least square error (LSE) method

As the network converges, the image space is divided into two groups, one group having $d(= \vec{W} \cdot \vec{U}) > T$ (the object pixels), and the other with $d < T$ (the background pixels). Let $\vec{m}_o$ and $\vec{m}_b$ be the mean values of the activation levels (input values) of the object and background regions, respectively. Since we are concerned with object/background situation only, we can assume that $\vec{m}_o$ and $\vec{m}_b$ are the ideal (constant) activation levels of the object and background pixels, respectively. Let N_o be the number of pixels in the object region, N_b that in the background region and V_i be the activation level of the i th element in the stable state. Then the total square error is

$$\epsilon(T) = \sum_{i=1}^{N_o} (\vec{V}_i - \vec{m}_o)^2 + \sum_{j=1}^{N_b} (\vec{V}_j - \vec{m}_b)^2. \quad (4.10)$$

The threshold (T) for which this error is minimum can be taken as the optimal threshold. This is optimal in the sense that the deviation of the segmented image from ideal scene is minimum.

Correlation maximization method

In this case, the statistical correlation coefficient between the input sequence (image) and the output sequence is maximized to get the optimal output. As the threshold is varied, the output sequence is changed thereby altering the value of the correlation coefficient. When the threshold corresponds to the actual boundary, the initial grouping is close to the expected output and thus the correlation coefficient is maximum.

Let x_i and y_i , $i = 1, 2, \dots, n$ be two sequences with mean values $\bar{x}$ and $\bar{y}$, respectively. Then the correlation co-efficient (r_{xy}) between these two sequences

is defined as

$$r_{xy} = \frac{\text{Cov}(x, y)}{\sigma_x \sigma_y} \quad (4.11)$$

where

$$\text{Cov}(x, y) = \frac{1}{n} \sum_i (x_i - \bar{x})(y_i - \bar{y})$$

and

$$\sigma_x^2 = \frac{1}{n} \sum_i (x_i - \bar{x})^2, \quad \sigma_y^2 = \frac{1}{n} \sum_i (y_i - \bar{y})^2.$$

In the present case, one sequence is the input values of the neurons. After the network has converged for a particular value of T , the second sequence contains values either m_o (for all elements belonging to objects) or m_b (for all elements belonging to the background), where m_o and m_b have the same meaning as in the previous case. Let N_o be the number of pixels belonging to the object region and N_b be that to the background. It can easily be seen that the mean values of the above two sequences are the same. Let the mean of the sequences be m . Then the variance of the input and the output sequences will be

$$\sigma_{inp}^2 = \frac{1}{n} \sum_i (U_i - m)^2, \quad (4.12)$$

$$\begin{aligned} \sigma_{out}^2(T) &= \frac{1}{n} \left\{ \sum_{i=1}^{N_o} (m_o - m)^2 + \sum_{j=1}^{N_b} (m_b - m)^2 \right\} \\ &= \frac{1}{n} \{ N_o (m_o - m)^2 + N_b (m_b - m)^2 \}, \end{aligned} \quad (4.13)$$

$$\text{Cov}(inp, out)_T = \frac{1}{n} \left\{ \sum_{i=1}^{N_o} (m_o - m)(U_i - m) + \sum_{j=1}^{N_b} (m_b - m)(U_j - m) \right\} \quad (4.14)$$

and

$$r(T) = \frac{\text{Cov}(inp, out)_T}{\sigma_{inp} \sigma_{out}(T)}. \quad (4.15)$$

4.3.4 Features of the modified model

Kohonen demonstrated a self-organizing feature map criterion [136, 282] for developing a self-supervised classification algorithm. The same input is given to all the neurons while learning the parameters. The present work on unsupervised classification (i.e., clustering or segmentation), on the other hand,

considers the number of neurons the same as that of the pixels and the input is different for different neurons.

The mathematical proof for the convergence of the algorithms is given in section 4.3.2. It can further be established logically as follows. The weight updating procedure is the same as that of Kohonen (equation 4.5). In Kohonen's model the inputs to all the neurons are the same, whereas in the present case the inputs to different neurons are different. The weight updating procedure is such that it always brings the randomly assigned weights towards the input. So, even if the inputs to the neurons are not the same, the weight updating procedure will converge thereby settling the network.

In the present case of object extraction, it is assumed that the object regions have more or less uniform intensities, thereby forming spatial clusters to some extent. So, it can be said that if a pixel belongs to an object, then the probability of its neighbors to belong to the same region is very high. Thus at the onset, the input values to the neurons in a particular region are more or less similar, but the weights may be different, as they are assigned randomly. The formation of the clusters can then be described as follows.

If the weights assigned are such that

$$\vec{W}_i \cdot \vec{U}_i > T, \forall i$$

in a particular region, then in course of time the weight updating will converge and a spatial cluster will be formed in that region.

On the other hand, if the assigned weights are such that

$$\vec{W}_i \cdot \vec{U}_i > T$$

for a few i 's in a particular region, then during updating of the weights of those neurons they will update the weights of their neighbors also and thereby bring them too into the cluster.

But a cluster will not be formed if

$$\vec{W}_i \cdot \vec{U}_i < T, \forall i$$

in a region. Since the weights are assigned randomly, the probability of occurrence of such a situation in the practical case is negligible. So it is clear that the clusters will be formed and extracted in the proposed technique.

Furthermore, instead of minimizing the difference between the input and weight vectors, their dot product is maximized here. Since the maximum length of the input and weight vectors is fixed, the two criteria are basically the same.

The above mentioned theory developed has then been used to formulate two algorithms considering different versions of input vector. These are explained below.

Case 1 :

Let the input vector have only one component and consequently the weight vector will have a single component. Let

$$U(i, j) = \frac{x(i, j) - l_{min}}{l_{max} - l_{min}} \quad (4.16)$$

where $x(i, j)$ is the gray level of the (i, j) th pixel, and l_{max} and l_{min} are the maximum and minimum available gray levels of the image. Note that the value of $U(i, j)$ lies in the range $[0, 1]$. The weights will be assigned as random numbers in $[0, 1]$. The method of object extraction may be written in algorithmic form as follows.

Algorithm 1

Step 1 : Assign input $U(i, j)$ to the neuron $y(i, j)$ using equation (4.16), $\forall(i, j)$.

Step 2 : Initialize the weights of the neurons by random numbers in $[0, 1]$.

Step 3 : Compute the products (for all neurons) as

$$d(i, j) = W(i, j).U(i, j).$$

Step 4 : Select those neurons for which $d(i, j) > T$ (T is a pre-assigned threshold, $0 < T < 1$) and update (along with the neighbors) the weights using equation (4.9).

Step 5 : Repeat *Steps 3 to 4* until convergence is obtained with respect to equation (4.8).

Step 6 : Vary T , iterate *Steps 2 to 5* and find the optimum one (say, T^*).

The neurons for which $d(i, j) > T^*$ constitute the object region.

Case 2 : (Algorithm 2)

In Algorithm 1 the input to a neuron is a scalar quantity, the representative of the gray level information of the corresponding pixel. No local information is incorporated. In the present algorithm, the input to each neuron is considered to be a vector quantity having two components, namely, the gray value of the pixel and the average gray value of its neighbors, each of them mapped onto $[0, 1]$. The computational steps of this algorithm are the same as those of Algorithm 1, except that the input has the aforesaid two components.

The self organizing ability of the network makes it possible to organize the object pixels (neurons having $d = \vec{W} \cdot \vec{U} > T$) together, thereby separating the objects from the background. The local feedback makes the procedure noise insensitive and robust.

4.4 Simulation and results

For the present problem, the neighborhood of a neuron is considered over a 3×3 window. The value of α (equation 4.9) is chosen as

$$\alpha_N = \frac{1}{2N},$$

i.e., α at the N th iteration is taken as $\frac{1}{2N}$. This ensures $0 < \alpha < 1$ and decrease of α with time. The value of δ is taken to be 0.001.

The proposed algorithms are implemented and tested on three different images. The images used are that of a *Biplane*, [Figure 2.2(a)], *Lincoln* [Figure 2.3(a)] and a *Noisy tank* [Figure 4.4(a)]. The size of the images is 64×64 .

The results obtained by Algorithms 1 and 2 for the *Biplane* [Figure 2.2(a)]

see Figure 2.2(a)

(a)



(b)



(c)

Figure 4.2: *Biplane* image : a) input, b) extracted object by Algorithm 1; c) extracted object by Algorithm 2

see Figure 2.3(a)

(a)

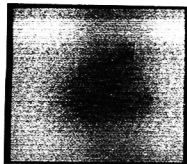


(b)

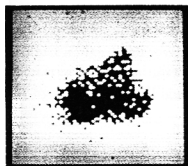


(c)

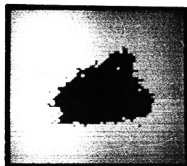
Figure 4.3: *Lincoln* image : a) input, b) extracted object by Algorithm 1, c) extracted object by Algorithm 2



(a)



(b)



(c)

Figure 4.4: *Noisy tank* image : a) input, b) extracted object by Algorithm 1, c) extracted object by Algorithm 2

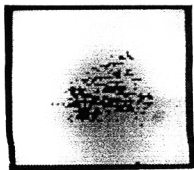


Figure 4.5: Segmented *Noisy tank* image

image are depicted in Figures 4.2(b) and 4.2(c), respectively. The propeller of the plane could not be perfectly extracted by Algorithm 2, which may be due to the incorporation of local information and thereby causing blurring effect. The optimal thresholds for the Algorithms 1 and 2 are 0.25 and 0.50, respectively for this image.

Figures 4.3(b) and 4.3(c) depict the extracted objects for *Lincoln* image [Figure 2.3(a)] corresponding to Algorithms 1 and 2. The results show that the output obtained by Algorithm 2 preserves continuity much more than that obtained by Algorithm 1. The optimal thresholds are 0.281 and 0.531 for Algorithms 1 and 2, respectively.

For the *Noisy tank* image the optimal outputs are obtained for the thresholds 0.156 and 0.25 by Algorithms 1 and 2, respectively. The corresponding results are shown in Figures 4.4(b) and 4.4(c). From the results it is noticed that the object extracted by Algorithm 2 [Figure 4.4(c)], as expected, is more compact than that by Algorithm 1 [Figure 4.4(b)]. This is because Algorithm 2 takes local information into account which Algorithm 1 does not.

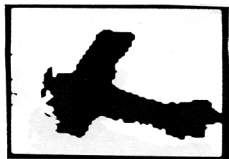
A very recent gray level thresholding algorithm proposed by Pal and Rosenfeld [116] dealt with the image of the *Noisy tank*. The algorithm is based on fuzzy compactness [102] minimization principle as described in Chapter 3. The algorithm takes the geometry of the object into consideration for its extraction. One of the good thresholds obtained by their algorithm was 31. The corresponding segmented version is given in Figure 4.5 (for comparison). From the segmented image [Figure 4.5] it is very difficult to identify the object as a tank. The results obtained by the proposed algorithms are comparatively better than those of Pal and Rosenfeld [116]. Besides this, the problem of choosing parameters does not arise in the present case.

To demonstrate the effectiveness of noise immunity of the proposed algorithms (in addition to their demonstration shown on a real *Noisy tank* image) they have been implemented on two noisy versions [Figures 3.7(a) and 3.7(c)] of the *Biplane* image [Figure 2.2(a)]. The noisy versions are obtained by adding $N(0, \sigma^2)$ noise with $\sigma=3$ and 5. The corresponding extracted objects by Algo-

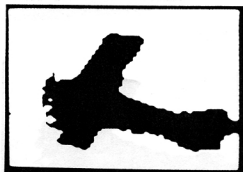
rithms 1 and 2 are given in Figures 4.6(b) and 4.7(b), and 4.6(c) and 4.6(c), respectively. From the results it is evident that the outputs are, to a great extent, noise insensitive. The results are seen to be comparable to those [Figures 3.7(b) and 3.7(d)] of the pixel classification algorithm described in the previous chapter.

see Figure 3.7(a)

(a)



(b)



(c)

Figure 4.6: A noisy version ($\sigma = 3$) of the *Biplane* image : a) input, b) extracted object by Algorithm 1, c) extracted object by Algorithm 2

From the results and the previous discussion it is evident that the outputs of Algorithm 2 are more compact and more noise immune compared to that of Algorithm 1. Thus, if the noise level of the input is less, one can prefer Algorithm 1, but for more noisy images Algorithm 2 is more suitable.

As a typical illustration, two curves showing the variation of the two objective criteria (equations 4.10 and 4.15, respectively) with change of threshold (for the *Noisy tank* image and Algorithm 1) are given in Figures 4.8 and 4.9. It may be mentioned here that for the images used in the present case, incidentally the

see Figure 3.7(c)

(a)



(b)



(c)

Figure 4.7: A noisy version ($\sigma = 5$) of the *Biplane* image : a) input, b) extracted object by Algorithm 1, c) extracted object by Algorithm 2

optimal thresholds obtained for both the objective criteria are the same.

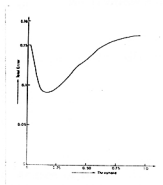


Figure 4.8: Variation of total square error with threshold

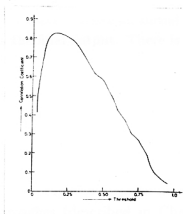


Figure 4.9: Variation of correlation co-efficient with threshold

4.5 Conclusions and discussion

The concept of self-organizing feature map proposed by Kohonen has been exploited here to extract objects from a scene. An unsupervised classification (segmentation) algorithm has been formulated by adopting some modifications in the structure of the network. Here one neuron is assigned for each pixel. Instead of giving the same input to all the neurons, input given to different

neurons are different. The convergence of the algorithms has been proved mathematically. Statistical criteria for choosing the optimal output are also suggested. The results obtained by the proposed algorithms are seen to be quite satisfactory.

It may be noticed that these algorithms basically are of relaxation type. The differences between the proposed technique and probabilistic relaxation labeling are described below. Some results of probabilistic relaxation are given in the next chapter.

In relaxation labeling, one compatibility function is defined depending on the apriori information about the degree of belonging of the elements into classes. A prior assignment of the elements into different classes is also necessary. The number of probability values (belonging to different classes) to be calculated is the same as the number of classes. Given an initial stage, the corresponding converged stage is taken as the final output. There is no provision for choosing the optimal output.

In the proposed algorithms there is no necessity of calculating the compatibility function. As far as class assignment is concerned, only one value representing the degree of belonging is sufficient. An optimal output selection criterion is specified in the present technique.

It may be mentioned that the proposed algorithms stabilize faster than other neural network based approaches (described in Chapters 5-7 using Hopfield type and multi-layer networks) which are found to be more suitable for noisy images. This is, possibly, due to the nature of the weight updating principle. In the present technique the weights associated with a neuron do not get affected by the status of the neighboring neurons, rather these are affected by the status of the concerned neuron.

Chapter 5

Hopfield type neural network
for object background
classification

5.1 Introduction

An application of a self-organizing artificial neural network has been described in the previous chapter where the object extraction problem was viewed as a self-organizing task. In this chapter we are going to describe a different network architecture (Hopfield type) which is used as an associative memory [136, 146] to extract objects from a scene. The present architecture is more suitable for noisy images compared to that described in Chapter 4. Moreover, unlike the methods described in Chapter 4, some guidelines for initialization of network parameters for different types of images (i.e., noisy and noise free) have also been provided. It has been established that the stable states of the network (local minima of its energy function) correspond to the optimal partitioning of the scene into compact (meaningful) regions. So given an initial state, the status of the neurons is modified iteratively so as to attain a stable state, when the neurons with ON status constitute the object (background) regions.

The architecture and dynamics of the network used here are such that self-organization of the network is enforced. The proposed technique exploits the inherent structure of the input data to extract objects from a scene. The network architecture used is as follows. A neuron is assigned corresponding to every pixel. Each neuron is connected to all of its neighbors. The network architecture can, therefore, be thought of as a modified version (topologically) of Hopfield's model [137]-[139] where the connection strengths of a neuron to other neurons outside its neighborhood are zero. Depending upon the gray value of the pixel, the initial status of the corresponding neuron is fixed. (Here we have used simply a linear transformed version of the gray value.) The connection strength (weight) between a pair of neurons is assigned depending on the apriori information about the joint distribution of the corresponding gray level pair. The neurons operate in parallel and update their status (while in transition) taking the consensus of their neighbors.

The convergence of the network is checked with respect to every neuron. If the change in output status of every neuron is less than some pre-assigned small positive quantity (ϵ) the network is said to have converged. The input/output

transfer function of a neuron can be any continuous monotonically increasing function of the instantaneous input to it. The exact nature of the function may depend upon the problem at hand. In the present work, the standard 'S' type function (equation 2.11, pp. 46) as used in fuzzy set theory is considered.

Since the weight assignment technique and the status updating procedure consider the effect of the neighborhood, the object extraction procedure will not be sensitive to noise and will try to preserve connectivity among the pixels of the extracted regions. It is to be noted that the initial activation levels may affect the ultimate output to some extent.

The effectiveness of the present technique (including the effect of initialization of weights) has been demonstrated on a synthetic binary image and two gray tone real images corrupted by noise. Results have been compared with those of the probabilistic relaxation technique [2, 68] considering the percentage of correct classification of pixels as an objective measure and visual inspection.

5.2 Hopfield's model

Hopfield's model of neural network consists of completely connected network (schematic representation is given in Appendix A, Figure A.3) of neurons. It acts as an associative memory. The model can have both discrete and continuous dynamics which are described below.

Case 1 : Discrete model

In this model two state neurons are used. Each neuron i , has two possible states, characterized by the output V_i of the neuron having values -1 or $+1$. The input to a neuron comes from two sources, external input bias I_i and inputs from other neurons (to which it is connected). The total input to the i th neuron is

$$U_i = \sum_j W_{ij} V_j + I_i \quad (5.1)$$

where W_{ij} is the synaptic interconnection strength (weight) from neuron i to j .

The status updating takes place according to the threshold rule (with threshold θ_i)

$$\begin{aligned} V_i &\longrightarrow -1 \quad \text{if } U_i \leq \theta_i \\ &\longrightarrow 1 \quad \text{if } U_i > \theta_i. \end{aligned} \quad (5.2)$$

The objective function (called the energy of the network) representing the overall status of the network is taken as

$$E = - \sum_i \sum_j W_{ij} V_i V_j - \sum_i I_i V_i + \sum_i \theta_i V_i. \quad (5.3)$$

For a particular set up (fixed I_i, W_{ij}, θ_i), the energy E is a function of the status (V_i) of the nodes. The local minima of this function correspond to the stable states of the network.

The change ΔE in energy E for a change in status $\Delta V_i, \forall i$ is

$$\Delta E = - \left[\sum_j W_{ij} V_j + I_i - \theta_i \right] \Delta V_i. \quad (5.4)$$

Case 2 : Continuous model

This model is based on continuous variables and responses but retains all the significant behaviors of the discrete model. Let the output variable V_i for the i th neuron lie in the range $[-1, +1]$, and be continuous and monotonically increasing non linear function of the instantaneous input U_i to it. The typical input/output relation

$$V_i = g(U_i) \quad (5.5)$$

is usually sigmoidal with asymptotes -1 and $+1$. It is also necessary that the inverse of g exists i.e., $g^{-1}(V_i)$ is defined.

A typical choice of the function g is [see Figure A.4]

$$g(x) = 2 \frac{1}{1 + e^{-(x-\theta)/\theta_0}} - 1. \quad (5.6)$$

The value of $g(x)$ lies in $[-1, 1]$ with 0.0 at $x = \theta$.

The energy function of this model is

$$E = - \sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \sum_i \frac{1}{R_i} \int_0^{V_i} g^{-1}(V) dV. \quad (5.7)$$

5.3 Description of the proposed model

In this section we will describe a modified version of Hopfield's network model used for the present investigation. Before that let us first define a neighborhood system for an image X .

Definition : The d th order neighborhood (N_{ij}^d) of any element (i, j) on X is described as

$$N_{ij}^d = \{(i, j) \in X\}$$

such that

- $(i, j) \notin N_{ij}^d$,
- if $(k, l) \in N_{ij}^d$ then $(i, j) \in N_{kl}^d$,

and the neighborhood system $N^d = \{N_{ij}^d\}$. Different ordered neighborhood systems can be defined considering different sets of neighboring pixels of (i, j) . For example, $N^1 = \{N_{ij}^1\}$ can be obtained by taking the four nearest neighbor pixels. Similarly, $N^2 = \{N_{ij}^2\}$ consists of the eight pixels neighboring (i, j) , and so on (as shown in Figure 5.1). Due to the finite size of the image, the neighborhood of the pixels on the boundaries are necessarily smaller unless a periodic structure is assumed.

In the present investigation one neuron is assigned to each pixel and each neuron is connected only to its neighboring neurons. In other words, the (i, j) th neuron is connected to all neurons in its neighborhood N_{ij}^d (d th order neighborhood). The selection of the neighborhood system (N^d) is problem dependent. With a second order (N^2) neighborhood, the network topology takes the form depicted in Figure 5.2. Thus we see that $W_{ij,kl} > 0$ if $(k, l) \in N_{ij}^d$, otherwise $W_{ij,kl} = 0$, where $W_{ij,kl}$ is the connection strength between (i, j) th and the

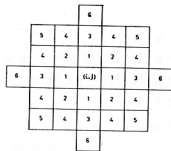


Figure 5.1: Neighborhood system

(k, l) th neurons. Obviously, if the size of the neighborhood increases the computation involved will also increase. The network can be viewed as a modified version (topologically) of Hopfield's model [138, 139] with the modification that the connection strength of a neuron to all neurons outside the neighborhood is zero. In the following discussion a N^2 neighborhood will be assumed.

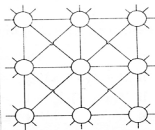


Figure 5.2: Topology of the neural network

From the input-output transfer function (equation 5.6) we notice that, $\lim_{x \rightarrow -\infty} g(x) = -1$ and $\lim_{x \rightarrow \infty} g(x) = 1$. In other words, as $x \rightarrow \pm\infty$, the function asymptotically approaches the limiting values of ± 1 . In the present case since the number of neighbors is fixed (8), for normalized values of the connection strengths the input value to a neuron lies in $[-8, 8]$, as the output value of an individual neuron is in $[-1, 1]$. So, a polynomial function defined over a finite domain can be used as an input/output transfer function. One

such choice can be

$$\begin{aligned}
 g(x) &= 0 && \text{if } x \leq a \\
 &= \frac{1}{2} \times 2^n \times \left(\frac{x-a}{c-a}\right)^n && \text{if } a \leq x \leq b \\
 &= 1 - \frac{1}{2} \times 2^n \times \left(\frac{x-c}{c-a}\right)^n && \text{if } b \leq x \leq c \\
 &= 1 && \text{if } c \leq x
 \end{aligned} \tag{5.8}$$

for $n \geq 2$ and $b = (a + c)/2$. In this case $g(x)$ lies in $[0, 1]$ with $g(x) = 0.5$ at $x = b$. The domain of x is $[a, c]$. The value of n controls the sharpness (steepness) of the function. The above function is nothing but a generalized version of the standard 'S' function as defined in equation (2.11) (pp. 46).

In the present case, the domain of x is $[-8, 8]$, i.e., $a = -8$, $c = 8$. The function g then takes the form

$$\begin{aligned}
 g(x) &= 0 && \text{if } x \leq -8 \\
 &= \frac{1}{2^{3n+1}}(x+8)^n && \text{if } -8 \leq x \leq 0 \\
 &= 1 - \frac{1}{2^{3n+1}}(8-x)^n && \text{if } 0 \leq x \leq 8 \\
 &= 1 && \text{if } 8 \leq x.
 \end{aligned} \tag{5.9}$$

To make the function lie in $[-1, +1]$ the expression can further be modified as

$$\begin{aligned}
 g(x) &= -1 && \text{if } x \leq -8 \\
 &= \frac{1}{2^{3n}}(x+8)^n - 1 && \text{if } -8 \leq x \leq 0 \\
 &= 1 - \frac{1}{2^{3n}}(8-x)^n && \text{if } 0 \leq x \leq 8 \\
 &= 1 && \text{if } 8 \leq x.
 \end{aligned} \tag{5.10}$$

Schematic representation of such a function is given in Figure 5.3.

The status updation principles of the network are identical to that of Hopfield's model and therefore its convergence is assured.

5.4 Object extraction algorithms

In this section we will describe a method of formulating an appropriate energy function, assigning weights and their realization with a network for object-

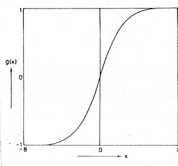


Figure 5.3: Input output transfer function

background classification.

5.4.1 Formulation of an energy function

Let us now formulate the energy function of a neural network in such a manner that in the stable state of the network the object regions are clearly separated from the background. Consider the energy function defined in equation (5.3) which has two parts. The first part is due to the local field or feedback and the second part corresponds to the input bias of the neuron. In terms of images the first part can be viewed as the impact of the gray levels of the neighboring pixels whereas, the second part can be attributed to the gray value of the pixel under consideration. The energy function can then be formulated in the following way.

Whatever be the input, without loss of generality, it can be assumed that the input values to the neurons lie in $[-1, 1]$. In an image the gray value of a pixel is highly influenced by that of its neighbors. So, if a pixel belongs to the object region, the probability of its neighbors to belong to the same object region is very high. This suggests that if a pair of adjacent pixels is having similar values then the amount of energy contributed by this pair to the total energy should be much less. If the gray values of two adjacent pixels are V_i and V_j (from now onwards the two dimensional indices of a neuron will be replaced by a one dimensional index), then one possible choice for the contribution of each

of these pairs to the total energy can be

$$-W_{ij}V_iV_j$$

where W_{ij} is a non-negative constant for a particular i and j . W_{ij} can be viewed as the connection strength between the i th and the j th neurons, and V_i is the output status of the i th neuron. So the total energy contributed by all pixel pairs will be

$$E = - \sum_{i,j} W_{ij}V_iV_j = - \sum_i (\sum_j W_{ij}V_j)V_i = - \sum_i h_iV_i \quad (5.11)$$

where $h_i = \sum_j W_{ij}V_j$ is termed as the local field. This constitutes the first part of the energy function.

For every neuron i , there is an initial input bias I_i which can be taken to be proportional to the gray level of the corresponding pixel. This is quite logical (even in a noisy environment) in the sense that if the gray level of a pixel is high (low), the corresponding intensity value of the scene is expected to be high (low). The input bias value is selected to be in the range $[-1, 1]$. Under the above framework an ON (1) neuron corresponds to an object pixel whereas, an OFF (-1) neuron represents a background pixel. So the threshold between object and background can logically be taken as 0. If any neuron is having a very high positive bias ($\simeq 1$) or very high negative bias ($\simeq -1$), then it is very much likely that in the stable state it will be ON or OFF respectively. So, the product I_iV_i should contribute less toward the total energy value. Thus the second part of the expression of energy may be taken as

$$- \sum_i I_iV_i. \quad (5.12)$$

Hence, the expression of energy for the object extraction problem takes the form

$$E = - \sum_{i,j} W_{ij}V_iV_j - \sum_i I_iV_i. \quad (5.13)$$

Thus we see that expression in equation (5.13) is to be minimized for separating the object regions from the background which requires the knowledge of W_{ij} and we address this issue in the next subsection.

5.4.2 Assignment of weights

We have mentioned before that if a particular pixel belongs to the object region then its neighboring pixels are also expected to belong to the same region. In other words, the probability of joint occurrence of that gray value pair is expected to be high. This suggests the use of probability of co-occurrence of the gray values as the connection strength. However, if the image is highly corrupted by noise then the joint probability measure may not convey the desired information, and hence may not be appropriate to be used as the connection strength. In such a case all the connection strengths are assigned as unity.

Let us now describe the connection strength estimation procedure based on the probability of co-occurrence of pixels. Let N_{kl} be the number of occurrences of the gray level k followed by l in a particular fashion in the entire image. Then the weight (W_{ij}) between the i th and the j th neurons can be taken as

$$W_{ij} = \frac{N_{kl}}{\sum_{m,n} N_{mn}} = p_{kl} \quad (5.14)$$

where k is the input gray level of the i th neuron and l that of the j th neuron, and p_{kl} is the probability of joint occurrence of the gray level k followed by l .

Note that $\sum_{i,j} W_{ij} = \sum_{k,l} p_{kl} = 1$. Now, instead of taking $W_{ij} = p_{ij}$ one can also choose

$$\begin{aligned} W_{ij} &= \frac{N_{kl}}{\max_{m,n} \{N_{mn}\}} \\ &= \frac{N_{kl}}{\sum_{m,n} N_{mn}} \times \frac{\sum_{m,n} N_{mn}}{\max_{m,n} \{N_{mn}\}} \\ &= C \frac{N_{kl}}{\sum_{m,n} N_{mn}} \quad \left[C = \frac{\sum_{m,n} N_{mn}}{\max_{m,n} \{N_{mn}\}} \right] \end{aligned} \quad (5.15)$$

with C as a positive constant for a given image.

The two criteria (equations 5.14 and 5.15) are basically the same, the only difference is in scaling. But the later one is preferred, since it strongly favors the most frequent gray value pairs which is intuitively desirable.

5.4.3 Realization with a neural network

Discrete model :

Equation (5.13) is equivalent to equation (5.3) with $\theta = 0$. Hence the network structure described in section 5.3 can be used to obtain the minima of the expression of energy in equation (5.13). In other words, the network [Figure 5.2] can solve the image segmentation (object extraction) problem. Here the status updating rule is the same as in equation (5.2) with $\theta = 0$.

Continuous model :

The expressions of energy in equation (5.7) and equation (5.13) differ by the term

$$\sum_i \frac{1}{R_i} \int_0^{V_i} g^{-1}(V) dV$$

only. For high gain [138] of 'g' (equation 5.6) the above term vanishes and the two expressions (equations 5.7 and 5.13) become identical. Hence under the continuous framework the object extraction problem can be solved by using a neural network whose energy function takes the form

$$E = - \sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \sum_i \frac{1}{R_i} \int_0^{V_i} g^{-1}(V) dV. \quad (5.16)$$

The search process can be activated by solving for the input U_i , $\forall i$ from the following differential equations

$$- \frac{\partial E}{\partial V_i} = (\sum_j W_{ij} V_j + I_i - U_i) = \frac{dU_i}{dt}. \quad (5.17)$$

Once the U_i values are known, equation (5.10) can be applied to get the output status V_i .

5.5 Simulation and results

We note from equation (5.17) that in order to get the input to all the neurons of the network (of size $M \times N$) at an instant $(t + \Delta t)$ one has to solve $M \times N$

differential equations with initial values at time t . For this purpose we used Euler method, i.e., we iterate (from equation 5.17) as

$$U_i(t + \Delta t) = U_i(t) + \Delta t \left\{ \sum_j W_{ij} V_j(t) + I_i - U_i(t) \right\} \quad (5.18)$$

until convergence. Numerical solutions for these differential equations require a stopping criterion which can be taken as

$$|U_i(t + \Delta t) - U_i(t)| < \epsilon, \quad \forall i \quad (5.19)$$

where ϵ is a pre-assigned small positive quantity. The present simulation study used $\Delta t \sim 10^{-5}$ and $\epsilon \sim 10^{-6}$.

The network is assumed to attain a stable state if for every neuron i , $|V_i(t) - V_i(t + \Delta t)| < \epsilon_1$, where ϵ_1 is another pre-assigned small positive quantity. The network may require a sufficiently large time to achieve the exact value of -1 (OFF) or +1 (ON). Obviously, if the precision level is increased (value of ϵ_1 is decreased) the network takes more time to converge. Quick convergence can be obtained by increasing Δt or ϵ_1 .

In this context it may be mentioned that the input bias of a neuron was taken to be

$$I_i = 2 \frac{i}{L} - 1 \quad (5.20)$$

where i is the gray value of the corresponding pixel and L is the maximum gray value. The input/output transfer function used was a second order polynomial (equation 5.10 with $n = 2$).

To check the validity and effectiveness of the proposed methods, a computer simulation is done using some synthetic images [Figures 5.5(a), 5.6(a), 5.7(a)] images and some real images [Figures 4.4(a), 2.2(a), 3.7(c)]. The original synthetic image [Figure 5.4] had only two levels which are then corrupted by additive $N \sim (0, \sigma^2)$ noise with different σ (5, 15, 24) values. The corruption level is made very high so as to make the signal to noise ratio $SNR=1.0$, where SNR is defined as

$$\frac{\text{range of gray levels}}{\sigma}$$

Three different noisy versions of Figure 5.4 and the corresponding extracted objects with both discrete and continuous dynamics of the network, and N^1 and N^2 neighborhood systems are given in Figures 5.5-5.7. One of the real images used as input is a *Noisy tank* [Figure 4.4(a)]. The outputs obtained by the discrete and the continuous models are depicted in Figures 5.8(b)-5.8(e). For these noisy images the connection strengths were initialized to unity (as mentioned in section 5.4.2).

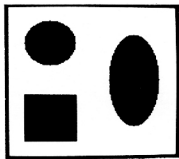


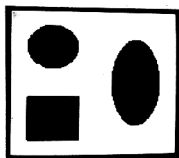
Figure 5.4: Original synthetic image

The algorithms are then tested on the original (uncorrupted version) and a noisy version [Figure 3.7(c)] of the *Biplane* image in order to demonstrate the effect of appropriate initialization of weights on segmented output. The extracted outputs for the original *Biplane* image with discrete and continuous models (with N^1 and N^2) are shown in Figure 5.9 when the weights are assigned depending on the joint distribution of gray levels (equation 5.15). Figure 5.10 represents the corresponding outputs with $W_{ij} = 1$. The similar sets of outputs for the noisy version [Figure 3.7(c)] of the *Biplane* image are shown in Figures 5.11 and 5.12.

Examining the results [Figures 5.5-5.8 and 5.12] it can be said that the outputs obtained by the continuous dynamics are better than those obtained by the discrete version for noisy images. It is seen that for high values of *SNR* (low noise), the results obtained by the two approaches are very close. Since the discrete dynamics takes much less time to stabilize, this model is preferred for



(a)



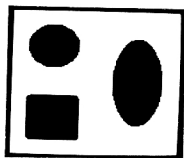
(b)



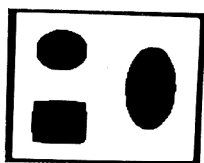
(c)



(d)

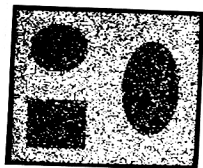


(e)

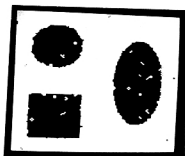


(f)

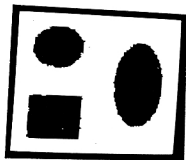
Figure 5.5: A noisy version of the synthetic image with $SNR=4.8$: a) input, b) output by discrete model with N^1 , c) output by continuous model with N^1 , d) output by discrete model with N^2 , e) output by continuous model with N^2 , f) output by relaxation method with N^2



(a)



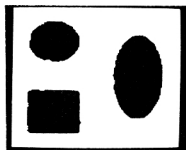
(b)



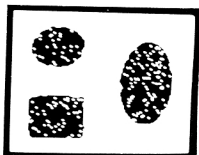
(c)



(d)

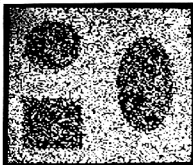


(e)



(f)

Figure 5.6: A noisy version of the synthetic image with $SNR=1.6$: a) input, b) output by discrete model with N^1 , c) output by continuous model with N^1 , d) output by discrete model with N^2 , e) output by continuous model with N^2 , f) output by relaxation method with N^2



(a)



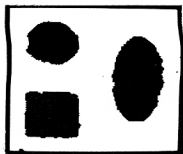
(b)



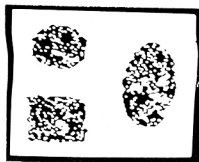
(c)



(d)



(e)



(f)

Figure 5.7: A noisy version of the synthetic image with $SNR=1.0$: a) input, b) output by discrete model with N^1 , c) output by continuous model with N^1 , d) output by discrete model with N^2 , e) output by continuous model with N^2 , f) output by relaxation method with N^2

see Figure 4.4(a)

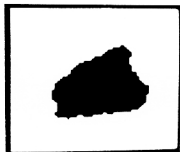


(a)

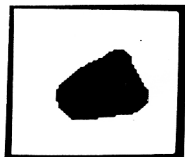
(b)



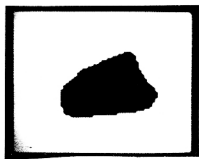
(c)



(d)



(e)



(f)

Figure 5.8: *Noisy tank image* : a) input, b) output by discrete model with N^1 , c) output by continuous model with N^1 , d) output by discrete model with N^2 , e) output by continuous model with N^2 , f) output by relaxation method with N^2

see Figure 2.2(a)

(a)



(b)



(c)



(d)



(e)

Figure 5.9: *Biplane* image : a) input, b) output by discrete model with N^1 (W_{ij} by equation 5.15), c) output by continuous model with N^1 (W_{ij} by equation 5.15), d) output by discrete model with N^2 (W_{ij} by equation 5.15), e) output by continuous model with N^2 (W_{ij} by equation 5.15)

see Figure 2.2(a)

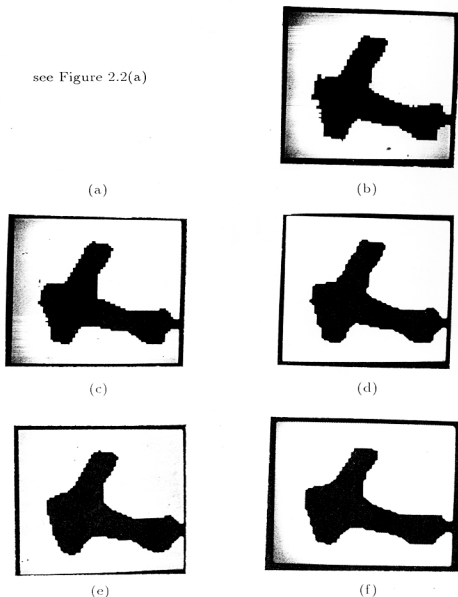
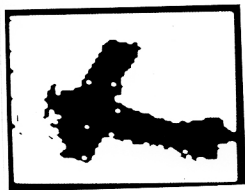


Figure 5.10: *Biplane* image : a) input, b) output by discrete model with N^1 , c) output by continuous model with N^1 , d) output by discrete model with N^2 , e) output by continuous model with N^2 , f) output by relaxation method with N^2

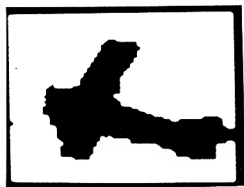
see Figure 3.7(c)

(a)



(b)

(c)



(d)



(e)

Figure 5.11: Noisy *Biplane* image : a) input, b) output by discrete model with N^1 (W_{ij} by equation 5.15), c) output by continuous model with N^1 (W_{ij} by equation 5.15), d) output by discrete model with N^2 (W_{ij} by equation 5.15), e) output by continuous model with N^2 (W_{ij} by equation 5.15)

see Figure 3.7(c)

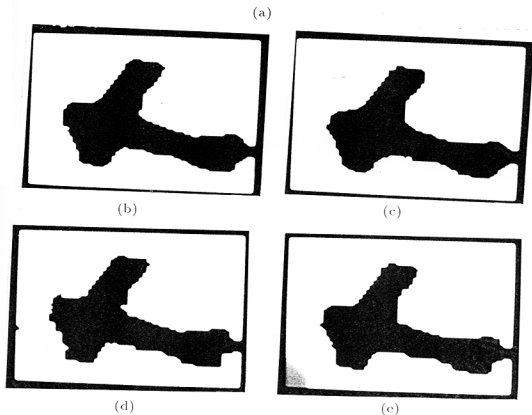


Figure 5.12: Noisy *Biplane* image : a) input, b) output by discrete model with N^1 , c) output by continuous model with N^1 , d) output by discrete model with N^2 , e) output by continuous model with N^2

higher values of SNR . It is also noticed from the results that with decrease of SNR , there is a slight deterioration, as expected, in the quality of outputs.

Moreover, it is clear from the results that N^1 connectivity is more noise sensitive than N^2 connectivity. But for higher values of SNR (i.e., for less corrupted images) the outputs are comparable. Thus for less corrupted images, N^1 connectivity with discrete dynamics can be used; thereby saving both space and time.

It may further be noted from the results that the proposed algorithms work better for noisy images than the method described in the previous chapter (see Figures 5.8, 5.11, 4.4 and 4.7 where the same set of input was considered). Some of the stray points observed in Figures 4.4 and 4.7 are seen to be absent in Figures 5.8 and 5.11. The incorporation of the consensus from the neighbors during status updation process of the proposed algorithm may increase the possibility of losing some portion of thin regions.

From the results [Figure 5.9] of the original *Biplane* image it is seen that the weight assignment procedure using equation (5.15) is fair enough to extract the exact shape of the object including the propeller of the *Biplane*. This is not true when the weights are considered to be unity [see Figure 5.10]. On the contrary, the outputs for the noisy version of the *Biplane* image [Figures 5.11 and 5.12] are seen to be better when $W_{ij} = 1$.

To establish the superiority of the proposed method over the existing iterative algorithms a result wise comparison is made with the probabilistic relaxation technique (which has been described in Chapter 1, pp. 12). For the computation of the compatibility function in the relaxation method the following formula is used :

$$C(i, j; h, k) = \frac{p(a_i \in C_j, a_h \in C_k)}{p(a_i \in C_j) \times p(a_h \in C_k)} \quad (5.21)$$

where $C(i, j; h, k)$ gives a measure of the compatibility that the element a_i belongs to the class C_j and a_h to C_k , $p(a_i \in C_j)$ is the apriori probability of $a_i \in C_j$. The results obtained with N^2 connectivity are included in Figures 5.5-5.10. From the results it is evident that the objects extracted by the proposed

techniques preserve the shapes and the approximate outlines of the actual input in a better way than those obtained by the relaxation algorithm.

An objective evaluation of the results has been made based on the percentage of correct classification of pixels. The numerical results are depicted in Table 5.1. From the table it is found that for all different types of connectivity and noise levels the percentage of pixels correctly classified by the present method is higher than that of the relaxation algorithm.

Table 5.1: Classification accuracy of different approaches

Synthetic image with $\sigma =$	Connectivity	Percentage of correct classification		
		neural network algorithms		relaxation technique
		discrete	continuous	
5	N^1	99.93	99.71	99.43
15	N^1	98.69	99.23	97.89
24	N^1	92.25	96.47	90.78
5	N^2	99.90	99.71	98.58
15	N^2	99.28	99.40	97.55
24	N^2	98.54	98.68	92.80

5.6 Conclusions and discussion

A modified version (topologically) of Hopfield's model has been designed to solve the image segmentation problem. A single neuron is assigned to each pixel. The energy function is formulated in such a manner that in a stable state of the network neurons corresponding to homogeneous regions will be in ON (OFF) state. Both the discrete and the continuous dynamics are studied. The problem of assignment of weights has been tackled using the concept of probability of co-occurrence of pixels.

The system is found to provide better performance for noisy images as compared

to the methods developed in the previous chapter. This is because of the fact that the status updating procedure of a neuron takes (unlike the algorithms in Chapter 4) consensus from its neighbors. This, of course, increases the possibility of losing some of the thin regions. (Some more results of these algorithms for low SNR values have been provided in the next chapter for comparison.) The guideline provided for the assignment of connection strengths seems to be appropriate; thereby resulting in better segmented output than the probabilistic relaxation algorithm.

The reason for inclusion of the synthetic images in Figures 5.5(a), 5.6(a) and 5.7(a) (having known objects) as test patterns is to verify the effectiveness of the proposed model in extracting compact regions from highly noisy images where the algorithms described in Chapter 4 usually show inferior performance. These sets of test patterns have also been used in the subsequent chapters with other noise levels. Since the objects in these images are known it enabled us to formulate an objective evaluation technique in Chapter 8 for a comparative study of the methodologies developed in Chapters 4-7.

Chapter 6

Gibbs distribution and Hopfield
type neural network for object
extraction

6.1 Introduction

In the previous chapter an object extraction algorithm using a Hopfield type neural network has been described. The architecture essentially acts as an auto-associative memory and employs associative recall to find out the object regions in a scene. In the present chapter we formulate another object extraction algorithm using a similar type of neural network having difference in the neuron characteristics. Here also, the problem of object extraction is viewed as an optimization task. In the previous chapter no apriori knowledge has been used to model the scene, but in the present case the scene is modeled by a *Gibbs Random Field*. The maximum aposterior probability estimate of the scene is taken as the solution for object extraction problem.

Computationally hard problems sometimes have exponential complexity. Therefore, when the size of the problem is very large computation becomes prohibitive. However, in practice one often needs a reasonably good solution (may not be the optimal one). When dealing with such problems, one normally finds several sub-optimal solutions and accepts the best one. Since neural networks converge very quickly and attempts to provide output in real time one can find several solutions in a short time and get the best result.

Hopfield and Tank [139] have shown that neural networks can be used to solve computationally hard problems, in particular they applied it to the *Traveling Salesman Problem (TSP)*. To investigate how well the network performs they simulated its behaviour and found very encouraging results. It was observed that the network frequently finds valid solutions of high quality.

Use of statistical techniques for modeling and processing image data is very common in computer vision. A specific example is the use of Markov Random Field (*MRF*) to model real life images. Cohen and Cooper [283], Geman and Geman [73], and Derin and Elliott [74] used *Gibbs distribution (GD)* for characterizing an *MRF*, since a Gibbs Random Field (*GRF*) can be viewed as an *MRF* for a large neighborhood system [284]-[287]. Most of the above mentioned authors used maximum aposterior probability (*MAP*) estimation as

the criterion for segmentation/restoration of noisy images modeled as a *GRF*. But the computation of *MAP* estimate of the scene in this type of problem is very hard. The use of *GD* for segmentation and restoration of a noisy image requires $L^{M \times N}$ possible combinations of its structure to be searched to get the best possible solution (L = number of discrete levels that the random variable can have, $M \times N$ is the size of the image). Thus the problem is having exponential complexity. With increase in size of the scene it becomes practically unfeasible to find its *MAP* estimate. Usually under this situation heuristic solutions [74] are suggested. Since most of the heuristic methods cannot be implemented in parallel, the output still cannot be obtained in real time. By using parallel processing and exploiting the collective computational abilities of the neural networks, the search space can be reduced drastically even after getting a substantially good solution.

The work presented in this chapter is an attempt to solve the above mentioned computationally hard problem of finding the *MAP* estimate of a scene with a neural network. To realize this, a neural network model which is a modified version (both topologically and neuron characteristics wise) of that of Hopfield [138] is proposed here. For the hardware realization of a neuron used in the proposed model, a negative feed back from the output is required in addition to the circuitry involved for the realization of a Hopfield neuron. The topology of the present network is similar to that developed in Chapter 5. In other words, a single neuron is assigned for every pixel and every neuron is assumed to be connected to its neighbors (number of neighbors may vary depending on the problem). The input bias of a neuron is a transformed version of the gray level of the corresponding pixel.

For the simulation study, the input bias to a neuron is taken to be linearly proportional to the gray value of the corresponding pixel. The input/output to/from a neuron is restricted to lie in $[-1,1]$ during transition (updating stage). When the network converges, the output status of a neuron is either -1 or +1. The status updating rule of the neurons is formulated in such a way that the resultant image approaches the *MAP* estimate of the scene by gradient descent search. In other words, the energy function of the network is so formulated

that the global minimum of it corresponds to the *MAP* estimate of the scene.

As in Chapter 5, investigation has been done under both discrete and continuous models. The simulation study is done using a few synthetic images corrupted by noise as well as some real (noisy) images. Since the proposed algorithms use apriori knowledge for modeling scenes they are found to be more robust and noise immune than the methods described in Chapter 5.

6.2 Background on Gibbs distribution

Here we present the basic definition of a particular class of Gibbs distribution (*GD*) which are mainly used for modeling images. For an extensive treatment on *GD*, related to the *MRF* and the equivalence of the two one can refer to the articles [284]-[287].

6.2.1 Basic definition of Gibbs distribution

We focus our attention on 2-D random fields defined over a finite $M \times N$ rectangular lattice of points (pixels) characterized by

$$R = \{(i, j) : 1 \leq i \leq M, 1 \leq j \leq N\}. \quad (6.1)$$

Let us consider the concept of d th order neighborhood system $N^d = \{N_{ij}^d\}$ on the lattice R as defined in section 5.3 (pp. 127).

The cliques associated with the d th order neighborhood system N^d are defined as follows.

Definition 1 : A clique for the d th order neighborhood system N^d defined on the lattice R , denoted by c , is a subset of R such that

- a) c consists of a single pixel, or
- b) for $(i, j) \neq (k, l)$, $(i, j) \in c$ and $(k, l) \in c$ implies $(i, j) \in N_{kl}^d$.

For illustration, first and second order neighborhoods (N^1 and N^2) and the associated cliques are shown in Figure 6.1.

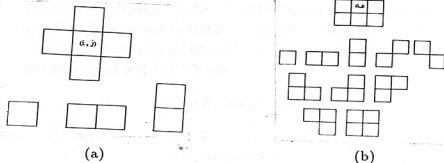


Figure 6.1: Neighborhood and associated cliques : a) first order, b) second order

Now a Gibbs distribution (*GD*) can be defined as follows.

Definition 2 : Let N^d be the d th order neighborhood system defined on a finite lattice R . A random field $X = \{X_{ij}\}$ defined on R has a *GD* or equivalently is a Gibbs Random Field (*GRF*) with respect to N^d , if and only if, its joint distribution is of the form

$$p(X = x) = \frac{1}{Z} e^{-U(x)} \quad (6.2)$$

with

$$U(x) = \sum_{c \in C} V_c(x) \quad (6.3)$$

where $V_c(x)$ is the potential associated with the clique c , C is the collection of all possible cliques and

$$Z = \sum_x e^{-U(x)}. \quad (6.4)$$

Here Z , the partition function, is a normalizing constant, and $U(x)$ is called the energy of the realization.

The joint distribution expression in equation (6.2) says that with a decrease in $U(x)$ the probability $p(X = x)$ increases. In other words, the smaller $U(x)$, the energy of the realization x , the more likely is the realization [i.e., larger $p(X = x)$] and vice versa.

A Gibbs distribution is basically an exponential distribution. However, by choosing the clique potential function $V_c(x)$ properly a wide variety of distributions, both for discrete and continuous random fields, can be formulated as *GD*. In this context it may be mentioned that any random field can be viewed as a Markov Random Field (*MRF*) when

$$N_{ij}^d = R, \forall (i, j) \in R$$

and every *MRF* with

$$p(X = x) > 0, \forall x$$

as a *GRF* and vice versa [284, 285]. Hence a scene realization can be taken as a *GRF*.

6.2.2 Gibbs distribution used for scene modeling

Let us now define a particular class of *GD* which is suitable for modeling scenes consisting of different regions with uniform intensities. We assume that the random field X consists of L -valued discrete random variables $\{X_{ij}\}$ taking values in $Q = \{q_1, q_2, \dots, q_l\}$. To define a *GD* it suffices to define the neighborhood system N^d and the associated clique potentials $V_c(x)$. The clique potentials are assumed to depend on the clique type and the pixel values in the cliques but not on the position of the cliques.

It has been further found that the second order neighborhood system is sufficient for modeling the spatial dependencies of a scene consisting of several objects. Hence we shall concentrate on the cliques associated with N^2 . Extension to higher order neighborhood systems and restriction to lower ones are self-evident.

In order to get the clique potential a parameter is associated with every clique type. This may be as follows

$$\begin{aligned} & [**, \beta_1], [*, \beta_2], [*, \beta_3], [*, \beta_4], \\ & [**, \gamma_1], [**, \gamma_2], [**, \gamma_3], [**, \gamma_4], [**, \delta_1], \end{aligned}$$

where a certain combination of $*$ s represent a particular type of clique. For example, $**$ and $*$ represent the second and fourth cliques of Figure 6.1(b).

The associated clique potentials are then assigned as

$$V_c(x) = \begin{cases} -\beta_i \text{ or } -\gamma_i \text{ or } -\delta_i, & \text{if all } x_{ij}\text{s are equal} \\ +\beta_i \text{ or } +\gamma_i \text{ or } +\delta_i, & \text{otherwise} \end{cases} \quad (6.5)$$

where β_i, γ_i and δ_i are positive quantities.

For single pixel cliques, the clique potential is defined as

$$V_c(x) = \alpha_k \text{ for } x_{ij} = q_k.$$

Here α_k is proportional to the number of pixels in the k th region type.

6.3 Image model and problem formulation

In this section the formulation of the object extraction algorithm which finds maximum a posterior probability (*MAP*) estimate of a scene is discussed.

6.3.1 Gibbsian model for image data

A digital image $y = \{y_{ij}\}$ can be described as an $M \times N$ matrix of observations. It is assumed that the matrix y is a realization from a random field $Y = \{Y_{ij}\}$. The lattice associated with this is the collection of $M \times N$ pixels $\{(i, j)\}$. The objective is to specify the random field Y so that it characterizes a class of noisy images under consideration.

The random field Y is defined in terms of the underlying random field $X = \{X_{ij}\}$ (scene). The scene random field X is a discrete valued one, where X_{ij} can take values in $Q = \{q_1, q_2, \dots, q_L\}$ for each $(i, j) \in R$. A realization $X = x$ is a partitioning of the lattice into L region types such that $x_{ij} = q_k$, if the pixel (i, j) belongs to the k th region type. Each region type can occur in more than one location in the lattice.

The Gibbsian distribution basically characterizes spatial clustering of pixels into regions. The distribution emphasizes spatial continuity, i.e., if a pixel belongs to region type k , then the probability of its neighboring pixels to belong to the same region type is very high. The novelty of the *GRF* lies in the fact that it takes into account the statistical information pertaining to size, shape, orientation and frequency of the regions.

Regions are assumed to have uniform intensities and the realized image is a corrupted version of the actual one, corrupted by white (additive independent and identically distributed, i.e., i.i.d.) noise. So Y can be written as

$$Y_{ij} = F(X_{ij}) + W_{ij}, \quad \forall (i, j) \in R \quad (6.6)$$

where $\{W_{ij}\}$ is i.i.d noise. It is also assumed that $W \sim N(0, \sigma^2)$. The function $F(\cdot)$ is a simple mapping of the region type to the corresponding gray level. In other words,

$$F(X_{ij}) = q_m \quad \text{if } X_{ij} = m. \quad (6.7)$$

6.3.2 Object extraction technique

Under the above discussed model the object extraction problem is simply the determination of the scene realization x that has given rise to the actual noisy image y . The realization x can not be obtained deterministically from y . So the problem is to estimate $\hat{x}$ of the scene X , based on the realization y . The statistical criterion of maximum aposterior probability (*MAP*) can be used to estimate the scene. The objective in this case is to have an estimation rule which yields $\hat{x}$ that maximizes the aposterior probability $p(X = x|Y = y)$.

From Bayes rule [45, 46, 288] we can write

$$p(X = x|Y = y) = \frac{p(Y = y|X = x) \times p(X = x)}{p(Y = y)}. \quad (6.8)$$

Since y does not affect the maximization process, it is equivalent to maximize only the numerator of the right hand side of equation (6.8), or its logarithm

$$\ln p(Y = y|X = x) + \ln p(X = x). \quad (6.9)$$

The second component of equation (6.9) can be written as (from equations (6.2) and (6.3))

$$\begin{aligned}\ln P(X = x) &= -\ln Z - \sum_{c \in C} V_c(x) \\ &= C_1 - \sum_{c \in C} V_c(x)\end{aligned}\quad (6.10)$$

where $C_1 = -\ln Z$ is a constant.

To simplify the first part of equation (6.9) we proceed as follows.

From equations (6.6) and (6.7) we can write

$$p(Y_{ij} = y_{ij} | X_{ij} = x_{ij}) = p(W_{ij} = w_{ij}). \quad (6.11)$$

For independent distributions, the joint probability of occurrence of a number of events can be expressed as the product of the probabilities of their independent occurrences. Thus,

$$\begin{aligned}p(Y = y | X = x) &= \prod_{i=1}^M \prod_{j=1}^N p(Y_{ij} = y_{ij} | X_{ij} = x_{ij}) \\ \text{or, } \ln p(Y = y | X = x) &= \sum_{i=1}^M \sum_{j=1}^N \ln p(Y_{ij} = y_{ij} | X_{ij} = x_{ij}) \\ &= \sum_{i=1}^M \sum_{j=1}^N \ln p(W_{ij} = w_{ij}) \quad [\text{from equation (6.11)}] \\ &= \sum_{i=1}^M \sum_{j=1}^N \ln \left\{ \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(w_{ij})^2} \right\} \\ &= -\frac{1}{2} \sum_{i=1}^M \sum_{j=1}^N \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^M \sum_{j=1}^N (w_{ij})^2 \\ &= -\frac{M \times N}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^M \sum_{j=1}^N (y_{ij} - x_{ij})^2.\end{aligned}$$

Since there are L region types, we can write

$$\ln p(Y = y | X = x) = -\frac{M \times N}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{m=1}^L \sum_{(i,j) \in S_m} (y_{ij} - q_m)^2$$

where $S_m = \{(i, j) \in R : x_{ij} = q_m\}$.

Thus,

$$\ln p(Y = y | X = x) = -C_2 - \frac{1}{2\sigma^2} \sum_{m=1}^L \sum_{(i,j) \in S_m} (y_{ij} - q_m)^2 \quad (6.12)$$

with $C_2 = \frac{M \times N}{2} \ln(2\pi\sigma^2) = \text{constant}$.

The difficulty in determining $\hat{x}$ is due to the fact that the maximization of equation (6.8) or (6.9) requires $L^{M \times N}$ possible scene configurations to be searched and thus makes the problem computationally hard. In the following sections, we shall exploit the collective computational abilities of a neural network to get the *MAP* estimate of a scene. In this context it may be mentioned that a link between Markov model and multi-layer perceptron type network has recently been established in [289].

Note that the topology of the network and the input-output transfer function used here are the same as those described in Chapter 5 (section 5.3).

6.4 Algorithm for solving the MAP estimation problem with a neural network

In the present section we shall be describing the procedure of finding the *MAP* estimate of a scene with a neural network.

6.4.1 Formulation of the algorithm

From equations (6.9)-(6.12) it is noticed that the problem is to maximize

$$\ln p(X = x) + \ln p(Y = y|X = x) \quad (6.13)$$

where

$$\ln p(X = x) = - \sum_{c \in C} V_c(x) + C_1 \quad (6.14)$$

and

$$\ln p(Y = y|X = x) = -\frac{1}{2\sigma^2} \sum_{m=1}^L \sum_{(i,j) \in S_m} (y_{ij} - q_m)^2 - C_2 \quad (6.15)$$

where $S_m = \{(i, j) \in R : x_{ij} = q_m\}$, and C_1 and C_2 are constants.

In this study we shall consider only binary (two tone) images corrupted by noise. In other words, the proposed method confines itself to the case when $L = 2$. Under this situation, without loss of generality we can assume that the gray values of the realized image lie in $[-1, 1]$. It has already been discussed in section 6.2 that even for the second order neighborhood system (N^2) different types of cliques are possible. For the sake of simplicity we shall restrict ourselves only to the cliques with two pixels. Further, we will not distinguish between the different possible types of cliques with two pixels. It has already been mentioned that if a pixel belongs to region type k , the probability of its neighboring pixels being in the same region type is very high. This suggests that if a pair of adjacent pixels have similar values then the potential of the corresponding clique should increase the value of $p(X = x)$, i.e., the potential of the clique associated with these two pixels should be negative. If the values V_i and V_j , of two adjacent pixels are of the same sign, then $V_c(x)$ should be negative, otherwise positive.

One possible choice of the clique potential $V_c(x)$, of a clique c of the realization x , containing i th and j th pixels is

$$V_c(x) = -W_{ij}V_iV_j \quad (6.16)$$

where W_{ij} is a constant for particular i and j , and $W_{ij} > 0$. This W_{ij} can be viewed as the connection strength between the i th and j th neurons and V_i as the output of the i th neuron.

In equation (6.12) y_{ij} is the realization of the modeled value x_{ij} of the (i, j) th pixel and q_m is the value of the (i, j) th pixel in the underlying scene, i.e., $q_m = x_{ij}$. For a neural network one can easily interpret x_{ij} as the present status of the (i, j) th neuron and y_{ij} as the initial bias of the same neuron. Thus we can write

$$\begin{aligned} & \sum_m \sum_{i,j} (y_{ij} - q_m)^2 \\ &= \sum_m \sum_{i,j} (y_{ij})^2 - 2 \sum_m \sum_{i,j} y_{ij} q_m + \sum_m \sum_{i,j} q_m^2 \\ &= C_3 - 2 \sum_m \sum_{i,j} I_{ij} V_{ij} + \sum_m \sum_{i,j} V_{ij}^2 \end{aligned} \quad (6.17)$$

using $y_{ij} = I_{ij}$ and $q_m = V_{ij}$. Here $C_3 = \sum_m \sum_{i,j} (y_{ij})^2$ is a constant.

This can also be written in the form (replacing the two dimensional indices by one dimensional index)

$$C_3 - 2 \sum_i I_i V_i + \sum_i V_i^2. \quad (6.18)$$

From equations (6.14)-(6.18) it can be said that maximization of equation (6.13) is identical to maximization of

$$C_1 + \sum_{i,j} W_{ij} V_i V_j - \frac{1}{2\sigma^2} (C_3 - 2 \sum_i I_i V_i + \sum_i V_i^2) - C_2$$

i.e., maximization of (excluding the constant terms)

$$\sum_{i,j} W_{ij} V_i V_j + \frac{1}{\sigma^2} \sum_i I_i V_i - \frac{1}{2\sigma^2} \sum_i V_i^2 \quad (6.19)$$

or, minimization of

$$- \sum_{i,j} W_{ij} V_i V_j - \frac{1}{\sigma^2} \sum_i I_i V_i + \frac{1}{2\sigma^2} \sum_i V_i^2. \quad (6.20)$$

6.4.2 Realization with a neural network

The problem is now reduced to the minimization of the expression in equation (6.20) with a neural network. To do that, it suffices to establish an equivalence between equation (6.20) and the expression of energy of a neural network, and to provide an updating rule so that the convergence is guaranteed. The state of the network changes with time t , because of the interactions of the neurons. The dynamics of the network are governed by differential equations which are obtained from

$$- \frac{\partial E}{\partial V_i} = \frac{dU_i}{dt}. \quad (6.21)$$

Discrete Case :

For the discrete model the neurons are having two possible states only, and the

updating rule is : if the total input to a neuron crosses a certain limit then it attains one state else, the other one.

Now in this model a neuron is either ON (1) or OFF (-1) (i.e., $V_i=1$ or -1). Hence $\sum_i V_i^2 = \text{constant}$. So the expression to be minimized takes the form (from equation 6.20)

$$E = - \sum_{i,j} W_{ij} V_i V_j - \frac{1}{\sigma^2} \sum_i I_i V_i. \quad (6.22)$$

This equation is equivalent to equation (5.3) (pp. 126) with $\theta = 0$ and a scaling over I_i . Hence the network structure described in Chapter 5 (section 5.3, pp. 127) with discrete dynamics can be used for obtaining the minima of the expression in equation (6.22). In other words, the network can solve the image segmentation (object extraction) problem by estimating the most probable scene. Here

$$\Delta E = - \left[\sum_j W_{ij} V_j + \frac{1}{\sigma^2} I_i \right] \Delta V_i \quad (6.23)$$

and the state updating rule is

$$V_i = \begin{cases} -1 & \text{if } \sum_j W_{ij} V_j + \frac{1}{\sigma^2} I_i < 0 \\ 1 & \text{otherwise.} \end{cases} \quad (6.24)$$

Continuous Case :

In this model the variables are assumed to be continuous. Let the output variable V_i for the i th neuron lie in the range $[-1, 1]$. V_i is obtained by applying a monotonically increasing non-linear transfer function (g) of the instantaneous total input U_i to the same neuron. The typical input/output relation is sigmoidal with asymptotes -1 and $+1$ (equation 5.6, pp. 126).

The gain function g can be realized by a non-linear operational amplifier with negligible response time. A neural network whose neurons are having hardware realization as shown in Figure A.2 can not be used for the minimization of equation (6.20) due to the presence of the last term. With a view to realize the expression (equation 6.20) with a neural network let us consider the electrical

circuit as shown in Figure 6.2. The circuit is similar to that of Figure A.2 except the feed back connection. It will be seen that the feed back connection will take account of the extra term in equation (6.20).

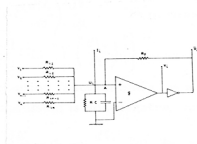


Figure 6.2: Electrical circuit equivalent of a neuron of the proposed model

Let U_i be the total input to the amplifier (potential at point A, Figure 6.2) with gain function g . Then applying Kirchoff's current law at point A we get

$$\sum_{j=1}^n \frac{V_j - U_i}{R_{ij}} + I_i + \frac{-V_i - U_i}{R_f} = \frac{U_i}{R} + C \frac{dU_i}{dt}$$

or,

$$C \frac{dU_i}{dt} = \sum_{j=1}^n \frac{V_j}{R_{ij}} - \sum_{j=1}^n \frac{U_i}{R_{ij}} + I_i - \frac{-U_i}{R} - \frac{-U_i}{R_f} - \frac{-V_i}{R_f}.$$

Hence,

$$C \frac{dU_i}{dt} = \sum_{j=1}^n W_{ij} V_j - \frac{U_i}{R_i} - \frac{V_i}{R_f} + I_i \quad (6.25)$$

where $W_{ij} = \frac{1}{R_{ij}}$ and $\frac{1}{R_i} = \sum_{j=1}^n \frac{1}{R_{ij}} + \frac{1}{R} + \frac{1}{R_f}$.

Here R_i is the total input resistance of the amplifier, and the output impedance of the amplifier is considered negligible. The circuit in the above case basically realizes a neuron having gain $g(> 0)$ and a negative self feed back. The V_j s ($j = 1, 2, \dots, n$) are actually the inputs to the i th neuron from other neurons and W_{ij} s are the connection strengths.

Now let us consider the quantity

$$E = - \sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \frac{1}{2 R_f} \sum_i V_i^2 + \sum_i \frac{1}{R_i} \int_0^{V_i} g^{-1}(V) dV. \quad (6.26)$$

Here

$$\begin{aligned}
\frac{dE}{dt} &= \sum_i \frac{\partial E}{\partial V_i} \frac{dV_i}{dt} \\
&= \sum_i \left(-\sum_j W_{ij} V_j - I_i + \frac{1}{R_f} V_i + \frac{U_i}{R_i} \right) \frac{dV_i}{dt} \\
&= \sum_i \left(-C \frac{dU_i}{dt} \right) \frac{dV_i}{dt} \quad [\text{from equation (6.25)}] \\
&= -\sum_i \left\{ C(g^{-1})' \left(\frac{dV_i}{dt} \right)^2 \right\}.
\end{aligned}$$

Since g is a monotonic increasing function and C is a positive constant, every term in $\{\cdot\}$ of the above expression is non-negative, and hence $\frac{dE}{dt} \leq 0$, and

$$\frac{dE}{dt} = 0 \implies \frac{dV_i}{dt} = 0, \forall i.$$

E is easily seen to be bounded. As $\frac{dE}{dt} \leq 0$, searching in the negative direction of gradient will lead us to a minimum of E . So, the evolution of the system is a motion in the state space that seeks out minima in E and stops at such points. Thus, E can be considered an energy function of the network shown in Figure 5.2 (pp. 128) with neurons electrically equivalent to Figure 6.2. The last term in equation (6.26) is the energy loss of the system. In the high gain limit [138, 139], the stable states of the network corresponds to the local minima of the quantity

$$E = -\sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \frac{1}{2R_f} \sum_i V_i^2. \quad (6.27)$$

The equations (6.20) and (6.27) are equivalent with proper adjustment of coefficients. So the minimal values of the expression in equation (6.20) can be easily determined with a neural network having the energy function as in equation (6.26). The topology of the network is shown in Figure 5.2 (pp. 128), and the realization of a neuron in Figure 6.2.

In order to get the equations governing the dynamics of the network, let us consider the energy function

$$E = -\sum_i \sum_j C W_{ij} V_i V_j - \frac{C}{\sigma^2} \sum_i V_i I_i + \frac{C}{2\sigma^2} \sum_i V_i^2 + \sum_i \frac{1}{R_i} \int_0^{V_i} g^{-1}(V) dV. \quad (6.28)$$

Here

$$-\frac{\partial E}{\partial V_i} = \sum_j CW_{ij}V_j + \frac{C}{\sigma^2}I_i - \frac{C}{\sigma^2}V_i - \frac{U_i}{R_i}.$$

Differentiating equation (6.26) with respect to V_i and equating with (6.25) we notice that

$$\begin{aligned} C\frac{dU_i}{dt} &= -\frac{\partial E}{\partial V_i} \\ &= \sum_j CW_{ij}V_j + \frac{C}{\sigma^2}I_i - \frac{C}{\sigma^2}V_i - \frac{U_i}{R_i}. \end{aligned}$$

Hence

$$\frac{dU_i}{dt} = \sum_j W_{ij}V_j + \frac{I_i}{\sigma^2} - \frac{V_i}{\sigma^2} - \frac{1}{\tau}U_i \quad (6.29)$$

where $\tau = RC$.

Thus the differential equation (6.29) will govern the state of the system.

To find a solution, we select an initial state of the network at random, and let it evolve according to equation (6.24) or (6.29) for discrete and continuous cases, respectively. It will eventually reach a steady state (a minimum of E) and stop. To get further improvement in the solution one can use the simulated annealing [291] type of approach. However, from the experimental results it will be seen that, for this type of problem annealing is not required.

6.5 Simulation and results

To check the validity and effectiveness of the proposed technique, a computer simulation has been done using synthetic and real images. The original synthetic image [Figure 5.4, pp. 135] had only two levels which are then corrupted by additive $N \sim (0, \sigma^2)$ noise with different σ (10, 20, 32) values. The corruption level is made very high so as to make the signal to noise ratio (SNR), defined in the previous chapter (pp. 135) as small as 0.75. The real images used are noise corrupted versions of a tank (*Noisy tank*) [Figure 4.4(a)] and a *Biplane* [Figure 3.7(c)].

It is to be noted from equation (6.29) that in order to get the input to all $M \times N$ neurons of a network at an instant $(t + \Delta t)$ (continuous case), one has to solve $M \times N$ differential equations with given initial values at time t . For this we used Euler method, i.e., we iterated (from equation 6.29)

$$U_i(t + \Delta t) = U_i(t) + \Delta t \left\{ \sum_j W_{ij} V_j(t) + \frac{I_i}{\sigma^2} - U_i(t) - \frac{V_i(t)}{\sigma^2} \right\} \quad (6.30)$$

until convergence. Here τ is set equal to 1. A numerical solution for these differential equations requires a stopping criterion which was (as in the previous chapter)

$$|U_i(t + \Delta t) - U_i(t)| < \epsilon, \quad \forall i$$

where ϵ is a pre-assigned small positive quantity. In the present simulation study we used $\Delta t \sim 10^{-5}$ and $\epsilon \sim 10^{-6}$.

As in the previous chapter the network is assumed to attain a stable state if for every neuron i , $|V_i(t) - V_i(t + \Delta t)| < \epsilon_1$, where ϵ_1 is a pre-assigned small positive quantity. If the precision level is increased (value of ϵ_1 is decreased) the network takes more time to converge. Quicker convergence can be obtained by increasing Δt or ϵ_1 .

In the practical case, even after a sufficiently long time, the neurons may not achieve the exact value of -1 (OFF) or +1 (ON). However, if the activity level is reasonably close to -1 or +1, one can consider the neurons to be OFF or ON. By reasonably close it is meant that either $V_i < -1 + \nu$ or, $V_i > 1 - \nu$, ν being a small positive quantity. In our simulation study we used $\nu = 0.1$.

For simulation of the network the input bias was taken (as in the previous chapter) to be linearly proportional to the gray level of the corresponding pixels mapped in the range [-1, 1]. Thus, i being the gray value of a pixel, L the maximum available gray level, the input bias I_i is taken as

$$I_i = 2 \frac{i}{L} - 1. \quad (6.31)$$

The input/output transfer function used was a second order polynomial (equa-

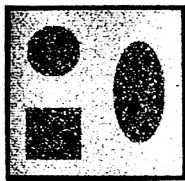
tion 5.10 with $n = 2$). The connection strengths were assigned as

$$W_{ij} = \begin{cases} 1 & \text{if } j \in N_i^2 \\ 0 & \text{otherwise.} \end{cases} \quad (6.32)$$

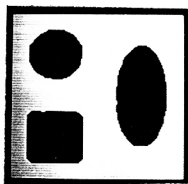
The results obtained by the proposed technique are shown in Figures 6.3-6.7. In Figures 6.3-6.5, the results shown are on synthetic images. From the results it is evident that with lower values of σ (high *SNR*) the objects extracted are very close to the actual ones; whereas with higher values of σ (low *SNR*) there is a slight deterioration in the results. However, even for low *SNR* the key features of the original image are preserved. From Figures 6.6 and 6.7 it is seen that for the real images the extracted object are compact and preserved their approximate outline. This shows the noise immunity of the proposed technique. (For the *Noisy tank* image a rough estimate of the noise level σ was made by considering the middle point of the gray level range as the threshold and then identifying the pixels of the object and the background regions for computing their mean and σ values.)

Figures 6.8-6.10 demonstrate some results obtained by the methods described in the previous chapter in order to provide a comparative study with the proposed technique for noisy images. It is seen that for high *SNR* values the difference in output is not significant, whereas for low *SNR* values the present model provides superior performance (see Figures 6.3-6.5 and 6.8-6.10). It may be mentioned here that another experiment for providing an objective comparison among these techniques has been described in Chapter 8.

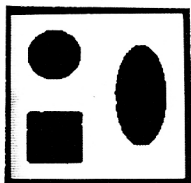
Like the methods described in the previous chapter, the present techniques are also superior to the probabilistic relaxation [2, 68] method in the sense of preserving the shapes and outlines of the objects. As expected, the output of the continuous model (though it takes more time) is always better than that of the corresponding discrete model. However, when the noise level is low, the segments produced by the continuous model is almost identical to that of the discrete model. Thus for high *SNR* the discrete model may be preferred.



(a)



(b)



(c)

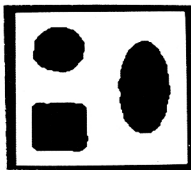
Figure 6.3: A noisy version of the synthetic image with $SNR=2.4$: a) input, b) extracted object with discrete model, c) extracted object with continuous model



(a)

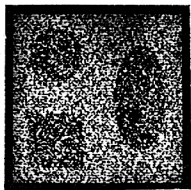


(b)

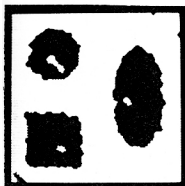


(c)

Figure 6.4: A noisy version of the synthetic image with $SNR=1.2$: a) input, b) extracted object with discrete model, c) extracted object with continuous model



(a)



(b)

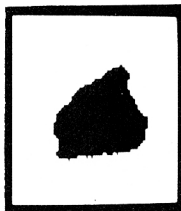


(c)

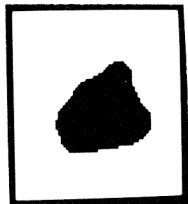
Figure 6.5: A noisy version of the synthetic image with $SNR=0.75$: a) input, b) extracted object with discrete model, c) extracted object with continuous model

see Figure 4.4(a)

(a)



(b)



(c)

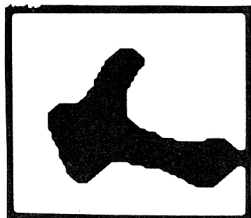
Figure 6.6: *Noisy tank* image : a) input, b) extracted object with discrete model, c) extracted object with continuous model

see Figure 3.7(c)

(a)



(b)

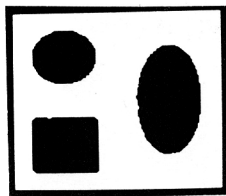


(c)

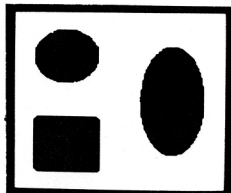
Figure 6.7: Noisy *Biplane* image ($\sigma = 5$) : a) input, b) extracted object with discrete model, c) extracted object with continuous model

see Figure 6.3(a)

(a)



(b)

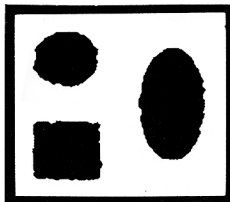


(c)

Figure 6.8: A noisy version of the synthetic image with $SNR=2.4$: a) input, b) extracted object with the discrete model described in Chapter 5, c) extracted object with the continuous model described in Chapter 5

see Figure 6.4(a)

(a)



(b)



(c)

Figure 6.9: A noisy version of the synthetic image with $SNR=1.2$: a) input, b) extracted object with the discrete model described in Chapter 5, c) extracted object with the continuous model described in Chapter 5

see Figure 6.5(a)

(a)



(b)



(c)

Figure 6.10: A noisy version of the synthetic image with $SNR=0.75$: a) input, b) extracted object with the discrete model described in Chapter 5, c) extracted object with the continuous model described in Chapter 5

6.6 Conclusions and discussion

The present work demonstrate a method for solving a computationally hard object extraction problem based on the *GRF* model with a modified Hopfield neural network architecture. The input/output transfer function used is a polynomial over a fixed domain (same as in the previous chapter). The energy function of the network has been designed in such a way that its minimum value corresponds to the *MAP* estimate of the scene. A possible hardware realization of a neuron is also suggested.

The results obtained by the proposed model are quite satisfactory, even with very high degree of noise. Noise immunity of these methods is seen to be more than those described in Chapters 4 and 5. The results obtained by continuous dynamics are always better than those of discrete dynamics. But the convergence of the later is very fast (as in the previous chapter). The present system takes more time to convergence than that described in Chapter 5.

Chapter 7

Multi-layer neural network with
fuzziness measures for object
extraction

7.1 Introduction

We have described various methodologies for image segmentation in Chapters 2-6. Chapters 2 and 3 dealt with some aspects of fuzzy set theory (namely, image correlation measures and fuzzy geometry), whereas Chapters 4-6 described the utility of some artificial neural network models (e.g., Kohonen's model and Hopfield's network) for image segmentation/object extraction problems. In the present chapter an attempt is made to integrate the concept of fuzzy sets and neural networks by incorporating various fuzziness measures in a multi-layer network.

Since the conventional feed forward multi-layer perceptron [134] (for details see Appendix A) with back propagation of error requires a set of labeled input patterns (training samples), as such it can not be used for image segmentation, particularly when only one image is available. (However, if images to be processed are of similar nature, one can use a set of known images for learning and then use the network for processing of other images.) Therefore the present work not only describes a method of integrating the concept of fuzzy sets and neural networks, but also demonstrates a way of using a multi-layer neural network for performing (unsupervised) self-organizing tasks, in general, and object extraction in particular.

A *self-organizing multi-layer neural network* architecture suitable for image processing operations is designed. The architecture is also a feed forward one with back propagation of error; but unlike conventional multi-layer perceptron (*MLP*) it does not require any supervised learning. The learning technique employed is *self-supervised* thereby attaining self-organizing capability. Each neuron is connected to the corresponding neuron in the previous layer and its neighbors. Another structural difference from the conventional *MLP* is that there exists a feed-back path from the output to the input layer. The status of neurons in the output layer is described as a *fuzzy set*. A fuzziness measure of this set is used as a measure of error of the system. Learning rates for various measures of fuzziness (e.g., index of fuzziness and entropy) have been theoretically and experimentally studied. Given an input, the system automatically

finds out the structure in the input data by *self-supervision/self-organization*.

The network has been employed to extract objects from noisy environments. The simulation study was done using synthetic and real images. The results are seen to be comparable to those of the methods described in Chapters 4-6.

7.2 Measures of fuzziness of a fuzzy set

Various measures of fuzziness of a fuzzy set A with n supports are defined below.

Linear index of fuzziness [90]

$$\nu_l(A) = \frac{2}{n} \sum_{i=1}^n [\min \{ \mu_A(x_i), (1 - \mu_A(x_i)) \}]. \quad (7.1)$$

Quadratic index of fuzziness [90]

$$\nu_q(A) = \frac{2}{\sqrt{n}} \sqrt{ \left[\sum_{i=1}^n \{ \mu_A(x_i) - \mu_{\bar{A}}(x_i) \}^2 \right] }. \quad (7.2)$$

Entropy [99]

$$H(A) = \frac{1}{n \ln(2)} \sum_{i=1}^n S_n \{ \mu_A(x_i) \} \quad (7.3)$$

with

$$S_n(\mu_A(x_i)) = -\mu_A(x_i) \ln \{ \mu_A(x_i) \} - \{ 1 - \mu_A(x_i) \} \ln \{ 1 - \mu_A(x_i) \}.$$

Entropy [97]

$$H(A) = \frac{1}{n(\sqrt{e} - 1)} \sum_{i=1}^n \{ S_n(\mu_A(x_i)) - 1 \} \quad (7.4)$$

with

$$S_n(\mu_A(x_i)) = \mu_A(x_i) e^{1-\mu_A(x_i)} + \{ 1 - \mu_A(x_i) \} e^{\mu_A(x_i)}.$$

These measures reflect an average amount of ambiguity in making a decision whether an element belongs to the fuzzy set (or possesses the concept or property represented by the fuzzy set) or not. Details on these measures are given in Appendix A.2.3.

7.3 A self-organizing multi-layer neural network

In this section we will be describing a new self-organizing multi-layer neural network model.

7.3.1 Description and operation of the network

Architecture :

In Figure 7.1 we depict a 3-layered version of the proposed network architecture. In each layer there are $M \times N$ neurons (for an $M \times N$ image). Each neuron corresponds to a single pixel. Besides the *input* and *output* layers, there can be a number of *hidden* layers (more than or equal to one). Neurons in the same layer do not have any connection among themselves. Each neuron in a layer is connected to the corresponding neuron in the previous layer and its neighbors over N^d , the d th ordered neighborhood (described in Chapter 5, pp. 127); thus each neuron in layer i ($i > 1$) will have $|N^d| + 1$ (where $|N^d|$ is the number of pixels in N^d) links to the $(i - 1)$ th layer. For N^1 a neuron has 5-links, whereas for N^2 9-links will be associated with every neuron. However, for boundary nodes (pixels) number of links may be less than $|N^d| + 1$. Every neuron in the output layer is also connected to the corresponding neuron in the input layer (feedback path). It may be noted that this architecture differs from the conventional *MLP* in two main points :

- the distribution of links, and

- the feed back connection from the output layer to the input layer.

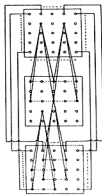


Figure 7.1: Schematic representation of a self-organizing multi-layer neural network

Initialization :

The input to a neuron in the input layer is given as a real number in $[0, 1]$ which is proportional to the gray value of the corresponding pixel. All weights are initialized to one (1). No external bias is imposed on the weights. Random initialization (of weights) may act as pseudo noise and the quality of the processed image may be deteriorated. As all the weights are set to unity, the total input (initially) to any node lies in $[0, n_i]$ (where n_i is the number of links a neuron has); hence the most unbiased choice for the threshold value θ for the input-output transfer function (see Appendix A for details)

$$g(x) = \frac{1}{1 + e^{-(x-\theta)}} \quad (7.5)$$

would be $n_i/2$ (the middle most value of the total initial input range).

Operation :

The input value (U_i) to any neuron in the i th layer [except the input layer] is calculated as

$$U_i = \sum_j W_{ij} V_j. \quad (7.6)$$

Here V_j is the output of the j th unit of the previous layer and W_{ij} is the connection weight between the i th node of one layer and the j th node of the previous layer.

The input-output transfer function g (equation 7.5) is then applied to get the output status of these neurons. Thus the output of a node i is

$$V_i = g(U_i). \quad (7.7)$$

These outputs are then fed as input to the next layer. In this way the input pattern is passed on from the input layer to the output layer and the corresponding output states are calculated. The output value of each neuron lies in $[0, 1]$.

The updating procedure of the network is such that in the stable state the output status of all the neurons in the output layer will be either 0 or 1 representing black (background) and white (object). But in the intermediate stage the output status of the neurons in the output layer usually will be in $[0, 1]$, and thus the status value may be considered to represent the degree of brightness (darkness) of the corresponding pixel in the image or, the degree of belonging of the pixel in the object (background) region. Therefore, the status of the neurons in the output layer may be viewed to represent a fuzzy set *object region*. The number of supports of this fuzzy set is equal to the number of neurons in the output layer. A measure of *fuzziness of this set*, on the global level, may be considered as the *error or instability of the whole system*, as this will reflect the deviation from the desired state of the network. Thus when we do not have any apriori target output value, we can take the fuzziness value as a measure of system error and *back-propagate it to adjust the weights* (mathematical expressions for this are given later) so that the system error reduces with passage of time and in the limiting case it becomes zero. The error measure (E) can

also be taken as a suitable function of a fuzziness measure, i.e.,

$$E = g(I) \quad (7.8)$$

where I is a *measure of fuzziness* (equations 7.1, 7.2, 7.3 and 7.4) of the fuzzy set.

The change in weights is calculated with the same method used in the conventional *MLP* (described in details in Appendix A). The expression is

$$\Delta W_{kj} = \begin{cases} -\eta \left(\frac{\partial E}{\partial V_k} \right) g'(U_k) V_j \\ -\eta \left(\sum_m \frac{\partial E}{\partial U_m} W_{mk} \right) g'(U_k) V_j \end{cases} \quad (7.9)$$

for the output layer and other layers respectively.

After the weights have been adjusted once by back-propagating the error, the output of the neurons in the output layer is fed back to the corresponding neurons in the input layer. The second pass is then continued with this as input. The error in the second pass is then back-propagated for correcting the weights for the second time. This procedure is continued until the network stabilizes, i.e., the error value (measure of fuzziness) becomes zero. When the network stabilizes the status of the neurons in the output layer becomes either 0 or 1. Neurons with output value 0 constitute one group (region) and those having output value 1 constitute the other group.

Note that, the system actually does some sort of *self-supervised* learning, and therefore *self-organizes* and finds out the structure in the input data. Thus for problems like clustering where there is no concept of apriori teaching such a network may be of great importance. In self-supervised learning the system learns to respond to *interesting* patterns in the input. In general, such a scheme should be able to form the basis for the development of feature detectors and should discover statistically salient features of the input population.

7.3.2 Weight correction for different measures of fuzziness

The mathematical derivation for weight updating rules using back propagation with different fuzziness measures (equations 7.1, 7.2, 7.3 and 7.4) is as follows. The derivations are given only for correcting the weights of the links connected to the output layer. For other layers similar expressions as in the second part of equation (7.9) are applicable.

Weight correction for linear index of fuzziness :

Let us consider

$$\begin{aligned} E &= g(\nu_l) \\ &= \nu_l \end{aligned} \quad (7.10)$$

where ν_l is the linear index of fuzziness (equation 7.1).

Here

$$-\frac{\partial E}{\partial V_j} = \begin{cases} -\frac{2}{n} & \text{if } 0 \leq V_j \leq 0.5 \\ \frac{2}{n} & \text{if } 0.5 \leq V_j \leq 1. \end{cases} \quad (7.11)$$

Thus from equation (7.9) we get

$$\Delta W_{ji} = \begin{cases} \eta_1(-\frac{2}{n})f'(U_j)V_i & \text{if } 0 \leq V_j \leq 0.5 \\ \eta_1(\frac{2}{n})f'(U_j)V_i & \text{if } 0.5 \leq V_j \leq 1 \end{cases} \quad (7.12)$$

or,

$$\Delta W_{ji} = \begin{cases} -\eta f'(U_j)V_i & \text{if } 0 \leq V_j \leq 0.5 \\ \eta f'(U_j)V_i & \text{if } 0.5 \leq V_j \leq 1 \end{cases} \quad (7.13)$$

where $\eta = \eta_1 \times \frac{2}{n}$.

Weight correction for quadratic index of fuzziness :

Choose the error function E as

$$\begin{aligned} E &= g(\nu_q) \\ &= \nu_q^2 \end{aligned} \quad (7.14)$$

where ν_q is the quadratic index of fuzziness (equation 7.2).

Now,

$$\nu_q^2 = \frac{4}{n} [\sum_j \{\min(V_j, 1 - V_j)\}^2] \quad (7.15)$$

and

$$-\frac{\partial E}{\partial V_j} = \begin{cases} \frac{4}{n} \{-2V_j\} & \text{if } 0 \leq V_j \leq 0.5 \\ \frac{4}{n} \{2(1 - V_j)\} & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases} \quad (7.16)$$

Thus

$$\Delta W_{ji} = \begin{cases} \eta_2 \frac{4}{n} (-2V_j) f'(U_j) V_i & \text{if } 0 \leq V_j \leq 0.5 \\ \eta_2 \frac{4}{n} \{2(1 - V_j)\} f'(U_j) V_i & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases} \quad (7.17)$$

In other words,

$$\Delta W_{ji} = \begin{cases} \eta (-V_j) f'(U_j) V_i & \text{if } 0 \leq V_j \leq 0.5 \\ \eta \{(1 - V_j)\} f'(U_j) V_i & \text{if } 0.5 \leq V_j \leq 1.0 \end{cases} \quad (7.18)$$

where $\eta = \eta_2 \times \frac{4}{n} \times 2$.

If we define the target output $\{t_j\}$ as follows

$$t_j = \begin{cases} 0 & \text{if } 0 \leq V_j \leq 0.5 \\ 1 & \text{if } 0.5 \leq V_j \leq 1.0, \end{cases} \quad (7.19)$$

then this fuzziness measure becomes equivalent to the sum of the squared error.

It may be noted here that $\frac{\partial E}{\partial V_j}$ is not defined when $V_j = 0.5$ for both ν_l and ν_q . Thus, when implementing the algorithm, if V_j takes a value of 0.5 for any node, then movement can be made in any direction by a small amount.

Weight correction for logarithmic entropy :

Now we consider

$$E = g(H) \quad (7.20)$$

$$= H \quad (7.21)$$

where H is the entropy of a fuzzy set (equation 7.3).

Thus

$$-\frac{\partial E}{\partial V_j} = \frac{1}{n \ln(2)} \ln \frac{V_j}{1 - V_j}. \quad (7.22)$$

Here we notice that as $V_j \rightarrow 0$ or 1 , $\text{abs}\{\ln \frac{V_j}{1-V_j}\} \rightarrow \infty$, and as $V_j \rightarrow 0.5$, $\text{abs}\{\ln \frac{V_j}{1-V_j}\} \rightarrow 0$.

For a fuzzy set we know, the fuzziness value is minimum when the membership values of all the elements are 0 or 1, and maximum when they all are 0.5. So for the network the error is minimum (i.e., the network has stabilized) when all the output values are either 0 or 1, and is maximum when they all are 0.5 (the network is most unstable). From the previous discussion (equation 7.22) we notice that $\text{abs}(\frac{\partial E}{\partial V_j})$ is minimum ($=0$) when all the output values are 0.5. Thus, if we use the gradient descent search, the rate of learning is minimum at the most unstable state. Hence, to expedite the learning, it is desirable to make the weight correction large when the network is most unstable (i.e., when all the output values are 0.5). In other words, for a neuron, the weight correction for its links should be maximum when its output status is very close to 0.5 and is minimum when its output status is close to 0 or 1. This can be achieved by taking

$$\Delta W_{ji} \propto -\frac{\frac{\partial E}{\partial V_j}}{|\frac{\partial E}{\partial V_j}|^q}, \quad q > 1 \quad (7.23)$$

i.e.,

$$\Delta W_{ji} = -\eta_3 \frac{\frac{\partial E}{\partial V_j}}{|\frac{\partial E}{\partial V_j}|^q} f'(U_j) V_i, \quad q > 1 \quad (7.24)$$

where $|\frac{\partial E}{\partial V_j}|$ represents the magnitude of the gradient.

Such a choice for ΔW_{ij} does not violate the necessary requirements for gradient descent search. In any gradient descent search, the gradient vector determines the direction of movement, while the quantum of movement is controlled by a scalar multiplier which is generally pre-selected by the user. The magnitude of the gradient vector is not very important as far as the gradient descent search

is concerned. In the present case, since we are changing only the magnitude of movement, keeping the direction same, it is also equivalent to a scalar multiplication. However, if the scalar multiplier is too small, searching would be slow; but if it is too large, it may result in oscillation.

Henceforth we will be considering only $q=2$. When $q=2$, for the logarithmic entropy we get

$$\Delta W_{ji} = -\eta_3 \frac{1}{\frac{\partial E}{\partial V_j}} f'(U_j) V_i \quad (7.25)$$

$$= \eta_3 (n \ln 2) \frac{1}{\ln \frac{V_i}{1-V_j}} f'(U_j) V_i \quad [\text{from equation (7.22)}]$$

$$= \eta \frac{1}{\ln \frac{V_i}{1-V_j}} f'(U_j) V_i \quad (7.26)$$

with $\eta = \eta_3 \times (n \ln 2)$,

i.e.,

$$\Delta W_{ji} = \begin{cases} -\eta \frac{1}{\ln \frac{V_i}{1-V_j}} f'(U_j) V_i & \text{if } 0 \leq V_j \leq 0.5 \\ \eta \frac{1}{\ln \frac{V_i}{1-V_j}} f'(U_j) V_i & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases} \quad (7.27)$$

The expression of ΔW_{ji} has been divided into two parts in order to keep analogy with that of the index of fuzziness.

Weight correction for exponential entropy :

If we consider

$$E = g(H) \quad (7.28)$$

$$= H \quad (7.29)$$

where H is the exponential entropy of a fuzzy set (equation 7.4).

Then

$$\frac{\partial E}{\partial V_j} = \frac{1}{n(\sqrt{e}-1)} \{(1-V_j)e^{1-V_j} - V_j e^{V_j}\}. \quad (7.30)$$

Applying the similar argument as in the case of logarithmic entropy, we get

$$\begin{aligned}\Delta W_{ji} &= -\eta_4 \frac{1}{\frac{\partial E}{\partial V_j}} f'(U_j) V_i \\ &= -\eta \frac{1}{(1-V_j)e^{1-V_j} - V_j e^{V_j}} f'(U_j) V_i\end{aligned}\quad (7.31)$$

with $\eta = \eta_4 \times n(\sqrt{e} - 1)$.

In other words,

$$\Delta W_{ji} = \begin{cases} -\eta \frac{1}{(1-V_j)e^{1-V_j} - V_j e^{V_j}} f'(U_j) V_i & \text{if } 0 \leq V_j \leq 0.5 \\ \eta \frac{1}{V_j e^{V_j} - (1-V_j)e^{1-V_j}} f'(U_j) V_i & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases}\quad (7.32)$$

Considering equations (7.13), (7.18), (7.27) and (7.32), one can see that in each of the four cases, the expression for ΔW_{ji} has a common factor $\eta f'(U_j) V_i$. Therefore, for the sake of comparison of different learning rates the common factor $[\eta f'(U_j) V_i]$ can be ignored. In the next section we shall investigate the rate of learning under the previously mentioned scheme.

7.4 Learning rate for different measures of fuzziness

Ignoring the common factor $\eta f'(U_j) V_i$ in equations (7.13), (7.18), (7.27) and (7.32), the learning rate for different error measures can be written as

$$\begin{aligned}\Delta W_{ji}^l &\propto -1 & \text{for } 0 \leq V_j \leq 0.5 \\ &\propto 1 & \text{for } 0.5 \leq V_j \leq 1.0\end{aligned}\quad (7.33)$$

for *linear index of fuzziness*;

$$\begin{aligned}\Delta W_{ji}^q &\propto -V_j & \text{for } 0 \leq V_j \leq 0.5 \\ &\propto 1 - V_j & \text{for } 0.5 \leq V_j \leq 1.0\end{aligned}\quad (7.34)$$

for *quadratic index of fuzziness*;

$$\begin{aligned}\Delta W_{ji}^{le} &\propto -\frac{1}{\ln \frac{1-V_j}{V_j}} & \text{for } 0 \leq V_j \leq 0.5 \\ &\propto \frac{1}{\ln \frac{V_j}{1-V_j}} & \text{for } 0.5 \leq V_j \leq 1.0\end{aligned}\quad (7.35)$$

for the *logarithmic entropy* ($q=2$); and

$$\begin{aligned}\Delta W_{ji}^{ee} &\propto -\frac{1}{(1-V_j)e^{1-V_j}-V_j e^{V_j}} & \text{for } 0 \leq V_j \leq 0.5 \\ &\propto \frac{1}{V_j e^{V_j}-(1-V_j)e^{1-V_j}} & \text{for } 0.5 \leq V_j \leq 1.0\end{aligned}\quad (7.36)$$

for the *exponential entropy* ($q=2$).

A critical examination of equations (7.33)-(7.36) reveals that in each of the four cases the learning rate is ≤ 0 for $V_j \leq 0.5$ and is ≥ 0 for $V_j \geq 0.5$. In other words, the direction of change is always the same in all the four cases. Therefore, if one compares different learning rates separately for $V_j \in [0, 0.5]$ and $V_j \in [0.5, 1.0]$, one can forget about the sign (direction) of learning. For example, if $\Delta w_{ji}^{le} = -5$ and $\Delta w_{ji}^{ee} = -3$, then although $\Delta w_{ji}^{le} \leq \Delta w_{ji}^{ee}$, the rate of learning is more for Δw_{ji}^{le} .

Figure 7.2 depicts the rate of learning for different error measures with variation of status value of an output neuron. Because of the reasons mentioned in the previous paragraph, only the magnitude of ΔW_{ji} has been plotted. From the curves it is noticed that for all the error measures, the rate of learning is low when the neurons are very close to stable states (i.e., status value 0 or 1) and is high when the neurons are highly unstable (i.e., status value 0.5). The curves are also symmetric around 0.5.

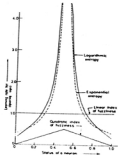


Figure 7.2: Rate of learning with variation of status for different error measures

A comparative study of the different curves shows that the learning rate is minimum (for all status values) for the quadratic index of fuzziness and its

absolute value is ≤ 0.5 . For linear index of fuzziness it is constant and is $= 1$. On the other hand, for entropy measures the rates of learning are high. For status values very close to 0.5, the learning rate is more for the logarithmic entropy than that of the exponential entropy, whereas for output values close to 0 or 1 the learning rate is less for the logarithmic entropy. In other words, the change in rate of learning for the exponential entropy is less than that of the logarithmic entropy. So the outputs corresponding to exponential entropy will be more noise immune than that of the logarithmic entropy, but it will take more time to converge. Regarding learning rate the following proposition can be made.

Proposition : *The learning rate for the quadratic index of fuzziness is the minimum.*

Proof : To prove the proposition it suffices to prove

1. $\Delta W_{ji}^q < \Delta W_{ji}^l$
2. $\Delta W_{ji}^q < \Delta W_{ji}^{le}$
3. $\Delta W_{ji}^q < \Delta W_{ji}^{ee}$.

Proof of condition 1 (i.e., $\Delta W_{ji}^q < \Delta W_{ji}^l$).

Leaving aside the signs

$$\Delta W_{ji}^q = \eta \kappa \text{ with}$$

$$\kappa = \begin{cases} V_j & \text{if } 0 \leq V_j \leq 0.5 \\ 1 - V_j & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases}$$

$$\text{But } \Delta W_{ji}^l = \eta \times 1.0 .$$

$$\text{Evidently } \Delta W_{ji}^q < \Delta W_{ji}^l.$$

Proof of condition 2 (i.e., $\Delta W_{ji}^q < \Delta W_{ji}^{le}$).

Leaving aside the signs

$$\Delta W_{ji}^{le} = \begin{cases} \eta \frac{1}{\ln \frac{1-V_j}{V_j}} & \text{if } 0 \leq V_j \leq 0.5 \\ \eta \frac{1}{\ln \frac{V_j}{1-V_j}} & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases}$$

Proving the relation $\Delta W_{ji}^q < \Delta W_{ji}^{le}$ is equivalent to prove

$$\frac{1}{\ln \frac{1-V_j}{V_j}} > V_j \quad \text{for } 0 \leq V_j \leq 0.5, \quad \text{and}$$

$$\frac{1}{\ln \frac{V_j}{1-V_j}} > 1 - V_j \quad \text{for } 0.5 \leq V_j \leq 1.0.$$

We have to prove

$$\frac{1}{\ln \frac{1-V_j}{V_j}} > V_j \quad [\text{for } 0 \leq V_j \leq 0.5]$$

$$\text{or,} \quad \ln \frac{1-V_j}{V_j} < \frac{1}{V_j}$$

$$\text{or,} \quad \ln \left(\frac{1}{V_j} - 1 \right) < \frac{1}{V_j}$$

$$\text{or,} \quad \ln (x - 1) < x \quad [\text{with } x = \frac{1}{V_j}].$$

Suppose

$$y = \ln (x - 1).$$

Then

$$x = 1 + e^y$$

$$= 1 + 1 + \frac{y}{1} + \frac{y^2}{2!} + \cdots \text{to } \infty$$

$$> y \quad [\text{since } x \geq 2 \text{ and thereby } y \geq 0]$$

Hence, $\ln (x - 1) < x$, thus proving the first part.

Let us now prove the other part

$$\frac{1}{\ln \frac{V_j}{1-V_j}} > 1 - V_j \quad [\text{for } 0.5 \leq V_j \leq 1.0]$$

$$\text{or,} \quad \ln \frac{V_j}{1-V_j} < \frac{1}{1-V_j}$$

$$\text{or,} \quad \ln (x - 1) < x \quad [\text{with } x = \frac{1}{1-V_j}].$$

Rest part of the proof is as in the previous case and hence the relation

$$\frac{1}{\ln \frac{1-V_j}{V_j}} > V_j \quad \text{for } 0 \leq V_j \leq 0.5, \quad \text{and}$$

$$\frac{1}{\ln \frac{V_j}{1-V_j}} > 1 - V_j \quad \text{for } 0.5 \leq V_j \leq 1.0.$$

Proof of condition 3 (i.e., $\Delta W_{ji}^q < \Delta W_{ji}^{ee}$).

Leaving aside the signs

$$\Delta W_{ji}^{ee} = \begin{cases} \eta \frac{1}{(1-V_j)e^{1-V_j} - V_j e^{V_j}} & \text{if } 0 \leq V_j \leq 0.5 \\ \eta \frac{1}{V_j e^{V_j} - (1-V_j)e^{1-V_j}} & \text{if } 0.5 \leq V_j \leq 1.0. \end{cases}$$

To prove the relation $\Delta W_{ji}^q < \Delta W_{ji}^{ee}$, is equivalent to prove

$$\begin{aligned}\frac{1}{(1-V_j)e^{1-V_j}-V_j e^{V_j}} &> V_j \quad \text{for } 0 \leq V_j \leq 0.5, \quad \text{and} \\ \frac{1}{V_j e^{V_j}-(1-V_j)e^{1-V_j}} &> 1 - V_j \quad \text{for } 0.5 \leq V_j \leq 1.0.\end{aligned}$$

Now to prove (for notational simplicity x will be used in place of V_j)

$$\begin{aligned}\frac{1}{(1-x)e^{1-x}-xe^x} &> x \quad \text{for } 0 \leq x \leq 0.5 \\ \text{or, } (1-x)e^{1-x} - xe^x &< \frac{1}{x} \\ \text{or, } x(1-x)e^{1-x} - x^2e^x &< 1 \\ \text{or, } xe^{1-x} - x^2(e^{1-x} + e^x) &< 1 \\ \text{or, } xe^{1-x} &< 1 + x^2(e^{1-x} + e^x).\end{aligned}$$

Let $f(x) = xe^{1-x}$.

Then $f'(x) = (1-x)e^{1-x} > 0$, $x \in [0, 0.5]$. Hence xe^{1-x} monotonically increases for $x \in [0, 0.5]$ and thus xe^{1-x} attains the maximum value at $x = 0.5$.

Hence, $\max\{xe^{1-x}\} = \frac{1}{2}\sqrt{e} < 1$, $x \in [0, 0.5]$.

On the other hand, $\min\{1 + x^2(e^{1-x} + e^x)\} = 1$.

Hence the relation.

Let us now prove

$$\begin{aligned}\frac{1}{V_j e^{V_j}-(1-V_j)e^{1-V_j}} &> 1 - V_j \quad \text{for } 0.5 \leq V_j \leq 1.0 \\ \text{or, } V_j e^{V_j} - (1 - V_j)e^{1-V_j} &< \frac{1}{1-V_j} \\ \text{or, } (1-x)e^{1-x} - xe^x &< \frac{1}{x} \quad \text{with } x = 1 - V_j, x \in [0, 0.5].\end{aligned}$$

Rest part of the proof is as in the previous case and thus we can write

$$\begin{aligned}\frac{1}{(1-V_j)e^{1-V_j}-V_j e^{V_j}} &> V_j \quad \text{for } 0 \leq V_j \leq 0.5 \quad \text{and} \\ \frac{1}{V_j e^{V_j}-(1-V_j)e^{1-V_j}} &> 1 - V_j \quad \text{for } 0.5 \leq V_j \leq 1.0.\end{aligned}$$

Hence the proposition. ♣

7.5 Application to object extraction

An application of the proposed network architecture is shown in object extraction (specially from noisy environments) problems. A 3-layered version of

the proposed network with second order neighborhood system (N^2) is used. A neuron thus gets input from 9 neurons of the previous layer. The threshold value θ in this case is $9/2=4.5$. The input gray levels are mapped in $[0, 1]$ by a linear transformation and is given as input to the network. The network is then allowed to be settled. When the network is stabilized, the neurons having the status values 0 constitute one region type say, background (object) and the remaining neurons with output status 1 constitute another region type, say object (background). Investigation has also been done with N^1 .

7.5.1 Simulation and results

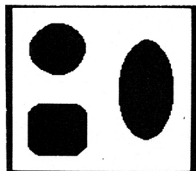
To check the validity and effectiveness of the proposed technique, a computer simulation has been done using synthetic and real images. The original synthetic image [Figure 5.4, pp. 135] had only two levels which are then corrupted by additive $N \sim (0, \sigma^2)$ noise with different σ (10, 20, 32) values (i.e., different *SNR* values as defined in Chapter 5, pp. 135), as in the previous chapter [Figures 6.3(a), 6.4(a) and 6.5(a)]. The real images used are noise corrupted versions of a tank (*Noisy tank*, Figure 4.4(a)) and a *Biplane* [Figure 3.7(c)]. For the simulation study η value was taken as 0.2.

The objects extracted by the proposed method with different error measures for different noisy versions of the synthetic and the noisy *Biplane* images are included in Figures 7.3-7.6. Figure 7.7 depicts the objects extracted from the *Noisy tank* image with different error models. As a typical illustration, the curves reflecting the average system error for different error measures with number of iterations (time) for a noisy synthetic image (with $\sigma=20$) are depicted in Figure 7.8.

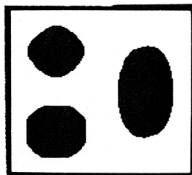
From the results of the synthetic images it is seen that as the noise level increases, the quality of the output, as expected, deteriorates; but approximate shapes and outlines of the objects are preserved. Comparing the results of different error models, it is noticed that the outputs with index of fuzziness measures (linear/quadratic) are better than those obtained by entropy measures ($q=2$). For a fixed value of η , the learning rate is low for the indices of

see Figure 6.3(a)

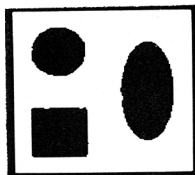
(a)



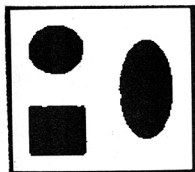
(b)



(c)



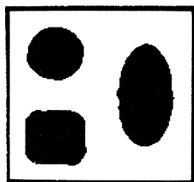
(d)



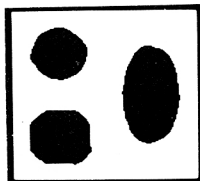
(e)

Figure 7.3: A noisy version of the synthetic image with $SNR=2.4$: a) input, b) extracted object with linear index of fuzziness, c) extracted object with quadratic index of fuzziness, d) extracted object with logarithmic entropy, e) extracted object with exponential entropy

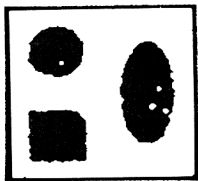
(a)



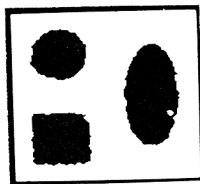
(b)



(c)



(d)

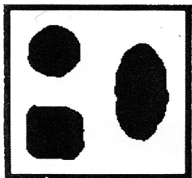


(e)

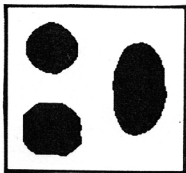
Figure 7.4: A noisy version of the synthetic image with $SNR=1.2$: a) input, b) extracted object with linear index of fuzziness, c) extracted object with quadratic index of fuzziness, d) extracted object with logarithmic entropy, e) extracted object with exponential entropy

see Figure 6.5(a)

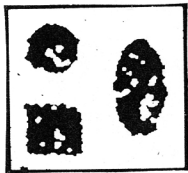
(a)



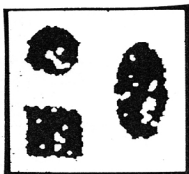
(b)



(c)



(d)

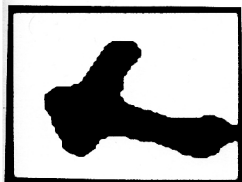


(e)

Figure 7.5: A noisy version of the synthetic image with $SNR=0.75$: a) input, b) extracted object with linear index of fuzziness, c) extracted object with quadratic index of fuzziness, d) extracted object with logarithmic entropy, e) extracted object with exponential entropy

see Figure 3.7(c)

(a)



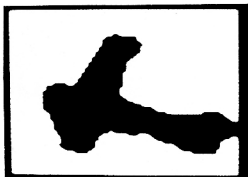
(b)



(c)



(d)

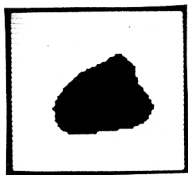


(e)

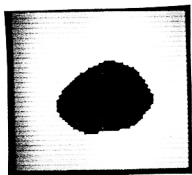
Figure 7.6: Noisy *Biplane* image : a) input, b) extracted object with linear index of fuzziness, c) extracted object with quadratic index of fuzziness, d) extracted object with logarithmic entropy, e) extracted object with exponential entropy

see Figure 4.4(a)

(a)



(b)



(c)



(d)



(e)

Figure 7.7: *Noisy tank* image : a) input, b) extracted object with linear index of fuzziness, c) extracted object with quadratic index of fuzziness, d) extracted object with logarithmic entropy, e) extracted object with exponential entropy

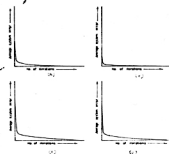


Figure 7.8: Variation of average system error with time for the image in figure 6.4(a) : a) linear index of fuzziness, b) quadratic index of fuzziness, c) logarithmic entropy, d) exponential entropy

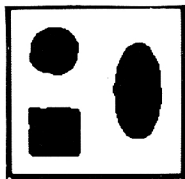
fuzziness, whereas it is high for entropy measures. When the learning rate is high, a particular neuron influences its neighbors to a great extent; thus the noisy elements affect the results strongly. The system thus fails to remove all the noise. Of the two entropy measures the one with exponential function is more noise immune due to its lower learning rate at the initial stage.

A critical examination of the results reveals that the index of fuzziness (both linear and quadratic) is consistently better than entropy measures for maintaining the compactness of the extracted objects. But shapes of objects are better preserved by entropy measures. This observation can be explained as follows. Since the rate of learning is slow for index of fuzziness, it smoothes out noises and creates compact regions, while the rapid learning ability of the entropy measures may not be able to remove all the noisy pixels, particularly when the SNR is very low (i.e., very high noise). On the other hand, entropy measures enable the network to preserve the object boundaries as their learning rates are very high around the most ambiguous region ($V_j \simeq 0.5$).

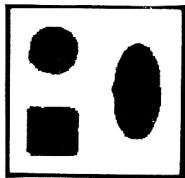
An investigation was also carried out with first order (N^1) neighborhood system and, as expected, the results are not as good (like those in Chapters 5 and 6) as with the N^2 system. As an illustration, the outputs (for different error measures) corresponding to the input image [Figure 6.4(a)] are shown in Figure 7.9.

see Figure 6.4(a)

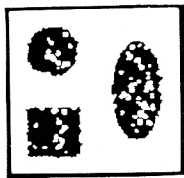
(a)



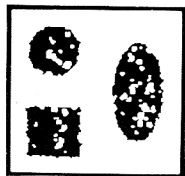
(b)



(c)



(d)



(e)

Figure 7.9: A noisy version of the synthetic image with $SNR=1.2$ and N^1 : a) input, b) extracted object with linear index of fuzziness, c) extracted object with quadratic index of fuzziness, d) extracted object with logarithmic entropy, e) extracted object with exponential entropy

Figure 7.8 depicts the variation of average system error (ASE), drawn to the scale, with time (number of iterations). A comparative study of the curves reflects the following : the initial ASE s for the entropy measures are much higher than those for the indices of fuzziness. In fact, for the quadratic index of fuzziness the initial ASE is the minimum. For all cases the ASE drops to a very low value (close to zero) within a few iterations which indicates a fast convergence of the system. Note that in each of the four cases the ASE drops to almost the same low value in more or less the same number of iterations, but the initial ASE s for the entropy measures are more higher than those for the linear and quadratic indices of fuzziness. Hence the rate of change of the ASE for any entropy measure is much higher than that for either of the indices the fuzziness. Again, of the two indices of fuzziness, the linear one has much higher initial value of the ASE , but both of them drop almost to the same low value in more or less the same number of iterations. This indicates that the rate of change of ASE for the linear index of fuzziness is greater than that for the quadratic index of fuzziness. These observations conform to the proposition stated and proved in section 7.4.

7.6 Conclusions and discussion

A *self-organizing multi-layer neural network* suitable for object extraction from noisy images has been developed by incorporating various fuzziness measures in a layered network framework. Each neuron in the network corresponds to an image pixel. A neuron in one layer is connected to the corresponding neuron in the previous layer and to the neighbors of that neuron. Neurons in the output layer are also connected to the corresponding neurons in the input layer. The status of the neurons in the output layer has been viewed as a *fuzzy set* and various measures of fuzziness of this set have been used to model the error (instability of the network) of the system. The weight updating rules under each model have been developed. This error is then back propagated to correct weights and to reduce the system error in the next stage. A comparative study (both analytical and experimental) on the rate of learning for different error

measures has also been made. Therefore the proposed model not only describes a way of integrating the concept of fuzzy sets and neural networks, but also demonstrates a method of using a multi-layer network for performing (unsupervised) self-organizing task in image processing problems.

The results obtained by the proposed model are found to be comparable to those obtained in Chapters 4-6 even for very low values of *SNR*. It is seen that the rate of learning affects the output, specially when the noise level is very high. The outputs are better for lower learning rates (*indices of fuzziness*) and they deteriorate with the increase of rate of learning (*entropy measures with $q \geq 2$*).

Chapter 8

Robustness analysis of neural network based segmentation techniques under component failure

8.1 Introduction

A neural network based system consists of a large number of neurons with massive connectivity among them (thereby introducing redundancy). Local connectivity among the neurons being very high, the system is claimed to be highly robust and can be applied even when the input is ill-defined and/or defective/partial or noisy. In Chapters 4-7 we have investigated the robustness of our neural network based techniques with respect to noisy input when they were used to extract object regions from highly noisy scenes. As the neural network based systems consist of a large number of components, the possibility of some of its components (nodes or links) being failed to work is very high. In the present chapter we investigate the ruggedness of these segmentation algorithms under failure of components. It may be mentioned here that nothing has been suggested, to date, in the literature about *how to check this ruggedness*.

The work described in the present chapter derives a way of checking the ruggedness of neural network based systems by damaging some of the nodes and/or links and noticing the change in their performance. The damaging process is viewed as a pure death process [292] and is formulated as a *Poisson process*. The nodes and/or links, to be damaged are chosen randomly. The time instants (i.e., when a damage occurs) are determined by drawing random samples from an appropriate probability distribution [292].

Though the principle is valid, in general, for any neural network based systems, we have considered here the methods developed in Chapters 4-7, for image segmentation and object extraction as a demonstration. Therefore, the present investigation also provides a comparison of these object extraction algorithms with respect to component failure.

To study the ruggedness of these systems under component failure, their performance is evaluated in terms of percentage (%) of pixels correctly classified. The % is computed with respect to the actual object-background regions of the image. Outputs are evaluated in four different situations : i) no damage ii) damaging only nodes iii) damaging only links, and iv) damaging both nodes and

links simultaneously. The simulation study is done on various noisy versions of a synthetic image and on a real image (*Noisy tank*) using different amount of damaging.

8.2 Modeling of failures and sampling in a neural network

In this section the model of failure process is described and a way of applying the sampling theory for simulation of the failure process is stated.

8.2.1 Modeling of failures

Let us consider a neural network system with N components, where a component could be a node (processor) or a link. During the operation of the network some of its components may fail. We make the following assumptions about the failure process :

- i : The system has N *identical* components at the time instant $t = 0$ (when the operation of the network starts).
- ii : If a component fails, it fails for ever (no repair or replacement).
- iii : Failure of components occurs at an average rate of μ per unit time.
- iv : The probability of an event occurring between time t and $t + h$ depends only on the length h , i.e., the probability does not depend on either the number of events that has occurred upto time t or the specific value of t , i.e., the probability density function has stationary increments.
- v : The probability of a failure during a very small interval of time h is positive but less than one (1), i.e., not certain.
- vi : At most, one failure can occur during a very small interval of time h .

Let $p_n(t)$ be the probability that the system has n components active at time instant t , i.e., $N - n$ failures during time interval $[0, t]$. It can be shown [292] that under the assumptions i-vi, $p_n(t)$ is given by the formula :

$$p_n(t) = \frac{e^{-\mu t}(\mu t)^{N-n}}{(N-n)!}, \quad n = 1, 2, \dots, N \quad (8.1)$$

and

$$p_0(t) = 1 - \sum_{n=1}^N p_n(t). \quad (8.2)$$

Thus we see that $p_n(t)$ is a *truncated Poisson distribution* with mean μt .

If $f(t)$ is the probability density function (*pdf*) of the inter-failure time (i.e., time interval between two successive failures), then one can show that for the above mentioned Poisson process, $f(t)$ is given by

$$\begin{aligned} f(t) &= \mu e^{-\mu t}, \quad t > 0 \\ &= 0, \quad t \leq 0. \end{aligned} \quad (8.3)$$

Thus, when the failure process is governed by a Poisson distribution, the inter-failure time is then described by an exponential distribution (equation 8.3) with expected value (mean)

$$\begin{aligned} E(t) &= \int_0^{\infty} t f(t) dt \\ &= \frac{1}{\mu}. \end{aligned} \quad (8.4)$$

8.2.2 Sampling

In order to simulate the failure process, one needs to draw random samples from the exponential distribution (equation 8.3). Before describing the exact algorithm, let us first consider the general strategy for sampling from any distribution.

Let $f(x)$ be the *pdf* of the random deviate x , and $F(x)$ be the cumulative density function (*cdf*) of x , i.e.,

$$F(x) = \int_{-\infty}^x f(t) dt. \quad (8.5)$$

It can be easily shown that the random variable $y = F(x)$ is uniformly distributed over $[0, 1]$, regardless of the distribution of x . Hence, if R is a random number drawn from uniform $[0, 1]$, then $x = F^{-1}(R)$ is a random sample from the pdf $f(x)$. Therefore, sampling from any distribution can be done using the following two step method.

Step 1 : Generate a random number R in $[0, 1]$ and assign it to $F(x)$.

Step 2 : Solve for x from $R = F(x)$.

The above sampling method is known as method of inversion.

Sampling from exponential distribution :

For an exponential distribution the pdf is

$$\begin{aligned} f(t) &= \mu e^{-\mu t}, & t > 0 \\ &= 0, & t \leq 0. \end{aligned} \quad (8.6)$$

Then

$$F(t) = \int_0^t \mu e^{-\mu x} dx = 1 - e^{-\mu t}.$$

If the random number drawn is R then

$$\begin{aligned} R &= F(t) \\ \text{or, } R &= 1 - e^{-\mu t} \\ \text{or, } t &= -\frac{1}{\mu} \ln(1 - R) = -\frac{1}{\mu} \ln R. \end{aligned} \quad (8.7)$$

The last step is possible, because if R is a random number in $[0, 1]$, then so is $(1 - R)$ and we can replace $(1 - R)$ by R for convenience.

It has been established before that if the failure process is described by a Poisson distribution, then the time between the occurrence of failures (inter-failure time) must follow the corresponding exponential distribution. Thus in order to simulate the component failure process described by the Poisson distribution with mean μt , over a time period $[0, T]$, all one has to do is to sample the corresponding exponential distribution with mean $1/\mu$ as many times as necessary until the sum of the corresponding exponential random samples generated exceeds T for the first time. It can further be explained as follows.

Suppose R_i is the i th random sample drawn from uniform $[0, 1]$ distribution, then from equation (8.7)

$$t_i = -\frac{1}{\mu} \ln R_i \quad (8.8)$$

is the i th sample from the exponential distribution (equation 8.3). Therefore

$$T_i = \sum_{j=1}^i t_j \quad (8.9)$$

gives the time instant when the i th failure occurs (a component failure). The process is repeated for the largest number of times (k , say) such that $T_k \leq T$.

8.3 Application of failure model to ruggedness evaluation

The ruggedness of a neural network based system can be investigated under three different situations :

- failure of only nodes,
- failure of only links, and
- failure of both nodes and links.

We have seen in section 8.2.1 that if the components are identical and failures are independent, then the inter-failure time follows an exponential distribution. Therefore, while considering failures of only nodes or only links, we can use exponential distribution (equation 8.3) to draw samples for inter-failure time. Section 8.2.2 discusses how to compute the time instant T_i of the i th failure, but does not mention about the selection of the node (link) to be damaged at that instant. Since all nodes (links) are assumed to be identical, one can select a node (to be damaged) randomly in such a manner that each node (link) has an equal chance of being selected.

The situation becomes a little more complicated when both nodes and links are allowed to fail. A node failure effectively disables several links connected to it. On the other hand, failure of a set of links may effectively disable a node. Some of the assumptions stated in section 8.2.1 are not valid if we consider both nodes and links failure together, because the set of components are not identical and failure of one component may cause effective failure of another. We propose to investigate the system under the following two simplifying situations :

Case I : Do not distinguish between nodes and links, i.e., treat either of them as a component. The damaging process is then viewed to follow a single Poisson distribution.

Case II : Nodes and links are treated as two different components. The damaging process will then be governed by two different Poisson distributions, one for the nodes and the other for the links. No interdependence between the two distributions is assumed.

The components which are getting *damaged* can be chosen *randomly*. When we are damaging only one type of component (i.e., either node or link) the easiest way to select a component is to assign a unique number to each component and use random numbers in such a way that each node has an equal chance of inclusion. If an already selected component reappears then one has to ignore that drawing. When one is damaging both nodes and links together the selection process will be as follows depending on the above mentioned two cases. Suppose there are M nodes and N links.

Case A : Each component is assigned a unique integer number in the range $[1, M + N]$, and a component can be selected by drawing randomly an integer in that range.

Case B : Assign each node a unique integer in the range $[1, M]$ and each link a unique integer in the range $[1, N]$. Selection of a node can be done by drawing a random integer in $[1, M]$. Links are also selected similarly.

Note that *Case A* does not discriminate between nodes and links, and hence inter-damage times are drawn from one distribution for both nodes and links.

Case B, on the other hand, does discriminate between nodes and links, and uses two independent distributions (one for links and the other for nodes).

The model of damaging process developed in this section is, in general, applicable for any neural network based system. In the sections that follow an application of the principle is demonstrated on the neural network based object extraction algorithms which have been developed in Chapters 4-7. Besides demonstrating the effectiveness of the model, this investigation will also enable us to provide a comparative study of ruggedness of these object extraction systems.

8.4 Simulation and results

In order to demonstrate the utility of the failure model described in section 8.2 and to establish the robustness of different object extraction algorithms described in Chapters 4-7, a simulation study is done. The only parameter for Poisson/exponential distribution is the mean value μ . For the simulation purpose, μ is estimated as follows.

8.4.1 Way of simulation

Suppose a single layer network has C components and requires T seconds for I iterations on a mono-process system. Then for parallel implementation, time per iteration = $\frac{T}{CI}$, and the total processing time is $T' = \frac{T}{C}$. Let D be the total number of components to be damaged during the processing time T' , then an estimate of μ is

$$\mu = \frac{D}{T'}.$$

Let the inter-damage time periods (samples) be t_j , $j = 1, 2, \dots, L$ ($L \approx D$). Now for each of the L time instants select a component to be damaged. In other words, select L components and damage the i th selected component after T_i

seconds, where

$$T_i = \sum_{j=1}^i t_j.$$

Now, if the i th component is damaged at a time instant T_i and T_i falls in the k th iteration, then for the k th and subsequent iterations we assume the i th component to be damaged.

For a multi-layer (m -layer, say) neural network, let there be C components per layer, and the network requires T seconds for stabilization. The i th ($i > 1$) layer can work only after the $(i - 1)$ th layer has finished its task. Since there exists a feedback path from the output to the input layer, the output layer may be considered to be the predecessor of the input layer, i.e., the input layer can work (except the starting moment) only after the output layer has completed its operation. Thus parallel implementation can be done only in the C components in a layer. Between layers the operations are always sequential. Hence, the total time required for parallel implementation of such a system is $T' = \frac{T}{C}$.

A performance evaluation criterion (percentage of correct pixel classification) of the object extraction algorithms used in the present work is described below.

Let N_1 be the number of pixels that actually belong to object, but are classified as background pixels by an algorithm; and N_2 be the number of background pixels that are labeled by the algorithm as object pixels. Then the percentage of pixels correctly classified (pcc) by the algorithm is given as

$$pcc = \left(1 - \frac{N_1 + N_2}{M \times N}\right) \times 100 \quad (8.10)$$

for an $M \times N$ image.

8.4.2 Results

The present simulation study is done for all the three object extraction algorithms (algorithms described in Chapters 5 and 6 with continuous dynamics and N^2 connectivity, and that in Chapter 7 with quadratic index of fuzziness

as error measure, which will be referred to as *Method 1*, *Method 2* and *Method 3* respectively in this chapter) using three different noisy versions [Figures 6.3(a), 6.4(a) and 6.5(a)] of a synthetic image as input. The noisy input have signal to noise ratio, *SNR*, (defined in Chapter 5, pp. 135) equal to 2.4, 1.2 and 0.75, respectively. The experiment is done under the following four conditions :

- i : without damaging nodes or links,
- ii : damaging nodes only;
- iii : damaging links only, and
- iv : damaging nodes and links both.

For each of the cases ii, iii and iv investigation has been done with a failure of 2%, 5% and 10% of total components. The percentages of pixels correctly classified under all the above mentioned conditions is given in Tables 8.1-8.12.

Method 1

Table 8.1: Classification accuracy after node-damage

Input image with <i>SNR</i>	Number of nodes damaged			
	0	328 ($\simeq 2\%$)	819 ($\simeq 5\%$)	1638 ($\simeq 10\%$)
2.4	99.71	99.67	99.68	99.40
1.2	99.12	99.11	99.04	98.83
0.75	97.55	97.44	97.54	97.18

From the tables it is evident that when only nodes or only links are damaged (even with a high percentage $\simeq 10$) the performance is not deteriorated much for all the methods. However, if the nodes and links (Tables 8.3-8.4, 8.7-8.8 and 8.11-8.12) are damaged together, the performance is degraded slightly more. Even under this condition the results for the first two methods (based on Hopfield type neural networks) are seen to be better compared to the third one. For the third method which is based on a multi-layer network, the initial assignment of target outputs (0 or 1) depends on their closest distance from

Method 1

Table 8.2: Classification accuracy after link-damage

Input image with <i>SNR</i>	Number of links damaged			
	0	2622 ($\simeq 2\%$)	6554 ($\simeq 5\%$)	13107 ($\simeq 10\%$)
2.4	99.71	99.66	99.55	99.48
1.2	99.12	99.11	99.06	98.77
0.75	97.55	97.57	97.70	97.61

Method 1

Table 8.3: Classification accuracy after node and link damage (nodes and links are treated as two separate components)

Input image with <i>SNR</i>	Number of nodes+links damaged			
	0	328+2622 ($\simeq 2\%$)	819+6554 ($\simeq 5\%$)	1638+13107 ($\simeq 10\%$)
2.4	99.71	99.54	98.92	99.05
1.2	99.12	99.02	98.92	98.38
0.75	97.55	97.67	97.60	96.88

Method 1

Table 8.4: Classification accuracy after node and link damage (nodes and links are treated as a single component)

Input image with <i>SNR</i>	Number of (nodes+links) damaged			
	0	2952 ($\simeq 2\%$)	7371 ($\simeq 5\%$)	14742 ($\simeq 10\%$)
2.4	99.71	99.69	99.43	99.04
1.2	99.12	98.63	98.86	98.67
0.75	97.55	97.35	97.38	96.95

Method 2

Table 8.5: Classification accuracy after node-damage

Input image with <i>SNR</i>	Number of nodes damaged			
	0	328 ($\simeq 2\%$)	819 ($\simeq 5\%$)	1638 ($\simeq 10\%$)
2.4	99.38	99.26	98.93	98.49
1.2	98.80	98.57	98.40	97.97
0.75	98.03	97.95	97.85	97.28

Method 2

Table 8.6: Classification accuracy after link-damage

Input image with <i>SNR</i>	Number of links damaged			
	0	2622 ($\simeq 2\%$)	6554 ($\simeq 5\%$)	13107 ($\simeq 10\%$)
2.4	99.38	99.22	98.93	98.55
1.2	98.80	98.75	98.62	98.30
0.75	98.03	97.90	97.71	97.72

Method 2

Table 8.7: Classification accuracy after node and link damage (nodes and links are treated as two separate components)

Input image with <i>SNR</i>	Number of nodes+links damaged			
	0	328+2622 ($\simeq 2\%$)	819+6554 ($\simeq 5\%$)	1638+13107 ($\simeq 10\%$)
2.4	99.38	98.97	98.50	97.97
1.2	98.80	98.68	98.59	97.75
0.75	98.03	97.72	97.53	97.30

Method 2

Table 8.8: Classification accuracy after node and link damage (nodes and links are treated as a single component)

Input image with <i>SNR</i>	Number of (nodes+links) damaged			
	0	2952 ($\simeq 2\%$)	7371 ($\simeq 5\%$)	14742 ($\simeq 10\%$)
2.4	99.38	99.05	98.46	97.95
1.2	98.80	98.63	98.36	97.97
0.75	98.03	97.64	97.02	96.60

Method 3

Table 8.9: Classification accuracy after node-damage

Input image with <i>SNR</i>	Number of nodes damaged			
	0	656 ($\simeq 2\%$)	1638 ($\simeq 5\%$)	3276 ($\simeq 10\%$)
2.4	99.58	99.58	99.57	99.42
1.2	98.90	98.90	98.88	98.74
0.75	97.17	97.18	97.37	97.24

Method 3

Table 8.10: Classification accuracy after link-damage

Input image with <i>SNR</i>	Number of links damaged			
	0	5898 ($\simeq 2\%$)	14746 ($\simeq 5\%$)	29491 ($\simeq 10\%$)
2.4	99.58	99.58	99.58	99.51
1.2	98.90	98.90	98.89	98.90
0.75	97.17	97.18	97.23	97.24

Method 3

Table 8.11: Classification accuracy after node and link damage (nodes and links are treated as two separate components)

Input image with <i>SNR</i>	Number of nodes+links damaged			
	0	656+5898 ($\simeq 2\%$)	1638+14746 ($\simeq 5\%$)	3276+29491 ($\simeq 10\%$)
2.4	99.58	99.57	99.51	97.88
1.2	98.90	98.89	98.82	97.05
0.75	97.17	97.20	97.08	95.17

Method 3

Table 8.12: Classification accuracy after node and link damage (nodes and links are treated as a single component)

Input image with <i>SNR</i>	Number of (nodes+links) damaged			
	0	6554 ($\simeq 2\%$)	16384 ($\simeq 5\%$)	32767 ($\simeq 10\%$)
2.4	99.58	99.58	99.51	98.38
1.2	98.90	98.89	98.90	97.69
0.75	97.17	97.18	97.08	95.17

the output status. This may result in initial mis-labeling of some of the pixels. Normally this initial mis-labeling gets corrected by incorporating information from the neighboring nodes while updating the weights of the links. But when a large number of nodes and links is damaged some of the initial mis-labeling fail to be corrected; thereby deteriorating the performance. Since the classification accuracy is not hampered much even when a large number of nodes and/or links are damaged, the systems can be said to be rugged to a great extent.

The objects extracted by different methods (with approximately 5% damaging of nodes/links) for a corrupted version ($SNR=1.2$ shown in Figure 6.4(a)) of the synthetic image are shown in Figures 8.1-8.3. The results obtained without damaging the links/nodes (e.g., Figures 6.9(c), 6.4(c) and 7.4(c), respectively) are also included here for the convenience of comparison. From the figures one can visualize that the 5% damage does not make it difficult to recognize the objects extracted from noisy environments.

The experiment has also been done on the image of a *Noisy tank* [Figure 4.4(a)]. For this image there is no known target output (i.e., the exact shape of the object is not known). So, the percentage of correct classification can not be computed here. For visual inspection the outputs for these methods (with approximately 5% damage) are depicted in Figures 8.4-8.6.



(a)



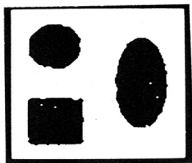
(b)



(c)

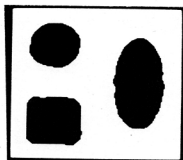


(d)

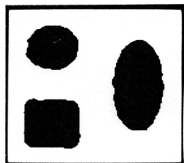


(e)

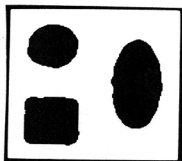
Figure 8.1: Extracted objects by Method 1 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)



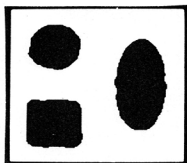
(a)



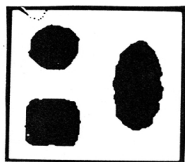
(b)



(c)



(d)

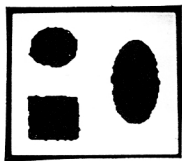


(e)

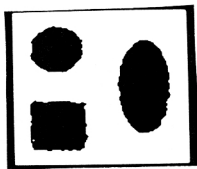
Figure 8.2: Extracted objects by Method 2 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)



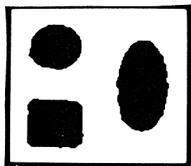
(a)



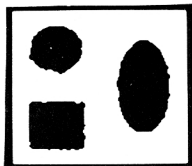
(b)



(c)

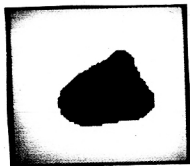


(d)



(e)

Figure 8.3: Extracted objects by Method 3 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)



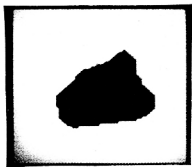
(a)



(b)



(c)

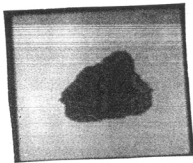


(d)

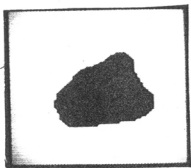


(e)

Figure 8.4: Extracted *Noisy tank* image by Method 1 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)



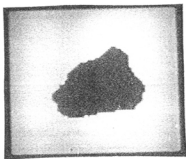
(a)



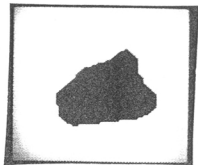
(b)



(c)

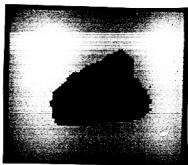


(d)



(e)

Figure 8.5: Extracted *Noisy tank* image by Method 2 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)



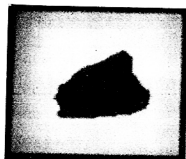
(a)



(b)



(c)

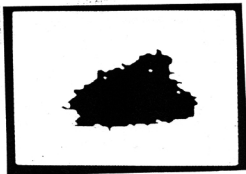


(d)

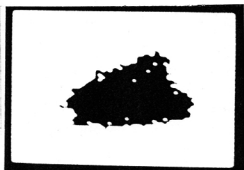


(e)

Figure 8.6: Extracted *Noisy tank* image by Method 3 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)



(a)



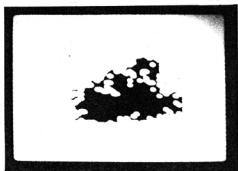
(b)



(c)



(d)



(e)

Figure 8.7: Extracted *Noisy tank* image by Method 4 : a) without damaging, b) damaging nodes only, c) damaging links only, d) damaging nodes and links (treating them as two separate components), e) damaging nodes and links (treating them as a single component)

It may be mentioned that the methodologies developed in Chapter 4 are comparatively less noise immune. One of the methods (Algorithm 2, pp. 114) has been implemented here on the *Noisy tank* image with approximately 5% damage, as an illustration. (We will refer this as *Method 4*.) The results are shown in Figure 8.7. From the results we notice that the deterioration in the quality of the output is not much.

8.5 Conclusions and discussion

Robustness with respect to component failure (i.e., ruggedness) is one of the most important characteristics of neural network based systems. We have investigated here the robustness of four different object extraction algorithms developed in Chapters 4-7. Since no method has been suggested to date describing a way of studying this characteristics, we have formulated, first of all, a general theory where the damaging process has been viewed as a *pure death process* and is modeled as *Poisson process*.

Robustness of these neural network based segmentation algorithms has been evaluated under different amount of noise and damaging. The performance is seen to be satisfactory even when a high percentage of nodes/links is damaged. This, in turn, establishes the robustness of the proposed neural network based object extraction algorithms under component failure. Among all the methods considered, Methods 1 and 2 are seen to be more robust. It may be noted that the noise immunity of the algorithms has already been addressed in Chapters 4-7.

Chapter 9

Conclusions and further scope of work

9.1 Conclusions

We have presented in this thesis some results of investigations, both theoretical and experimental, which demonstrate the effectiveness of some of the concepts of fuzzy set theory and artificial neural network models in formulating various image segmentation/object extraction methodologies for image analysis. Initially, a set of fuzzy set theoretic thresholding and pixel classification techniques using some new measures on image correlation and fuzzy geometrical properties of images has been described for providing both fuzzy and nonfuzzy segmentation. Then some neural network based methods for extracting objects from noisy environments are presented using self-organizing feature map, auto-associative memory model and a multi-layer neural network. An attempt has also been made in this context to integrate the concepts of fuzzy sets and a neural network model by incorporating various fuzziness measures in a multi-layer network. Ruggedness of these systems under component failure is studied with the help of a stochastic model. The performance of the proposed methods has been compared with that of several related techniques when a set of images with widely different types of histograms is considered as input.

The fuzzy set theoretic tools developed are as follows. A concept of image correlation between two fuzzy properties of an image has been introduced based on the definition of correlation between two membership functions [120]. Two new measures called *transitional correlation* and *within-class correlation* are then defined which involve information on co-occurrence of pixels. These measures reflect grayness ambiguity in an image. Some new fuzzy geometrical properties namely, length, breadth, index of area coverage (*IOAC*), major axis, minor axis, center of gravity and density have been introduced to model spatial (geometrical) ambiguities in an image. Modified versions of *adjacency* and *degree of adjacency* between regions (introduced by Rosenfeld [102]) have been defined.

Various algorithms have been formulated based on some of these new properties/measures for providing both fuzzy (representing the degree of belonging of a pixel to a region) and nonfuzzy segmented outputs through gray level thresholding and pixel classification. (Fuzzy segmentation enables to avoid

committing oneself to a specific hard thresholding for ill-defined regions.) Gray level or histogram thresholding algorithms involve maximization of the correlation measures and minimization of the geometrical properties namely, *IOAC* and degree of adjacency. Adjacency measure is considered for pixel classification. These algorithms are seen to be flexible with respect to the selection of membership functions. The nonfuzzy thresholds provided by the proposed measures are seen to be comparable to those obtained by the index of fuzziness minimization [115] and some of the recently developed algorithms [62, 64] based on the co-occurrence of pixels. Algorithms involving information from the quadrants C_2 and C_4 of the co-occurrence matrix (Figure 2.1, pp. 56) of an image are more efficient as compared to those based on C_1 and C_3 in extracting object regions. This corroborate the earlier findings in [62, 64]. The measure *IOAC* is seen to provide better results for noncompact objects as compared to the *compactness* measure [102].

It has been proved that the criterion of *maximizing correlation* is able to detect the valleys in a histogram. A relation between the fuzzy correlation and the quadratic index of fuzziness has been theoretically established and experimentally verified. Efficient techniques for computing some of the fuzzy geometrical properties are also suggested.

Different types of neural network models have been designed in order to develop methodologies for object extraction or object background classification from noisy scenes. In one of the algorithms the object extraction problem has been viewed as a self-organizing task and is solved by a modified version of Kohonen's self-organizing feature map model. Here a single neuron is assigned to each pixel. Unlike Kohonen's model, the input given to different neurons are different. The convergence of the algorithm is proved mathematically. Statistical criteria such as square error minimization and correlation maximization have been suggested for choosing the optimal output.

In another case, the task of image segmentation is viewed as an optimization problem and solved by a Hopfield type network. An objective function (which is similar to the expression of energy of a Hopfield type network) has been formulated for object background separation in such a manner that in the stable

state of the network neurons corresponding to homogeneous regions would be in ON state. A modified version (topologically) of Hopfield's model is then proposed for this optimization task. Both the discrete and the continuous dynamics have been studied. A polynomial over a fixed domain is used as the input/output transfer function, instead of an asymptotic function. The problem of assignment of weights is tackled using the concept of probability of co-occurrence of pixels.

The problem of finding the maximum a posterior probability (*MAP*) estimate of a scene from a noise corrupted realization (modeled by Gibbs Random Field) is computationally exponential in nature [74]. A modified version (both topologically and neuron characteristic wise) of Hopfield's model has been designed for solving this problem. The energy function of the network is formulated in such a way that its minimum value corresponds to the *MAP* estimate of the scene. A possible hardware realization of the neurons is provided. The discrete and the continuous dynamics, the input/output transfer function used are the same as in the previous case. Mathematical proof for the convergence of the algorithm is also given.

An attempt is then made to incorporate various fuzziness measures in a multi-layer network for object extraction. Each neuron is connected to the corresponding neuron in the previous layer and to its neighbors. Another structural difference from the standard multi-layer perceptron is that there exists a feedback path from the output to the input layer. The status of the neurons in the output layer is described as a *fuzzy set*. A fuzziness measure of this set is then used as a measure of error of the system (amount of instability of the network). Besides describing a method of integrating the concepts of fuzzy sets and neural networks, the work also demonstrates a way of using a multi-layer neural network for performing (unsupervised) self-organizing task in image processing.

The outputs (object regions extracted from noisy images) obtained by the neural network based algorithms are found to be quite satisfactory and noise immune. Methods based on Hopfield type model (Chapters 5 and 6) are seen to be more efficient than those employing Kohonen's feature mapping algorithm (Chapter 4) and a self-organizing multi-layer network (Chapter 7). The sys-

tem developed using the concept of feature mapping criterion converges faster compared to others. Among the two systems developed using Hopfield type networks, the second one (Chapter 6) which employs Gibbsian model for image data produced better results. Again, the results obtained by the continuous dynamics are found to be more noise immune than those by the discrete dynamics. The outputs produced by the multi-layer network based system are comparable to those obtained by Hopfield type model, but better than those obtained with the feature mapping criterion. The performance of all the neural network based systems is better than that by the probabilistic relaxation technique [2, 68].

Finally, the ruggedness of these systems with respect to component failure (damaging of nodes/links) has been evaluated. In this context, we formulated a general theory applicable to any neural network based information processing system. Here the damaging (or component failure) process has been viewed as a pure death process and is modeled by *Poisson process*. In order to choose the instants or moments of damaging, a statistical sampling technique is used. The nodes/links to be damaged were determined randomly. The object extraction systems based on Hopfield type network (described in Chapters 5 and 6) are found to be more rugged than others when various noisy images are considered as input.

9.2 Further scope of work

Applications of the fuzzy geometrical properties like, major and minor axes, center of gravity and density have not been considered in the present investigation. Their definitions are introductory and may be further investigated. The utility of some of the fuzzy geometrical properties and the fuzzy correlation measure as objective functions of neural network based object extraction algorithms needs to be investigated. Some of them may also be used as input features while training an artificial neural network. As the fuzzy geometrical properties are time consuming to compute; parallel implementation of fuzzy geometry based algorithms using neural network architectures should constitute

another problem of study. Fuzzy segmentation algorithms described here are basically based on gray level thresholding by selecting an optimal membership function. Procedures based on *if-then* rule using various image properties may also be attempted in this context.

The algorithms using neural networks have been developed for two class segmentation problem and use only one feature. An extension of them is necessary to deal with more than one feature and more than two classes in order to make them more general. These algorithms are seen to be suitable for extracting compact regions. Proper modification is therefore necessary in selecting neighborhoods depending on local information so that noncompact and thin regions can be extracted. We have incorporated various fuzziness measures in a multi-layer network to use it as an unsupervised system. The intermediate outputs are fuzzy, but the final output is nonfuzzy. Incorporation of fuzzy hedges like *small*, *medium*, *high* in the input layer, and fuzzy membership in the output status has to be made for handling the uncertainty in the input data efficiently. Incorporation of fuzzy set theoretic concepts in various stages of self-organizing or auto-associative network based object extraction algorithms in order to handle ill-defined input data more efficiently may be tried out. In the self-organized feature mapping based object extraction algorithm the weights associated with a neuron do not get affected by the status of the neighboring neurons, during their updating. Information about the status of the neighboring neurons may be used while updating the weights of a neuron to enhance the noise immunity of the algorithm.

In general, the different neural network architectures and the algorithms developed in this thesis for object extraction problem, should be extended in order to make them applicable to other image processing problems (like enhancement, edge detection, matching etc.). The robustness of the proposed algorithms with respect to noise and component failure needs to be theoretically established. The proposed model for checking ruggedness of the neural network based systems should be applied to other pattern recognition problems to increase its utility and to establish its generality. Application of the proposed network models to color image processing may be also attempted.

Development of methodologies based on genetic algorithms [279, 280] for determining the global optimal set of parameters (weights and biases) for neural network based systems should constitute another part of further investigation.

Appendix A

Preliminaries of fuzzy set theory and artificial neural networks

A.1 Introduction

Here we present some basic concepts and tools of fuzzy set theory and artificial neural network theory which are relevant (and not covered in Chapter 1) to the new segmentation methodologies developed in Chapters 2-7. It has two parts. The first part deals with some basic operations on fuzzy sets, including 'S' and 'π' functions, various measures of fuzziness in a set. The second part presents a brief review on the theory of neural networks including general description of neural networks, learning paradigm and various models such as Hopfield's network, Kohonen's network and the multi-layer perceptron.

A.2 Theory of fuzzy sets

As mentioned in section 1.3.2, a fuzzy subset A of the universe X is defined as a collection of ordered pairs

$$A = \{(\mu_A(x), x), \forall x \in X\} \quad (\text{A.1})$$

where $\mu_A(x)$, ($0 \leq \mu_A(x) \leq 1$) denotes the degree of belonging of the element x to the fuzzy set A .

Because fuzzy sets are a generalization of the classical sets, the embedding of conventional models into a larger setting endows fuzzy models with greater flexibility to capture various aspects of incompleteness or imperfection (i.e., deficiencies) in whatever information and data are available about real processes. In other words, the membership functions possess *elasticity*; thus, the higher the value of membership of an element to a set, the less the imprecisely defined concept of the fuzzy set must be stretched to accommodate the element. Hard membership functions, of course, are *inelastic*.

A.2.1 S and π functions

The 'S' and ' π ' functions [89] are defined as

$$\begin{aligned} S(x; a, b, c) &= 0 & \text{if } x \leq a \\ &= 2 \{(x-a)/(c-a)\}^2 & \text{if } a \leq x \leq b \\ &= 1 - 2 \{(x-c)/(c-a)\}^2 & \text{if } b \leq x \leq c \\ &= 1 & \text{if } c \leq x \end{aligned} \quad (\text{A.2})$$

where $b = \frac{a+c}{2}$.

$$\begin{aligned} \pi(x; b, c) &= S(x; c-b, c-\frac{b}{2}, c) & \text{if } x \leq c \\ &= 1 - S(x; c, c+\frac{b}{2}, c+b) & \text{if } x \geq c. \end{aligned} \quad (\text{A.3})$$

In $S(x; a, b, c)$, the parameter b is the crossover point, i.e., $S(b; a, b, c) = 0.5$. In $\pi(x; b, c)$ b is the bandwidth, i.e., the separation between the crossover points of π function and c is the central point at which $\pi = 1$.

Equations (A.2) and (A.3) define the membership or compatibility functions corresponding to fuzzy sets x is large and x is c , respectively. These two standard functions are found in many cases to be convenient to represent the membership function of a fuzzy set in the real problems. The parameters a , b and c are only to be adjusted to fit a specific function approximately.

A.2.2 A few basic operations on fuzzy sets

Let A and B be two fuzzy subsets of X . Some basic operations that can be performed on A and B are as follows :

- i : A and B are said to be equal (i.e., $A = B$) iff $\mu_A(x) = \mu_B(x)$, $\forall x \in X$.
- ii : A is contained in B (i.e., $A \subseteq B$) iff $\mu_A(x) \leq \mu_B(x)$, $\forall x \in X$.
- iii : The union of A and B (i.e., $A \cup B$) is characterized as

$$A \cup B = \{(\max_x[\mu_A(x), \mu_B(x)], x), \forall x \in X\}.$$
- iv : The intersection of A and B (i.e., $A \cap B$) is characterized as

$$A \cap B = \{(\min_x[\mu_A(x), \mu_B(x)], x), \forall x \in X\}.$$

- v : A^c , the complement of A is defined as
 $A^c = \{([1 - \mu_A(x)], x), \forall x \in X\}.$
- vi : Let A_α be the α -cut of the fuzzy set A . Then A_α is defined as
 $A_\alpha = \{x : \mu_A(x) \geq \alpha\}.$
- vii : Let β be a positive number, the β -level set A^β is defined as
 $A^\beta = \{([\mu_A(x)]^\beta, x), \forall x \in X\}.$
The operations *concentration* and *dilation* are defined as two special cases for $\beta = 2$ and $\beta = \frac{1}{2}$ as
 $con(A) = A^2$, and
 $dil(A) = A^{\frac{1}{2}}$, respectively.
- viii : The contrast intensification operation INT of a fuzzy set A is defined as

$$\mu_{INT}(x) = \begin{cases} [2\mu_A(x)]^2 & \text{if } 0 \leq \mu_A(x) \leq 0.5 \\ 1 - 2[1 - \mu_A(x)]^2 & \text{otherwise.} \end{cases}$$

- ix : The sharpened version A^* of a fuzzy set A is defined as

$$\begin{aligned} \mu_{A^*}(x) &\geq \mu_A(x) & \text{if } \mu_A(x) &\geq 0.5 \\ &\leq \mu_A(x) & \text{if } \mu_A(x) &\leq 0.5. \end{aligned}$$

A.2.3 Measures of fuzziness

Let us now discuss some measures which give the degree of fuzziness in a fuzzy set A . The degree of fuzziness expresses the average amount of ambiguity in making a decision whether an element belongs to the fuzzy set (or possesses the concept or property represented by the fuzzy set) or not. Such a measure (I, say) should have the following properties.

- i : $I(A) = \text{minimum}$ iff $\mu_A(x) = 0$, or $1, \forall x$.
- ii : $I(A) = \text{maximum}$ iff $\mu_A(x) = 0.5, \forall x$.
- iii : $I(A) \geq I(A^*)$, where A^* is a sharpened version of A .
- iv : $I(A) = I(A^c)$ where A^c is the complement of A .

Several authors [90, 97, 98, 99, 100, 101] have made attempts to define fuzziness measures based on these properties. A few of them [90, 97, 99] relevant to the present thesis are described here.

Index of fuzziness : The index of fuzziness of a fuzzy set A having n supporting points is defined as [90]

$$\nu(A) = \frac{2}{n^k} d(A, \underline{A}) \quad (\text{A.4})$$

where $d(A, \underline{A})$ denotes the distance between fuzzy set A and its nearest ordinary set $\underline{A}$. An ordinary set $\underline{A}$ nearest to the fuzzy set A is defined as

$$\mu_{\underline{A}}(x) = \begin{cases} 0 & \text{if } \mu_A(x) \leq 0.5 \\ 1 & \text{otherwise.} \end{cases} \quad (\text{A.5})$$

The value of k depends on the type of distance measure used. For example, $k = 1$ is used for generalized Hamming distance whereas, $k = 0.5$ for Euclidean distance. The corresponding indices of fuzziness are called *linear index of fuzziness* $\nu_l(A)$ and *quadratic index of fuzziness* $\nu_q(A)$. Thus we have

$$\begin{aligned} \nu_l(A) &= \frac{2}{n} \sum_{i=1}^n |\mu_A(x_i) - \mu_{\underline{A}}(x_i)| \\ &= \frac{2}{n} \sum_{i=1}^n [\min \{ \mu_A(x_i), (1 - \mu_A(x_i)) \}]. \end{aligned} \quad (\text{A.6})$$

Similarly, for Euclidean distance we have

$$\nu_q(A) = \frac{2}{\sqrt{n}} \sqrt{\left[\sum_{i=1}^n \{ \mu_A(x_i) - \mu_{\underline{A}}(x_i) \}^2 \right]}. \quad (\text{A.7})$$

Entropy : Entropy of a fuzzy set as defined by De Luca and Termini [99] is given by

$$H(A) = \frac{1}{n \ln(2)} \sum_{i=1}^n S_n(\mu_A(x_i)) \quad (\text{A.8})$$

with

$$S_n(\mu_A(x_i)) = -\mu_A(x_i) \ln \{ \mu_A(x_i) \} - \{ 1 - \mu_A(x_i) \} \ln \{ 1 - \mu_A(x_i) \}. \quad (\text{A.9})$$

Another definition of entropy given by Pal and Pal [97] using exponential gain function is

$$H(A) = \frac{1}{n(\sqrt{e} - 1)} \sum_{i=1}^n \{S_n(\mu_A(x_i)) - 1\} \quad (\text{A.10})$$

with

$$S_n(\mu_A(x_i)) = \mu_A(x_i)e^{1-\mu_A(x_i)} + \{1 - \mu_A(x_i)\}e^{\mu_A(x_i)}. \quad (\text{A.11})$$

All the above measures lie in $[0, 1]$ and satisfy properties i through iv.

Note that, these measures first of all compute the fuzziness related to individual support of the set and then make an average over all the supports to get a quantification of the amount of ambiguity the set possesses. Such global measures of uncertainty on a fuzzy set are not always sufficient. Often we are interested in questions such as : if a team of, say, eleven players from a set of twenty players, to what extent, on the average, might the team be called *good*? This raises two important issues : namely, how to measure the degree to which a collection of objects, as a whole, possesses the property of *goodness* and how to measure the average amount of ambiguity related to such collections. Answers to these issues have been proposed in [118] by defining r th order entropy $H^r(A)$, $r \geq 1$ of a fuzzy set A (using both logarithmic and exponential gain functions) which provides a measure of the average amount of uncertainty (ambiguity) in making a decision on any subset with r elements as regards its possession of the crispness property. Equations (A.8) through (A.11) correspond to a special case of H^r for $r = 1$.

As mentioned in section 1.3.2.A these measures may be used in representing grayness ambiguity in an image, i.e., the indefiniteness in making a decision whether an individual pixel is black or white, or a collection of pixels possesses an image property or not. There is another kind of ambiguity in an image called *spatial or geometrical ambiguity* (which refers to indefiniteness in the shape and geometry of regions within the image). These can be represented by fuzzy geometrical properties as described in Chapter 3.

A.3 Theory of neural networks

The primary task of all neural systems is to control various biological functions. A major part of it is connected with behavioral tasks, i.e., controlling the state of the organism with respect to its environment. Human beings can do it almost instantaneously and without much effort because of the remarkable processing capabilities of the nervous system. For example, if we look at a scene or listen to a music, we can easily (without conscious effort and time) recognize it. This high performance rate must be derived, at least in part, from the large number of neurons ($\sim 10^{11}$) participating in the task and also from the huge connectivity (thereby providing consensus) among them. The participating neurons operate in parallel and try to pursue a large number of hypotheses simultaneously that emulate the system to provide output almost in no time.

Artificial neural network (*ANN*) models [134]-[165] try to emulate the biological neural network/nervous system with electronic circuitry. *ANN* models have been studied for many years with the hope of achieving human like performance (artificially), particularly in the field of image and pattern recognition. The models are extreme simplifications of the actual human nervous system. The computational elements (called neurons/nodes/processors) are analogous to the fundamental constituents (called neurons) of the biological nervous system. The biological neurons have a large number of connectivity to its neighbors via synapses, whereas in the electronic model the connectivity is very low. Like biological neurons the artificial neurons also get input from different sources, combine them and provide a single output. The topological organization of these neurons tries to emulate the organization of the human nervous system.

Figure A.1 shows the structure of a biological neuron. Such a neuron gets input via synaptic connections from a large number of its neighbors. The accumulated input is then transformed to provide a single output which is transmitted to other neurons through the axon. If the total input is greater than some threshold (θ , say), then the neuron transmits output to others and is said to have fired. Total amount of output given by the neuron is measured by its firing rate, i.e., the number of times the neuron fires per unit time.

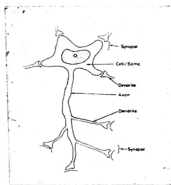


Figure A.1: Schematic representation of a biological neuron

A schematic representation of an electronic neuron is given in Figure A.2. The neuron gets input through the resistors from other neighboring neurons. By the operational amplifier the total input is then converted to a single output (summing the inputs and then amplifying it) and passed on to other neurons. So, as far as the primitive type of operations are concerned, an electronic neuron emulates a biological neuron. The artificial neurons are then connected to form a network structure which topologically as well as functionally tries to emulate the human nervous system. The resistance value by which a neuron is connected to another determines the connection strength (weight) or synaptic strength between the neurons. The larger is the resistance, the smaller is the connection strength and vice versa.

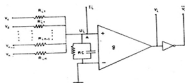


Figure A.2: Schematic representation of an electronic neuron

A.3.1 General properties of neural networks

ANN models (normally referred as neural networks) attempt to capture the key ingredients responsible for the remarkable capabilities of the human nervous system. A wide variety of models has been proposed for various purposes; they differ in structure and details, but share some common features. The common features are :

- Each node/neuron/processor is usually described by a single real variable (called its *state*), representing its output.
- The nodes are densely connected so that the state of one node affects the potential (total input) of many other nodes according to the weight or strength of the connection.
- The new state of a node is a non-linear function of the potential created by the firing activity of other neurons to it.
- Input to the network is given by setting the states of a subset of the nodes to specific values initially.
- The processing takes place through the evolution of the states of all the nodes of the network according to the dynamics of a specific network and continues until it stabilizes.
- The training (learning) of the net is a process whereby the values of the connection strengths are modified in order to achieve the desired output.

Artificial neural network models are reputed to enjoy the following four major advantages:

- *adaptivity*- adjusting the connection strengths to new data/information,
- *speed*- due to massively parallel architecture,
- *robustness*- to missing, confusing, ill-defined/noisy data and failure of components,

- *optimality*- as regards error rates in performance.

Though there are some common features of the different proposed *ANN* models, they differ in finer details and in the underlying philosophy. They are mainly classified according to the learning strategies involved in them. Before describing the details of the different architectures, let us go through various learning techniques.

A.3.2 Learning paradigm in neural networks

The learning situations in artificial neural networks can be broadly categorized into following two distinct types :

- i : associative learning;
- ii : regularity detector.

Associative Learning : In associative learning the system learns to produce a particular pattern of activation on one set of units whenever another pattern appears on another set of units. This further can be divided into

- autoassociator, and
- hetero/pattern associator.

In auto associator a set of patterns is repeatedly presented and the system is supposed to store these patterns. Then, later, a part of one of the original patterns or possibly a pattern similar to one of the original patterns is presented and the task is to *retrieve* the original pattern through a kind of pattern completion. This is an auto-associative process in which a pattern is associated with itself so that a degenerated version of the original pattern can act as a retrieval cue. Essentially it acts as a content addressable memory (*CAM*). Hopfield's model of *ANN* [137, 139, 293] employs this type of learning.

In hetero/pattern associator a set of pairs of patterns is repeatedly presented for training or learning of the network parameters. The system is to learn

that when one member of the pair is presented, it produces the other. In this paradigm one seeks a mechanism in which an essentially arbitrary set of input patterns can be paired with an arbitrary set of output patterns. Therefore, associative learning is supervised where a set of training samples (samples of known classes/origin) is first of all used to learn the network parameters and then based on this learning it provides output corresponding to some new input. Multi-layer perceptron [134] is an example which exploits hetero-associative learning.

Regularity detector : In this paradigm the system learns to respond to some common features/properties present in the input patterns. In general, such a scheme should be able to form the basis for the development of feature detectors and should discover statistically salient features of the input population. Unlike the other learning methods, there is no prior set of categories into which the patterns are to be classified. Here, the system must develop its own featural representation of the input stimuli which captures the most salient features of the population of the input pattern. In this sense, this learning is unsupervised because this does not use labeled samples for learning the network parameters. It may be mentioned that this type of learning resembles the biological learning to a great extent. Kohonen's model of self-organized feature mapping [135, 136], adaptive resonance theory of Carpenter and Grossberg [144] and neo-cognitron of Fukushima [142] try to model the function of regularity detectors.

Though, there exist a large number of ANN models we shall be describing here only three of them which are relevant to the new methodologies developed in this thesis for image segmentation. They are Hopfield's model of associative memory, Kohonen's model of self-organizing feature map and the multi-layer perceptron.

A.3.3 Hopfield's model

Hopfield [137, 138, 139] proposed a neural network model which functions as an auto-associative memory. Before going into the details of the dynamics of the network and how it acts as an associative memory, we shall specify how we

can construct an associative memory (also refer [136, 146]).

Let there be given an associated pattern pair (x_k, y_k) where x_k is an N_1 dimensional vector and y_k is an N_2 dimensional column vector. The matrix

$$M = y_k x_k^t \quad (\text{A.12})$$

can serve as an associative memory where $y_k x_k^t$ is the direct product/dyadic product between the two vectors. The retrieval of the pattern y_k can be obtained when the input x_k is given, using

$$\begin{aligned} M x_k &= y_k x_k^t x_k \\ &= y_k (x_k^t \cdot x_k) \\ &= y_k \quad [\text{if } x_k^t \cdot x_k = 1]. \end{aligned} \quad (\text{A.13})$$

For illustration, let us consider the vectors x_k and y_k to be of 3-dimensional. Then

$$M = y_k x_k^t = \begin{pmatrix} y_{k1}x_{k1} & y_{k1}x_{k2} & y_{k1}x_{k3} \\ y_{k2}x_{k1} & y_{k2}x_{k2} & y_{k2}x_{k3} \\ y_{k3}x_{k1} & y_{k3}x_{k2} & y_{k3}x_{k3} \end{pmatrix}.$$

Retrieval is as follows

$$\begin{aligned} M x_k &= \begin{pmatrix} y_{k1}x_{k1} & y_{k1}x_{k2} & y_{k1}x_{k3} \\ y_{k2}x_{k1} & y_{k2}x_{k2} & y_{k2}x_{k3} \\ y_{k3}x_{k1} & y_{k3}x_{k2} & y_{k3}x_{k3} \end{pmatrix} \begin{pmatrix} x_{k1} \\ x_{k2} \\ x_{k3} \end{pmatrix} \\ &= (x_{k1}^2 + x_{k2}^2 + x_{k3}^2) \begin{pmatrix} y_{k1} \\ y_{k2} \\ y_{k3} \end{pmatrix} \\ &= \begin{pmatrix} y_{k1} \\ y_{k2} \\ y_{k3} \end{pmatrix} \quad \text{if } (x_{k1}^2 + x_{k2}^2 + x_{k3}^2) = 1. \end{aligned}$$

If the stimulus is not x_k but is x'_k , a distorted form of x_k , then the response is

$$\begin{aligned} M x'_k &= y_k x_k^t x'_k \\ &= (x_k^t \cdot x'_k) y_k. \end{aligned} \quad (\text{A.14})$$

So, y_k will be recovered in un-distorted form with amplitude $(x_k^t \cdot x'_k)$.

In case of a set of associated pairs, the associative memory is formed as

$$M = \sum_k y_k x_k^t. \quad (\text{A.15})$$

Recall of y_m on presentation of stimulus x_m is obtained as

$$\begin{aligned} Mx_m &= \left(\sum_k y_k x_k^t \right) x_m \\ &= (x_m^t \cdot x_m) y_m + \sum_{k \neq m} (x_k^t \cdot x_m) y_k \\ &= y_m \end{aligned} \quad (\text{A.16})$$

where $(x_k^t \cdot x_m) = \delta_{km}$, is the Kronecker delta. From equation (A.16) we notice, a perfect recall is possible if the stored patterns constitute a set of *orthogonal basis vectors* spanning the N_1 dimensional space. For such patterns $(x_k^t \cdot x_m) = 0$ when $k \neq m$ and $(x_m^t \cdot x_m) = 1$.

If the actual stimulus is a distorted version of the original stored pattern, that is if we have x'_m instead of x_m then the regenerated response will be a degraded version of y_m .

Hopfield's model of *ANN* acts as an associative memory storing the patterns in the above fashion. The model consists of completely connected network (schematic representation is given in Figure A.3) of idealized neurons. Here the cross products of the input patterns (vectors) are represented as the weights linking two nodes (whereas in matrix associative memory it becomes the elements of the matrix). The details of the model will be given in the paragraphs that follow. In the associative memory we can store and retrieve several patterns. In case of a network or physical system, the stored patterns can be represented as locally stable states of the system. So any physical system whose dynamics in the state space is dominated by a substantial number of locally stable states can be regarded as an associative memory/content addressable memory (*CAM*).

Consider a physical system described by a vector X . Let the system have locally stable points $X_a, X_b, \dots, X_n$. Then, if the system is started sufficiently close to any point X_a , say at $X_a + \Delta$, it will proceed in time until $X = X_a$. We can regard the information stored in the system as the vectors $X_a, X_b, \dots, X_n$.

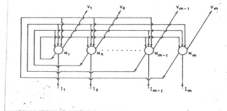


Figure A.3: Hopfield's neural network model

The starting point $X = X_a + \Delta$, represents a partial knowledge of the item X_a , and the system then generates the actual information X_a . The dynamics of such a network governing the states and the status updating rules can either be discrete or continuous as described below.

Case 1 : Discrete model

In this model two state neurons are used. Each neuron i , has two possible states, characterized by the output V_i of the neuron having values -1 or $+1$. The input to a neuron comes from two sources, external input bias I_i and inputs from other neurons (to which it is connected). The total input to the i th neuron is

$$U_i = \sum_j W_{ij} V_j + I_i \quad (\text{A.17})$$

where W_{ij} is the synaptic interconnection strength (weight) from neuron i to j .

Useful computations in this system involve change of state of the system with time. The motion of the state of the neural network in state space describes the computation that it performs. The present model describes it in terms of a stochastic evolution. Each neuron samples its input at any random time. It changes the value of its output or leaves it fixed according to a threshold rule with a threshold θ_i . The rule is

$$\begin{aligned} V_i &\longrightarrow -1 \quad \text{if } U_i \leq \theta_i \\ &\longrightarrow 1 \quad \text{if } U_i > \theta_i. \end{aligned} \quad (\text{A.18})$$

The interrogation of each neuron is a stochastic process, taking place at a fixed mean rate for each neuron. The times of interrogation of each neuron are independent of that of others thereby making the algorithm asynchronous.

The objective function (called the energy of the network) representing the overall status of the network is taken as

$$E = - \sum_i \sum_j W_{ij} V_i V_j - \sum_i I_i V_i + \sum_i \theta_i V_i. \quad (\text{A.19})$$

For a particular set up (fixed I_i, W_{ij}, θ_i), the energy E is a function of the status (V) of the nodes. The local minima of this function correspond to the stable states of the network. In this respect, the energy function E is a Liapunov function. A function $E(V)$ is said to be a Liapunov function in V^* if it satisfies the following properties :

There exists a δ such that if $O(V^*, \delta)$ be an open neighborhood of V^* of radius δ , we have

- i : $E(V) > E(V^*), \forall V \in O(V^*, \delta), V \neq V^*$,
- ii : $\frac{dE}{dt}(V^*) = 0$, and
- iii : $\frac{dE}{dt}(V) < 0, \forall V \in O(V^*, \delta), V \neq V^*$.

If the state V_i of the neuron i is changed to $V_i + \Delta V_i$ by the thresholding function (equation A.18), then the change ΔE in the energy E (equation A.19) is

$$\Delta E = - \left[\sum_j W_{ij} V_j + I_i - \theta_i \right] \Delta V_i. \quad (\text{A.20})$$

Now if ΔV_i is positive then from the thresholding rule we notice that the bracketed quantity in equation (A.20) is also positive, making ΔE negative. Similarly, when ΔV_i is negative, the bracketed quantity becomes negative resulting again ΔE negative. Thus any change in E under the previous thresholding rule (equation A.18) is negative. Since E is bounded, the time evolution of the system is a motion in the state space that seeks out minima in E and stops at such points. Given any initial state, the system will automatically update its status and will reach a stable state. The condition which guarantees convergence flow

$$W_{ij} = W_{ji} \text{ and } W_{ii} = 0;$$

i.e., the weight matrix is symmetric and zero diagonal.

The operation of the network is as follows. First give the exemplars for all pattern classes. During this the weights between different pairs of neurons get adjusted. Then an unknown pattern is imposed on the net at time zero. Following this initialization, the net iterates in discrete time steps using equation (A.18). The network is considered to have converged when the outputs no longer change on successive iterations. The pattern specified by the node outputs after convergence is the actual output. The above procedure can be written in algorithmic form as follows.

Step 1 : Assign connection weights as

$$W_{ij} = \begin{cases} \sum_{s=1}^M x_i^s x_j^s & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}$$

W_{ij} is the connection strength between node i and j . x_i^s is the i th component of the s th pattern (assuming there are M N -dimensional patterns).

Step 2 : Present the next unknown pattern,

$$V_i(0) = x_i, \quad 1 \leq i \leq N.$$

Here $V_i(t)$ is the output of node i at time index t .

Step 3 : Iterate until convergence

$$V_i(t+1) = g\left(\sum_{j=1, j \neq i}^N W_{ij} V_j(t)\right).$$

The function $g(\cdot)$ is the thresholding function (equation A.18). At the converged stage the node outputs give the exemplar pattern that best matches the unknown pattern.

Step 4 : Repeat by going to *Step 2*.

Case 2 : Continuous model

This model is based on continuous variables and responses but retains all the

significant behaviors of the discrete model. Let the output variable V_i for the i th neuron lie in the range $[-1, +1]$, and be continuous and monotonically increasing non linear function of the instantaneous input U_i to it. The typical input/output relation

$$V_i = g(U_i) \quad (\text{A.21})$$

is usually sigmoidal with asymptotes -1 and $+1$. It is also necessary that the inverse of g exists i.e., $g^{-1}(V_i)$ is defined.

A typical choice of the function g is [Figure A.4]

$$g(x) = 2 \frac{1}{1 + e^{-(x-\theta)/\theta_0}} - 1. \quad (\text{A.22})$$

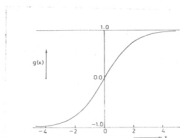


Figure A.4: Sigmoidal activation function

Here the parameter θ controls the shifting of the function g along the x axis and θ_0 determines the steepness (sharpness) of the function. Positive values of θ shift $g(x)$ towards positive x axis and vice versa. Lower values of θ_0 make the function steeper while higher values make it less sharp. The value of $g(x)$ lies in $[-1, 1]$ with 0.0 at $x = \theta$. Another commonly used form of the function g is

$$g(x) = \tanh\left(\frac{x - \theta}{\theta_0}\right). \quad (\text{A.23})$$

For the realization of the gain function g let us use a non-linear operational amplifier with negligible response time. A possible electrical circuit for the realization of a neuron is shown in Figure A.2. In the diagram $V_1, V_2, \dots, V_n$ represent the output voltage (status) of the amplifiers (neurons) to which the

i th amplifier is connected. $R_{ij}(=1/W_{ij})$ is the connecting resistance between the i th and the j th amplifiers. I_i is the input bias current for the i th amplifier. R is the input resistance of the amplifier and C the input capacitance of the same.

Suppose U_i is the total input voltage (potential at the point A , Figure A.2) to the amplifier having a gain g . Then applying Kirchoff's current law at A we get

$$\begin{aligned} \sum_{j=1}^n \frac{V_j - U_i}{R_{ij}} + I_i &= \frac{U_i}{R} + C \frac{dU_i}{dt} \\ \text{or, } C \frac{dU_i}{dt} &= \sum_{j=1}^n \frac{V_j}{R_{ij}} - \sum_{j=1}^n \frac{U_i}{R_{ij}} + I_i - \frac{U_i}{R} \\ &= \sum_{j=1}^n \frac{V_j}{R_{ij}} - \frac{U_i}{R_i} + I_i \quad \left[\frac{1}{R_i} = \sum_{j=1}^n \frac{1}{R_{ij}} + \frac{1}{R} \right] \\ &= \sum_{j=1}^n W_{ij} V_j - \frac{U_i}{R_i} + I_i. \end{aligned} \quad (\text{A.24})$$

Here R_i is the total input resistance of the amplifier. The output impedance of the amplifier is considered to be negligible.

Now let us consider the quantity

$$E = - \sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \sum_i \frac{1}{R_i} \int_0^{V_i} g^{-1}(V) dV. \quad (\text{A.25})$$

Now,

$$\begin{aligned} \frac{dE}{dt} &= \sum_i \frac{\partial E}{\partial V_i} \frac{dV_i}{dt} \\ &= \sum_i \left(- \sum_j W_{ij} V_j - I_i + \frac{U_i}{R_i} \right) \frac{dV_i}{dt} \quad (\text{from equation A.25}) \\ &= \sum_i \left(-C \frac{dU_i}{dt} \right) \frac{dV_i}{dt} \quad (\text{from equation A.24}) \\ &= - \sum_i \left\{ C(g^{-1}(V_i))' \left(\frac{dV_i}{dt} \right)^2 \right\} \left[\text{as, } \frac{dU_i}{dt} = (g^{-1}(V_i))' \frac{dV_i}{dt}, \text{ from (A.21)} \right]. \end{aligned}$$

Since g is a monotonic increasing function and C is a positive constant, every term in $\{.\}$ of the above expression is non-negative, and hence, $\frac{dE}{dt} \leq 0$, and $\frac{dE}{dt} = 0 \implies \frac{dV_i}{dt} = 0, \forall i$.

The function E is thus seen to be a Liapunov function. The function is also bounded. As $\frac{dE}{dt} \leq 0$, the searching in the direction of gradient will lead us to a minimum value of E . So the evolution of the system is a motion in the state space that seeks out minima in E and stops at such points. Thus here also we can infer that given any initial state, the system will automatically update its status and will reach a stable state. The last term in equation (A.25) is the energy loss term of the system. For a high gain of g (the input output transfer function, equation A.22), this term vanishes.

The network model with continuous dynamics is mainly used for optimization of functions. For this, only thing one has to do is to establish a correspondence between the energy function of a suitable network and the function to be optimized. In such an optimization task it is likely that one will be trapped in some local optimum. To come out of that, simulated annealing [291] can be used i.e., run the network once again by giving perturbation from that local stable state. By repeatedly doing this one can, possibly, obtain the global optimum. Hopfield and Tank demonstrated an application of this model to solve computationally hard optimization problems in [139], in particular they applied it to solve the *traveling salesman problem*. Applications of this model in image processing/pattern recognition problems can be found in [174, 196, 199, 207].

This network has two major drawbacks when used as a *CAM*. First, the number of patterns that can be stored and accurately retrieved is severely limited. If too many patterns are stored, the network may converge to a spurious pattern different from the exemplar pattern. The number of patterns that can be stored and retrieved perfectly is called the capacity of the network. The asymptotic capacity of such a network [294] is $\frac{n}{4 \log n}$, where n is the number of nodes. A second limitation of this model is that an exemplar pattern will be unstable (i.e., if it is applied at time zero it will converge to some other exemplar) if it shares many bits in common with another exemplar pattern, i.e., the input patterns are not orthogonal. This shows that the orthogonality assumption of the storing patterns is a must (as in the matrix associative memory) for perfect retrieval of the stored pattern.

An extension of this associative memory is made by Kosko [148, 149]. He

introduced the concept of bi-directionality, forward and backward associative search. It was shown that any real matrix representing the connection strength or weights of a network can be considered as a bi-directional associative memory. This overcomes the symmetricity (of the weight matrix) assumption of Hopfield's model. Kosko's model is basically a two layered non-linear feed back dynamical system. In contrast to Hopfield's model (which acts as auto-associator) the present model acts as hetero associative *CAM* storing and recalling different pattern vectors.

A.3.4 Kohonen's model

Self-organization refers to the ability to find out the structure in the input data even when there is no prior information available about the structure. For example, if we look at a picture, we can very easily distinguish which portion of it belongs to object and which portion to background. Though we did not have any prior knowledge of the structure of the object, we can find it out because of the self organizing ability of our nervous system. Kohonen's self-organizing network consists of a single layer of neurons as shown in Figure A.5. The neurons, however, are highly interconnected within the layer as well as with the outside world.

see Figure 4.1

Figure A.5: Kohonen's model of neural network

Let the signal

$$\vec{U} = \{u_1, u_2, \dots, u_n\}' \in R^n \quad (\text{A.26})$$

be connected as input to all the processing units. Let the length of the input vector be fixed (to some constant length) and the unit (processor/neuron) i have the weights (chosen as random numbers)

$$\vec{W}_i = \{w_{i1}, w_{i2}, \dots, w_{in}\} \quad (\text{A.27})$$

where, w_{ij} is the weight of the i th unit from j th component of the input. The i th unit then forms the discriminant function

$$\eta_i = \sum_{j=1}^n w_{ij} u_j = \vec{W}_i \cdot \vec{U} \quad (\text{A.28})$$

where, η_i is the output of that unit. A discrimination mechanism will then detect the maximum of η_i . Let

$$\eta_k = \max_i \{\eta_i\}. \quad (\text{A.29})$$

For the k th unit and all of its neighbors (within a specified radius of r , say) the following adaptation rule

$$\vec{W}_k(t+1) = \frac{\vec{W}_k(t) + \alpha \vec{U}(t)}{\|\vec{W}_k(t) + \alpha \vec{U}(t)\|} \quad (\text{A.30})$$

is applied. Here the variables have been labeled by a discrete time index t . α ($0 < \alpha < 1$) is a gain parameter that decreases with time and the denominator is the length of the weight vector (which is used only for normalization of the weight vector). Note that the process does not change the length of $\vec{W}_k$, rather rotates $\vec{W}_k$ towards $\vec{U}$. Note that the decrement of α with time guarantees the convergence of the adaptation of the weights when sufficient input vectors are presented. This type of learning is known as *competitive learning* (*regularity detector/unsupervised learning/clustering*). Here all the processors try to match with the input signal, but the one which matches most (winner) takes the benefit of updating the weights (sharing with its neighbors). The basic idea is winner takes all. In this context it may be mentioned that sometimes the learning algorithm is modified in a way such that the nodes which had won the competition for a large number of times (greater than some fixed positive number) are not allowed to take part in the competition.

The concept can be written in algorithmic form as follows :

Step 1 : Initialize weights of the neurons to random values.

Step 2 : Present new input.

Step 3 : Compute match η_i between the input and each output node i using equation (A.28).

Step 4 : Select node i^* as the output node with maximum η_i .

Step 5 : Update weights to node i^* and that of its neighbors using equation (A.30).

Step 6 : Repeat by going to *Step 2*.

When sufficient number of inputs are presented, the weights will be stabilized and clusters will be formed. The basic points to be noted here are :

- i : Real valued inputs are presented.
- ii : After enough input vectors have been presented, weights will specify clusters.
- iii : The weights will organize in such a fashion that the topologically close nodes/neurons/processors will be sensitive to inputs that are close in the feature space.
- iv : The weight vectors are normalized to have constant length.
- v : A set of neurons (may differ from cluster to cluster and from network to network) is allocated for a cluster.

The algorithm for feature map formation requires a neighborhood to be defined around the neurons. The size of the neighborhood normally decreases with time.

A.3.5 Multi-layer perceptron

In the last two sections we have described two *ANN* architectures which employ auto-associative learning and regularity detector mechanisms for their operation. In the present section we shall be describing another *ANN* architecture which employ hetero-associative learning and acts as a supervised classifier.

One of the most exciting developments during the early days of pattern recognition was the **perceptron** [133]. It may be defined as a network of elementary processors arranged in a manner reminiscent of biological neural nets which will be able to learn how to recognize and classify patterns in an autonomous manner. In such a system the processors are simple linear elements arranged in

one layer. This classical (single layer) perceptron, given two classes of patterns, attempts to find a linear decision boundary separating the two classes. If the two sets of patterns are linearly separable, the perceptron algorithm is guaranteed to find a separating hyper-plane in a finite number of steps. However, if the pattern space is not linearly separable, the perceptron fails and it is not known when to terminate the algorithm in order to get a reasonably good decision boundary. Keller and Hunt [231] attempted to provide a good stopping criterion (may not estimate a good decision boundary) for linearly non separable classes by incorporating the concept of fuzzy set theory into the learning algorithm of the perceptron. Thus, a single layer perceptron is inadequate for situations with multiple classes and non-linear separating boundaries. This motivated the invent of a multi-layer network with non-linear learning algorithms known as the **multi-layer perceptron (MLP)** [134].

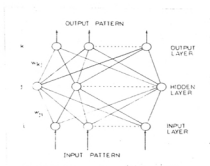


Figure A.6: Schematic representation of the multi-layer perceptron

A schematic representation of the multi layer perceptron (*MLP*) is given in Figure A.6. In general, such a network is made up of sets of nodes arranged in layers. Nodes of two consecutive layers are connected by links or weights, but there is no connection among the elements of the same layer. The layer where the inputs are presented is known as *input layer*. Similarly, the layer producing output is called *output layer*. The layers in between the input and the output layers are known as *hidden layers*. The output of nodes in one layer are transmitted to nodes in another layer via links/connections that amplify or attenuate or inhibit such outputs through weighting factors. Except for the input layer nodes, the total input to each node is the sum of weighted outputs

of the nodes in the prior layer. Each node is activated in accordance with the input to the node and the activation function of the node. The total input to the i th unit of any layer is

$$U_i = \sum_j W_{ij} V_j. \quad (\text{A.31})$$

Here V_j is the output of the j th unit of the previous layer and W_{ij} is the connection weight between the i th node of one layer and the j th node of the previous layer. The output of a node i is $V_i = g(U_i)$, g is the activation function. Mostly the activation function is sigmoidal, with the form [Figure A.7]

$$g(x) = \frac{1}{1 + e^{-(x-\theta)/\theta_0}}. \quad (\text{A.32})$$

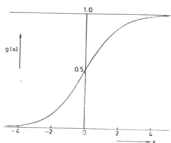


Figure A.7: Activation function for the multi-layer perceptron

In the learning phase (training) of such a network we present a pattern $\vec{X} = \{x_k\}$, where x_k is the k th component of the vector $\vec{X}$, as input and ask the net to adjust its set of weights in the connecting links and also the thresholds in the nodes such that the desired outputs $\{t_k\}$ are obtained at the output nodes. After this we present another pair of $\vec{X}$ and $\{t_k\}$, and ask the net to learn that association also. In fact, we desire the net to find a simple set of weights and biases that will be able to discriminate among all the input/output pairs presented to it. This process can pose a very strenuous learning task and is not always readily accomplished. Here the desired output $\{t_k\}$ basically acts as a teacher which tries to minimize the error. In this context it should be mentioned that such a procedure will always be successful if the pattern is learnable by the *MLP*.

In general, the outputs $\{V_k\}$ will not be the same as the target or desired values $\{t_k\}$. For a pattern vector, the error is

$$E = \frac{1}{2} \sum_k (t_k - V_k)^2 \quad (\text{A.33})$$

where the factor of one half is inserted for convenience of mathematical simplification. The procedure for learning the correct set of weights is to vary the weights in a manner calculated to reduce the error E as rapidly as possible. This can be achieved by moving in the direction of negative gradient of E . In other words the incremental change ΔW_{kj} can be taken as proportional to $-\frac{\partial E}{\partial W_{kj}}$ for a particular pattern, i.e., $\Delta W_{kj} = -\eta \frac{\partial E}{\partial W_{kj}}$. However, E , the error is expressed in terms of the outputs $\{V_k\}$, each of which is a non-linear function of the total input to the node. That is,

$$V_k = g(U_k) \quad (\text{A.34})$$

where $U_k = \sum_j W_{kj} V_j$. Hence $\frac{\partial U_k}{\partial W_{kj}} = V_j$.

Now

$$\begin{aligned} \Delta W_{kj} &= -\eta \frac{\partial E}{\partial W_{kj}} = -\eta \frac{\partial E}{\partial U_k} \frac{\partial U_k}{\partial W_{kj}} = -\eta \frac{\partial E}{\partial U_k} V_j \\ &= -\eta \frac{\partial E}{\partial V_k} \frac{\partial V_k}{\partial U_k} V_j = -\eta \frac{\partial E}{\partial V_k} g'(U_k) V_j. \end{aligned} \quad (\text{A.35})$$

Now if the k th element comes from the output layer then

$$\frac{\partial E}{\partial V_k} = -(t_k - V_k) \quad [\text{from equation A.33}],$$

and thus the change in weight is given by

$$\Delta W_{kj} = \eta (t_k - V_k) g'(U_k) V_j. \quad (\text{A.36})$$

If the weights do not affect the output nodes directly [for links between the input and the hidden layer, and also between two consecutive hidden layers], the factor $\frac{\partial E}{\partial V_k}$ can not be calculated directly. In this case we use the chain rule to write

$$\frac{\partial E}{\partial V_k} = \sum_m \frac{\partial E}{\partial U_m} \frac{\partial U_m}{\partial V_k} = \sum_m \frac{\partial E}{\partial U_m} \frac{\partial}{\partial V_k} \sum_i W_{mi} V_i = \sum_m \frac{\partial E}{\partial U_m} W_{mk}. \quad (\text{A.37})$$

Hence

$$\Delta W_{kj} = \begin{cases} -\eta(\frac{\partial E}{\partial V_k})g'(U_k)V_j \\ -\eta(\sum_m \frac{\partial E}{\partial U_m}W_{mk})g'(U_k)V_j \end{cases} \quad (\text{A.38})$$

for the output layer and other layers respectively.

It may be mentioned here that a large value of η corresponds to rapid learning but might result in oscillation. A *momentum* term of $\alpha\Delta W_{kj}(t)$ can be added to avoid this and thus expression (A.35) can be modified as

$$\Delta W_{kj}(t+1) = -\eta\frac{\partial E}{\partial V_k}g'(U_k)V_j + \alpha\Delta W_{kj}(t) \quad (\text{A.39})$$

where the quantity $(t+1)$ is used to indicate the $(t+1)$ th time instant, and α is a proportionality constant. The second term is used to ensure that the change in W_{kj} at $(t+1)$ th instant should be somewhat similar to the change undertaken at instant t .

During training, each pattern of the training set is used in succession to clamp the input and output layers of the network. Feature values of the input patterns are clamped to the input nodes whereas the output nodes are clamped to class labels. The input is then passed on to the output layer via the hidden layers to produce output. Error is then calculated using equation (A.33) and is back propagated to modify the weights. A sequence of forward and backward passes is made for each input pattern. A set of training samples is used several times for the stabilization (minimum error) of the network. After the network has been settled down the weights specify the boundaries of the classes. The learning of this type of network is therefore supervised. Since the error is back propagated for learning the parameters (weight correction) the process is also called back propagation learning. Next phase is testing/recognition. Here we give an unknown input and the neuron with maximum output value will indicate the class to which it belongs.

The problem with this type of network is that it may get stuck to local optima and the training process is very slow. A solution to overcome the first problem is recently suggested in [295] and a new fast learning algorithm described in [296]. Applications of this model can be found in [166, 167, 176, 183, 184, 185]

It may be mentioned here that at present researchers are trying to fuse the concepts of fuzzy set theory and artificial neural network technologies in order to exploit their merits for designing more efficient systems. A discussion on it is given in Chapter 1 (section 1.4, pp. 31).

Bibliography

- [1] R. C. Gonzalez and P. Wintz, *Digital Image Processing*. Reading, MA: Addison-Wesley, 1987.
- [2] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*. New York: Academic Press, 1982.
- [3] W. K. Pratt, *Digital Image Processing*. New York: Wiley, 1978.
- [4] E. L. Hall, *Computer Image Processing and Recognition*. New York: Academic Press, 1979.
- [5] J. M. S. Prewitt, "Object enhancement and extraction," in *Picture Processing and Psycho-Pictorics*, (B. S. Lipkin and A. Rosenfeld, eds.), New York: Academic Press, 1970.
- [6] S. K. Pal and R. A. King, "Image enhancement using smoothing with fuzzy sets," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, pp. 494-501, 1981.
- [7] M. K. Kundu and S. K. Pal, "Automatic selection of object enhancement operator with quantitative justification based on fuzzy set theoretic measure," *Pattern Recognition Letters*, vol. 11, pp. 811-829, 1990.
- [8] D. C. C. Wang, A. H. Vagnucii, and C. C. Li, "Digital image enhancement : a survey," *Computer Vision, Graphics and Image Processing*, vol. 24, pp. 368-381, 1983.
- [9] R. E. Wood and R. C. Gonzalez, "Real time digital enhancement," *Proceedings of the IEEE*, vol. 69, pp. 643-654, 1981.

- [10] J. B. Zimmerman *et al.*, "An evaluation of adaptive histogram equalization for contrast enhancement," *IEEE Transactions on Medical Imaging*, vol. 7, pp. 304-312, 1988.
- [11] H. Li and H. S. Yang, "Fast and reliable image enhancement using fuzzy relaxation," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, pp. 1276-1286, 1989.
- [12] W. Frei, "Image enhancement by histogram hyperbolization," *Computer Vision, Graphics and Image Processing*, vol. 6, pp. 286-294, 1977.
- [13] "Special issue on image restoration and enhancement," *IEE Proceedings, Part E*, 1980.
- [14] H. C. Andrew and B. R. Hunt, *Digital Image Restoration*. New York: Prentice-Hall, 1977.
- [15] E. C. Hildreth, "The detection of intensity changes by computer and biological vision system," *Computer Vision, Graphics and Image Processing*, vol. 22, pp. 1-27, 1983.
- [16] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society, London*, vol. B-207, pp. 187-217, 1980.
- [17] D. Marr, *Vision*. New York: W. H. Freeman and Company, 1982.
- [18] M. K. Kundu and S. K. Pal, "Thresholding for edge detection using human psychovisual response," *Pattern Recognition Letters*, vol. 4, pp. 433-441, 1986.
- [19] L. S. Davis, "A survey of edge detection techniques," *Computer Vision, Graphics and Image Processing*, vol. 4, pp. 248-270, 1975.
- [20] P. Meer, W. Wang, and H. Wechsler, "Edge detection by associative mapping," *Pattern Recognition*, vol. 22, pp. 491-503, 1989.
- [21] R. M. Haralick, "Digital step edge from zero crossing of second directional derivative," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 58-68, 1984.

- [22] J. F. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679-698, 1986.
- [23] F. Bergholm, "Detection of edges and busy regions using Canny's edge detector," Tech. Rep. TRITA-NA-E8472, Royal Institute of Technology, Stockholm, Sweden, 1987.
- [24] J. F. Haddon, "Generalized threshold selection for edge detection," *Pattern Recognition*, vol. 21, pp. 195-203, 1988.
- [25] T. Pavlidis, *Algorithms for Graphics and Image Processing*. New York, Berlin: Springer-Verlag, 1982.
- [26] S. Peleg and A. Rosenfeld, "A min-max medial axis transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 208-210, 1981.
- [27] T. V. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communication of the Association for Computing Machinery*, vol. 27, pp. 236-239, 1984.
- [28] R. W. Hall, "Fast parallel thinning algorithms : parallel speed and connectivity preservation," *Communication of the Association for Computing Machinery*, vol. 32, pp. 124-131, 1989.
- [29] E. Salari and P. Siy, "The ridge-seeking method for obtaining the skeleton of digital images," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 14, pp. 524-528, 1984.
- [30] C. Ronse, "A topological characterization of thinning," *Theoretical Computer Science*, vol. 43, pp. 31-41, 1986.
- [31] Y. S. Chen and W. H. Hsu, "A comparison of some one pass thinning algorithms," *Pattern Recognition Letters*, vol. 11, pp. 35-41, 1990.
- [32] S. K. Pal, "Fuzzy skeletonization of images," *Pattern Recognition Letters*, vol. 10, pp. 17-23, 1990.

- [33] A. K. Jain, "Image data compression : a review," *Proceedings of the IEEE*, vol. 69, pp. 349-389, 1981.
- [34] M. Kunt, A. Ikonomopoulos, and K. Kocher, "Second generation image coding techniques," *Proceedings of the IEEE*, vol. 73, pp. 549-574, 1985.
- [35] A. N. Netravali and J. Limb, "Picture coding : a review," *Proceedings of the IEEE*, vol. 68, pp. 366-406, 1980.
- [36] M. Kunt, I. K. Benard, and R. Leonardi, "Recent results in high compression image coding," *Proceedings of the IEEE*, vol. 34, pp. 1106-1137, 1987.
- [37] M. K. Kundu, *On Image Data Compression and Related Preprocessing Techniques*. PhD thesis, Calcutta University, Calcutta, 1991.
- [38] K. S. Fu, *Syntactic Pattern Recognition and Applications*. London: Academic Press, 1982.
- [39] T. Pavlidis, *Structural Pattern Recognition*. New York: Springer-Verlag, 1977.
- [40] L. S. Davis, "Understanding shape : angles and sides," *IEEE Transactions on Computer*, vol. 26, pp. 236-242, 1977.
- [41] H. Samet, "Connected component labeling using quadtrees," *Journal of the Association of Computing Machinery*, vol. 28, pp. 487-501, 1981.
- [42] K. S. Fu and T. L. Both, "Grammatical inference : introduction and survey," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 5, pp. 95-111, 409-423, 1975.
- [43] R. C. Gonzalez and M. G. Thomson, *Syntactic Pattern Recognition : An Introduction*. Reading, MA: Addison-Wesley, 1978.
- [44] A. Rosenfeld, *Picture Languages*. New York: Academic Press, 1981.
- [45] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: John Wiley, 1973.

- [46] J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*. Reading, MA: Addison-Wesley, 1974.
- [47] D. H. Ballard and C. M. Brown, *Computer Vision*. Englewood Cliffs, New Jersey: Prentice-Hall, 1982.
- [48] K. S. Fu and J. K. Mui, "A survey on image segmentation," *Pattern recognition*, vol. 13, pp. 3-16, 1981.
- [49] R. M. Haralick and L. Shapiro, "Survey of image segmentation techniques," *Computer Vision, Graphics and Image Processing*, vol. 29, pp. 100-132, 1985.
- [50] P. K. Shao, S. Soltani, A. K. C. Wong, and Y. C. Chen, "A survey of thresholding techniques," *Computer Vision, Graphics and Image Processing*, vol. 41, pp. 233-260, 1988.
- [51] T. Taxt, P. J. Flynn, and A. K. Jain, "Segmentation of document images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 1322-1329, 1989.
- [52] N. Otsu, "A threshold selection method from gray-level histogram," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, pp. 62-66, 1979.
- [53] Y. Nakagawa and A. Rosenfeld, "Some experiments on variable thresholding," *Pattern Recognition*, vol. 11, pp. 191-204, 1979.
- [54] J. Kittler and J. Illingworth, "Minimum error thresholding," *Pattern Recognition*, vol. 19, pp. 41-47, 1986.
- [55] T. Pun, "Entropic thresholding : a new approach," *Computer Vision, Graphics and Image Processing*, vol. 9, pp. 210-239, 1981.
- [56] J. N. Kapur, P. K. Shao, and A. K. C. Wong, "A new method for gray level picture thresholding using the entropy of histogram," *Computer Vision, Graphics and Image Processing*, vol. 29, pp. 273-285, 1985.
- [57] A. K. C. Wong and P. K. Sahoo, "A gray level threshold selection method based on maximum entropy principle," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 4, pp. 866-871, 1989.

- [58] M. D. Levine and A. M. Nazif, "Dynamic measurement of computer generated image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 7, no. 2, pp. 155-164, 1985.
- [59] N. R. Pal and S. K. Pal, "Image model, Poisson distribution and object extraction," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 459-483, 1991.
- [60] J. S. Weska, "Survey of threshold selection techniques," *Computer Vision, Graphics and Image Processing*, vol. 7, pp. 259-265, 1978.
- [61] F. Deravi and S. K. Pal, "Gray level thresholding using second order statistics," *Pattern Recognition Letters*, vol. 1, pp. 417-422, 1983.
- [62] S. K. Pal and N. R. Pal, "Segmentation based on measures of contrast, homogeneity and region size," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 17, pp. 857-868, 1987.
- [63] N. R. Pal, *On Image Information Measures and Object Extraction*. PhD thesis, Indian Statistical Institute, Calcutta, 1990.
- [64] N. R. Pal and S. K. Pal, "Entropic thresholding," *Signal Processing*, vol. 16, pp. 97-108, 1989.
- [65] A. S. Abutaleb, "Automatic thresholding of gray level pictures using two dimensional entropy," *Computer Vision, Graphics and Image Processing*, vol. 47, pp. 22-32, 1989.
- [66] C. K. Chow and T. Kaneko, "Automatic boundary detection of the left-ventricle from cineangiograms," *Computer Biomedical Research*, vol. 5, pp. 388-420, 1972.
- [67] S. D. Yanowitz and A. M. Bruckstein, "A new method for image segmentation," *Computer Vision, Graphics and Image Processing*, vol. 46, pp. 82-95, 1989.
- [68] A. Rosenfeld, R. A. Hummel, and S. W. Zucker, "Scene labeling by relaxation operations," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, pp. 420-433, 1976.

- [69] R. A. Hummel and S. W. Zucker, "On the foundation of relaxation labeling process," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 3, pp. 267-287, 1983.
- [70] A. K. Jain, "Advances in mathematical models for image processing," in *Proceedings of the IEEE*, pp. 502-528, 1981.
- [71] F. R. Hanson and H. Elliott, "Image segmentation using Markov random field models," *Computer Vision, Graphics and Image Processing*, vol. 20, pp. 101-132, 1982.
- [72] H. Derin, H. Elliott, R. Cristi, and D. Geman, "Bayes smoothing algorithms for segmentation of binary images by Markov random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 707-720, 1984.
- [73] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721-741, 1984.
- [74] H. Derin and H. Elliott, "Modeling and segmentation of noisy and textured images using Gibbs random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 39-55, 1987.
- [75] M. Askar and H. Derin, "A recursive algorithm for Bayes solution of the smoothing problem," *IEEE Transactions on Automatic Control*, vol. 26, pp. 558-561, 1981.
- [76] M. Amadasun and R. A. King, "Low level segmentation of multispectral images via agglomerative clustering of uniform neighbors," *Pattern Recognition*, vol. 21, pp. 261-268, 1988.
- [77] R. Hoffman and A. K. Jain, "Segmentation and classification of range images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 608-620, 1987.
- [78] P. C. Besl and A. K. Jain, "Segmentation using variable order surface fitting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 167-192, 1988.

- [79] R. D. Rimey and F. S. Cohen, "A maximum likelihood approach to segmentation of range data," *IEEE Journal of Robotics and Automation*, vol. 4, pp. 277-287, 1988.
- [80] N. Yokoya and M. D. Levine, "Range image segmentation based on differential geometry : a hybrid approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 643-649, 1989.
- [81] T. Peli and D. Malah, "A study of edge detection algorithms," *Computer Vision, Graphics and Image Processing*, vol. 20, pp. 1-21, 1982.
- [82] Y. T. Zhou, V. Venkateswara, and R. Chellappa, "Edge detection and linear feature extraction using 2-D random field model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 84-95, 1989.
- [83] J. Kittler, J. Eagleton, J. Illingworth, and K. Paler, "An average edge detector," *Pattern Recognition Letters*, vol. 6, pp. 27-32, 1987.
- [84] Y. Lu and R. C. Jain, "Behavior of edges in scale space," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 337-356, 1989.
- [85] M. Gokmen and C. C. Li, "Edge detection with iteratively refined regularization," in *Proceedings International Conference on Pattern Recognition*, pp. 690-693, 1990.
- [86] S. E. Reichenbach, S. K. Park, and R. A. Gartenberg, "Optimal, small kernels for edge detection," in *Proceedings International Conference on Pattern Recognition*, pp. 57-63, 1990.
- [87] E. T. Lee, "Proximity measures for the classification of geometric figures," *Journal of Cybernetics*, vol. 2, pp. 43-59, 1972.
- [88] E. T. Lee, "Shape-oriented chromosome classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 5, pp. 629-632, 1975.
- [89] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338-353, 1965.

- [90] A. Kaufmann, *Fuzzy Subsets - Fundamental Theoretical Elements*. New York: Academic Press, 1980.
- [91] D. Dubois and H. Prade, *Fuzzy Sets and Systems : Theory and Applications*. New York: Academic Press, 1980.
- [92] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum Press, 1981.
- [93] S. K. Pal and D. D. Majumder, *Fuzzy Mathematical Approach to Pattern Recognition*. New York: John Wiley (Halsted Press), 1986.
- [94] A. Kandel, *Fuzzy Techniques in Pattern Recognition*. New York: Wiley Interscience, 1982.
- [95] A. Kandel, *Fuzzy Mathematical Techniques with Applications*. New York: Addison-Wesley, 1986.
- [96] G. J. Klir and T. Folger, *Fuzzy Sets, Uncertainty and Information*. Reading, MA: Addison-Wesley, 1989.
- [97] N. R. Pal and S. K. Pal, "Object background segmentation using a new definition of entropy," *IEE Proceedings, Part E*, pp. 284-295, 1989.
- [98] B. Kosko, "Fuzzy entropy and conditioning," *Information Sciences*, vol. 40, pp. 165-174, 1986.
- [99] A. Deluca and S. Termini, "A definition of a non probabilistic entropy in the setting of fuzzy set theory," *Information and Control*, vol. 20, pp. 301-312, 1972.
- [100] S. G. Loo, "Measures of fuzziness," *Cybernetics*, vol. 10, pp. 201-210, 1977.
- [101] W. X. Xie and S. D. Bedrosian, "An information measure for fuzzy sets," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 14, pp. 151-156, 1984.
- [102] A. Rosenfeld, "Fuzzy geometry of image subsets," *Pattern Recognition Letters*, vol. 2, pp. 311-317, 1984.

- [103] A. Rosenfeld, "Fuzzy geometry : an overview," in *Proceedings First IEEE Internal Conference on Fuzzy Systems*, (San Diego, USA), pp. 113–117, 1992.
- [104] W. Pedrycz, "Fuzzy sets in pattern recognition : methodology and methods," *Pattern Recognition*, vol. 23, no. 1/2, pp. 121–146, 1990.
- [105] J. C. Bezdek and S. K. Pal (eds.), *Fuzzy Models for Pattern Recognition : Methods that Search for Structures in Data*. New York: IEEE Press, 1992.
- [106] S. K. Pal, "Fuzzy sets in image processing and recognition," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 119–126, 1992.
- [107] R. Krishnapuram and J. M. Keller, "Fuzzy set theoretic approach to computer vision : an overview," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 135–142, 1992.
- [108] L. A. Zadeh, K. S. Fu, K. Tanaka, and M. Shimura (eds.), *Fuzzy Sets and Their Applications to Cognitive and Decision Process*. London: Academic Press, 1975.
- [109] T. L. Huntsberger, "Representation of uncertainty in low level vision," *IEEE Transactions on Computer*, vol. 23, no. 2, p. 145, 1986.
- [110] N. R. Pal and S. K. Pal, "A survey on image segmentation," *Pattern Recognition*, 1992 (communicated).
- [111] S. K. Pal, *Fuzzy Set Theory in Gray Tone Image Processing*. PhD thesis, Imperial College of Science and Technology, London, 1982.
- [112] T. L. Huntsberger, C. L. Jacobs, and R. L. Cannon, "Iterative fuzzy image segmentation," *Pattern Recognition*, vol. 18, no. 2, pp. 131–138, 1985.
- [113] J. M. Keller and C. L. Carpenter, "Image segmentation in the presence of uncertainty," *International Journal of Intelligent Systems*, vol. 5, pp. 193–208, 1990.

- [114] B. L. Deekshatulu, G. K. Rao, and R. Krishnan, "A quantitative index of information and a method of enhancement for digital images using fuzzy sets," in *Proceedings IEEE International Conference on Systems, Man, and Cybernetics*, (New Delhi, India), pp. 1105-1109, 1983.
- [115] S. K. Pal, R. A. King, and A. A. Hashim, "Automatic gray level thresholding through index of fuzziness," *Pattern Recognition Letters*, vol. 1, pp. 141-146, 1983.
- [116] S. K. Pal and A. Rosenfeld, "Image enhancement and thresholding by optimization of fuzzy compactness," *Pattern Recognition Letters*, vol. 7, pp. 77-86, 1989.
- [117] C. A. Murthy and S. K. Pal, "Fuzzy thresholding : mathematical framework, bound functions and weighted moving average," *Pattern Recognition Letters*, vol. 11, pp. 197-206, 1990.
- [118] N. R. Pal and S. K. Pal, "Higher order fuzzy entropy and hybrid entropy of a set," *Information Sciences*, vol. 61, no. 3, pp. 211-231, 1992.
- [119] S. K. Pal and A. Dasgupta, "Spectral fuzzy sets and soft thresholding," *Information Sciences*, 1992 (in press).
- [120] C. A. Murthy, S. K. Pal, and D. D. Majumder, "Correlation between two fuzzy membership functions," *Fuzzy Sets and Systems*, vol. 7, pp. 23-38, 1985.
- [121] W. X. Xie and S. D. Bedrosian, "Experimentally driven fuzzy membership function for gray level images," *Journal of Franklin Institute*, vol. 325, pp. 154-164, 1988.
- [122] S. K. Pal and A. Ghosh, "Image segmentation using fuzzy correlation," *Information Sciences*, vol. 62, no. 3, pp. 223-250, 1992.
- [123] R. L. Cannon, R. Dave, and J. C. Bezdek, "Efficient implementation of fuzzy c-means clustering algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 2, 1986.

- [124] M. Trivedi and J. C. Bezdek, "Low-level segmentation of aerial images with fuzzy clustering," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, no. 4, pp. 589–598, 1986.
- [125] M. Sugeno, *Theory of Fuzzy Integral and its Applications*. PhD thesis, Tokyo Institute of Technology, Japan, 1974.
- [126] R. L. Cannon, R. Dave, J. C. Bezdek, and M. Trivedi, "Segmentation of a thematic mapper image using the fuzzy c-means clustering algorithm," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 24, pp. 400–407, 1986.
- [127] J. M. Keller, H. Qui, and H. Tahani, "The fuzzy integral in image segmentation," in *Proceedings NAFIPS*, (New Orleans, LA), pp. 324–338, 1986.
- [128] L. O. Hall *et al.*, "A comparison of neural network and fuzzy clustering techniques in segmenting magnetic resonance images of the brain," *IEEE Transactions on Neural Networks*, 1992 (to appear).
- [129] E. Backer, *Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets*. PhD thesis, Delft University, Delft, The Netherlands, 1978.
- [130] S. K. Pal and R. A. King, "On edge detection of X-ray images using fuzzy sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 69–77, 1983.
- [131] V. Goetcherian, "From binary to gray tone image processing using fuzzy logic concepts," *Pattern Recognition*, vol. 12, pp. 7–15, 1980.
- [132] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas imminent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, 1943.
- [133] M. Minsky and S. Papert, *Perceptrons*. Cambridge, MA: MIT Press, 1969.
- [134] D. E. Rumelhart, J. McClelland, *et al.*, *Parallel Distributed Processing : Explorations in the Microstructure of Cognition*. Vol. 1, Cambridge, MA: MIT Press, 1986.

- [135] T. Kohonen, "An introduction to neural computing," *Neural Networks*, vol. 1, pp. 3-16, 1988.
- [136] T. Kohonen, *Self-organization and Associative Memory*. Berlin: Springer Verlag, 1989.
- [137] J. J. Hopfield, "Neural network and physical systems with emergent collective computational abilities," in *Proceedings National Academy of Science*, (USA), pp. 2554-2558, 1982.
- [138] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two state neurons," in *Proceedings National Academy of Science*, (USA), pp. 3088-3092, 1984.
- [139] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization of problems," *Biological Cybernetics*, vol. 52, pp. 141-152, 1985.
- [140] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Magazine*, pp. 3-22, 1987.
- [141] D. E. Rumelhart, J. McClelland, *et al.*, *Parallel Distributed Processing*. Vol. 2, Cambridge, MA: MIT Press, 1986.
- [142] K. Fukushima, "Neocognitron : a self-organizing multilayer neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193-202, 1980.
- [143] M. A. Arbib, *Brains, Machines and Mathematics*. New York: Springer-Verlag, 1987.
- [144] G. A. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer Vision, Graphics and Image Processing*, vol. 37, pp. 54-115, 1987.
- [145] S. Grossberg (ed.), *Neural Networks and Natural Intelligence*. Reading, MA: MIT Press, 1988.
- [146] Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*. New York: Addison-Wesley, 1989.

- [147] P. D. Wassermann, *Neural Computing : Theory and Practice*. New York: Van Nostrand Reinhold, 1990.
- [148] B. Kosko, "Adaptive bidirectional associative memories," *Applied Optics*, vol. 26, no. 23, pp. 4947-4960, 1987.
- [149] B. Kosko, *Neural Networks and Fuzzy Systems*. Englewood Cliffs, New Jersey: Prentice-Hall, 1992.
- [150] L. O. Chua and L. Yang, "Cellular neural network : theory," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1257-1272, 1988.
- [151] L. O. Chua and L. Yang, "Cellular neural networks : applications," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1273-1290, 1988.
- [152] B. M. Forrest *et al.*, "Neural network models," *Parallel Computing*, vol. 8, pp. 71-83, 1988.
- [153] J. Bruck and S. Sanz, "A study on neural networks," *International Journal of Intelligent Systems*, vol. 3, pp. 59-75, 1988.
- [154] G. A. Carpenter and S. Grossberg, "ART2 : self-organization of stable category recognition codes for analog input patterns," *Applied Optics*, vol. 26, no. 23, pp. 4919-4930, 1987.
- [155] G. A. Carpenter, "Neural network models for pattern recognition and associative memory," *Neural Networks*, vol. 2, pp. 243-257, 1989.
- [156] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART : fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks*, vol. 4, 1991.
- [157] B. L. Deekshatulu, B. Manna, and R. Krishnan, "Introduction to neural networks," *Journal of the Institute of Electronics and Telecommunication Engineers*, vol. 37, pp. 561-571, 1991.
- [158] A. Ghosh, N. R. Pal, and S. K. Pal, "Neural computing : an introduction and some applications," *Students' Journal of the Institute of Electronics and Telecommunication Engineers*, 1992 (accepted).

- [159] E. Deprit, "Implementing recurrent backpropagation on the connection machine," *Neural Networks*, vol. 2, pp. 295-314, 1989.
- [160] J. Diederich, "Connectionist recruitment learning," in *Proceedings Eighth European Conference on Artificial Intelligence*, pp. 351-356, 1988.
- [161] S. E. Fahlman and G. E. Hinton, "Connectionist architectures for artificial intelligence," *IEEE Transactions on Computer*, vol. 20, pp. 100-109, 1989.
- [162] G. E. Hinton, "Connectionist learning procedures," *Artificial Intelligence*, vol. 40, pp. 185-234, 1989.
- [163] E. Gelenbe, "Theory of random neural network model," in *Neural Networks : Advances and Applications*, (E. Gelenbe, ed.), North Holland: Elsevier Publishers B. V., 1991.
- [164] F. J. Pineda, "Generalization of backpropagation to recurrent neural networks," *Physical Review Letters*, vol. 59, no. 19, pp. 2229-2232, 1987.
- [165] F. J. Pineda, "Dynamics and architecture for neural computation," *Journal of Complexity*, vol. 4, pp. 216-245, 1988.
- [166] R. P. Lippmann, "Pattern classification using neural networks," *IEEE Communications Magazine*, pp. 47-64, 1989.
- [167] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets and classification," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, 1992.
- [168] S. I. Gallant, "Connectionist expert system," *Communication of the Association for Computing Machinery*, vol. 31, pp. 152-169, 1988.
- [169] E. Sanchez, "Fuzzy connectionist expert systems," in *Proceedings First International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 31-35, 1990.
- [170] J. J. Hopfield and D. W. Tank, "Computing with neural circuits : a model," *Science*, vol. 233, pp. 625-633, 1986.

- [171] G. V. Wilson and G. S. Pavley, "On the stability of the traveling salesman problem algorithm of Hopfield and Tank," *Biological Cybernetics*, vol. 58, pp. 63-70, 1988.
- [172] D. W. Tank and J. J. Hopfield, "Simple neural optimization networks : an A/D converter, signal decision circuits, and linear programming circuits," *IEEE Transactions on Circuits and Systems*, vol. 3, pp. 533-541, 1986.
- [173] M. Vidyasagar, "Improved neural networks for analog to digital conversion," in *Proceedings International Symposium on Intelligent Robotics*, (Bangalore, India), pp. 698-707, 1991.
- [174] B. K. Parsi and B. K. Parsi, "On problem solving with Hopfield neural networks," *Biological Cybernetics*, vol. 62, pp. 415-423, 1990.
- [175] S. C. Newton and S. Mitra, "An adaptive fuzzy system for control and clustering of arbitrary data patterns," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 363-370, 1992.
- [176] D. J. Burr, "Experiments on neural net recognition of spoken and written text," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 7, pp. 1162-1168, 1988.
- [177] J. A. Benediktsson, P. H. Swain, and O. K. Ersoy, "Neural network approach versus statistical methods in classification of multisource remote sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 28, no. 4, pp. 540-552, 1990.
- [178] T. Hryčej, "On the relationship between information processing in probabilistic and neural networks," in *Proceedings First International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 243-246, 1990.
- [179] C. H. Chen, "On the relationship between statistical pattern recognition and artificial neural networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 4, pp. 655-661, 1992.

- [180] A. Ghosh and S. K. Pal, "Self-organisation of neural network and object extraction," in *Proceedings Workshop on Signal Processing, Communications and Networking*, (Bangalore, India), pp. 241–246, 1990.
- [181] A. Ghosh, N. R. Pal, and S. K. Pal, "Object extraction using a self-organizing neural network," in *Proceedings International Symposium on Intelligent Robotics*, (Bangalore, India), pp. 686–697, 1991.
- [182] A. Ghosh, N. R. Pal, and S. K. Pal, "Neural network, Gibbs distribution and object extraction," in *Proceedings International Symposium on Intelligent Robotics*, (Bangalore, India), pp. 95–106, 1991.
- [183] J. Lee, R. C. Weger, S. K. Sengupta, and R. M. Welch, "A neural network approach to cloud classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 28, no. 5, pp. 846–855, 1990.
- [184] W. E. Blanz and S. L. Gish, "A connectionist classifier architecture applied to image segmentation," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 6, pp. 603–617, 1991.
- [185] N. Babaguchi, K. Yamada, K. Kise, and Y. Tezuku, "Connectionist model binarization," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 6, pp. 629–644, 1991.
- [186] G. W. Cottrel, P. Munro, and D. Zipser, "Image compression by back propagation : an exemplar of extensional programming," Tech. Rep. ICS-8702, University of California, San Diego, 1987.
- [187] G. W. Cottrel and P. Munro, "Principal component analysis of images via back propagation," *SPIE : Visual Communication and Image Processing*, vol. 1001, pp. 1070–1077, 1988.
- [188] S. P. Luttrell, "Image compression using a multilayer neural network," *Pattern Recognition Letters*, vol. 10, pp. 1–7, 1989.
- [189] G. L. Bilbro, M. White, and W. Synder, "Image segmentation with neuro-computers," in *Neural Computers*, (R. Eckmiller and C. V. D. Malsberg, eds.), Springer-Verlag, 1988.

- [190] C. T. Chen, E. C. Tsao, and W. C. Lin, "Medical image segmentation by a constraint satisfaction neural network," *IEEE Transactions on Nuclear Science*, vol. 38, no. 2, pp. 678-686, 1991.
- [191] A. Ghosh, N. R. Pal, and S. K. Pal, "Image segmentation using a neural network," *Biological Cybernetics*, vol. 66, no. 2, pp. 151-158, 1991.
- [192] A. Ghosh and S. K. Pal, "Neural network, self-organization and object extraction," *Pattern Recognition Letters*, vol. 13, no. 5, pp. 387-397, 1992.
- [193] A. Ghosh, N. R. Pal, and S. K. Pal, "Object extraction using Hopfield type neural network," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 6, no. 5, 1992 (in press).
- [194] A. Ghosh, N. R. Pal, and S. K. Pal, "Self-organization for object extraction using multilayer neural network and fuzziness measures," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, 1993 (in press).
- [195] A. Ghosh, N. R. Pal, and S. K. Pal, "On robustness of neural network based systems under component failure," *IEEE Transactions on Neural Networks*, 1992 (revised).
- [196] L. Bedini and A. Tonazzini, "Neural network use in maximum entropy image restoration," *Image and Vision Computing*, vol. 8, pp. 108-114, 1990.
- [197] Y. T. Zhou *et al.*, "Image restoration using a neural network," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, pp. 940-943, 1988.
- [198] S. C. Newton, B. S. Nutter, and S. Mitra, "Statistical and neural network classifiers in model-based 3-D object recognition," in *Proceedings of the SPIE : Intelligent Robotics and Computer Vision*, (Boston, MA), pp. 209-218, 1990.
- [199] N. M. Nasrabadi and W. Li, "Object recognition by a Hopfield neural network," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 6, 1991.

- [200] T. A. Jamison and R. J. Schalkoff, "Image labeling : a neural network approach," *Image and Vision Computing*, vol. 6, no. 4, pp. 203–213, 1988.
- [201] R. H. Silverman and A. S. Noetzel, "Image processing and pattern recognition in ultrasonograms by backpropagation," *Neural Networks*, vol. 3, pp. 593–603, 1990.
- [202] R. H. Silverman, "Segmentation of ultrasonic images with neural networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 6, pp. 619–628, 1991.
- [203] B. S. Manjunath, T. Simchony, and R. Chellappa, "Stochastic and deterministic networks for texture segmentation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 6, pp. 1039–1049, 1990.
- [204] J. Shah, "Parameter estimation, multiscale representation and algorithms for energy-minimizing segmentation," in *Proceedings International Conference on Pattern Recognition*, pp. 815–819, 1990.
- [205] C. Cortes and J. A. Hertz, "A network system for image segmentation," in *Proceedings International Conference on Neural Networks*, pp. 121–125, 1989.
- [206] W. J. Ho and C. F. Osborne, "Texture segmentation using multilayer backpropagation," in *Proceedings International Joint Conference on Neural Networks*, pp. 981–986, 1991.
- [207] S. S. Yu and W. H. Tsai, "Relaxation by Hopfield neural network," *Pattern Recognition*, vol. 25, pp. 197–209, 1992.
- [208] R. D. Overheim and D. L. Wagner, *Light and Color*. New York: Wiley, 1982.
- [209] Y. Ohta, T. Kanade, and T. Sakai, "Color information for region segmentation," *Computer Vision, Graphics and Image Processing*, vol. 13, pp. 224–241, 1980.

- [210] R. Ohalander, K. Price, and D. R. Reddy, "Picture segmentation using a recursive splitting method," *Computer Vision, Graphics and Image Processing*, vol. 8, pp. 313-333, 1978.
- [211] W. X. Xie, "An information measure for a color space," *Fuzzy Sets and Systems*, vol. 36, pp. 157-165, 1990.
- [212] Y. W. Lim and S. U. Lee, "On the color image segmentation algorithm based on the thresholding and fuzzy c-means techniques," *Pattern Recognition*, vol. 23, no. 9, pp. 935-952, 1990.
- [213] M. D. Levine and A. M. Nazif, "An experimental rule based system for testing low level segmentation strategy," in *Multicomputers and Image Processing : Algorithms and Programs*, (K. Preston and L. Uhr, eds.), New York: Academic Press, 1982.
- [214] N. R. Pal and D. Bhandari, "Object background classification : some new techniques," *Signal Processing*, 1992 (revised).
- [215] A. D. Brink, "Gray level thresholding of images using a correlation criterion," *Pattern Recognition Letters*, vol. 9, pp. 335-341, 1989.
- [216] M. Vidyasagar, "An analysis of the equilibrium of neural networks with linear interconnections," 1992 (submitted for publication).
- [217] "Special issue on neural networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 6, 1991.
- [218] "Special issue on neural networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 6, no. 1, 1992.
- [219] "Special issue on fuzzy logic and neural networks," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, 1992.
- [220] "Special issue on fuzzy logic and neural networks," *International Journal of Approximate Reasoning*, vol. 6, no. 2, 1992.
- [221] *Proceedings First International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), 1990.

- [222] *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), 1992.
- [223] *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), 1992.
- [224] M. Bertero, T. Poggio, and V. Torre, "Ill-posed problems in early vision," *Proceedings of the IEEE*, vol. 76, no. 8, pp. 869–889, 1988.
- [225] C. Koch, J. Morrochin, and A. Yuille, "Analog neural networks in early vision," in *Proceedings National Academy of Science*, (USA), pp. 4263–4267, 1986.
- [226] T. Poggio, V. Torre, and C. Koch, "Computational vision and regularization theory," *Nature*, vol. 317, pp. 314–319, 1985.
- [227] Y. T. Zhou and R. Chellappa, "Stereo matching using neural networks," in *Proceedings IEEE International Conference on Acoustic, Speech and Signal Processing*, New York, pp. 940–943, 1988.
- [228] C. Koch, J. Luo, C. Mead, and J. Hutchinson, "Computation of motion using resistive networks," in *Proceedings Neural Information Processing Systems*, (Denver), 1987.
- [229] H. Bulthoff, J. Little, and T. Poggio, "A parallel algorithm for real time computation of optical flow," *Nature*, vol. 337, pp. 549–553, 1989.
- [230] J. C. Bezdek, "On the relationship between neural networks, pattern recognition, and intelligence," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 85–107, 1992.
- [231] J. M. Keller and D. J. Hunt, "Incorporating fuzzy membership functions into the perceptron algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 7, no. 6, pp. 693–699, 1985.
- [232] R. Krishnapuram and J. Lee, "Fuzzy connective based hierarchical aggregation networks for decision making," *Fuzzy Sets and Systems*, vol. 46, pp. 11–27, 1992.

- [233] S. C. Lee and E. T. Lee, "Fuzzy neural networks," *Mathematical Biosciences*, vol. 23, pp. 151-177, 1975.
- [234] T. Yamakawa and S. Tomada, "A fuzzy neuron and its applications to pattern recognition," in *Proceedings Third IFSA Congress*, (Seattle), pp. 30-38, 1989.
- [235] H. Takagi and I. Hayashi, "Artificial neural network driven fuzzy reasoning," *International Journal of Approximate Reasoning*, vol. 5, pp. 191-212, 1991.
- [236] J. M. Keller and H. Tahani, "Implementation of conjunctive and disjunctive fuzzy logic rules with neural networks," *International Journal of Approximate Reasoning*, vol. 6, pp. 221-240, 1991.
- [237] J. M. Keller, R. R. Yager, and H. Tahani, "Neural network implementation of fuzzy logic," *Fuzzy Sets and Systems*, vol. 45, no. 1, pp. 1-12, 1992.
- [238] J. M. Keller and H. Tahani, "Backpropagation neural networks for fuzzy logic," *Information Sciences*, vol. 62, pp. 205-221, 1992.
- [239] M. M. Gupta, W. Pedrycz, and J. Kiszka, "Fuzzy control : from fuzzy controller to cognitive controller," in *Proceedings Third IFSA Congress*, (Seattle), pp. 258-261, 1989.
- [240] N. P. Archer and S. Wang, "Fuzzy set representation of neural network classification boundaries," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, pp. 735-742, 1991.
- [241] T. L. Huntsberger and P. Ajijmerangsee, "Parallel self-organizing feature maps for unsupervised pattern recognition," *International Journal of General Systems*, vol. 16, pp. 357-372, 1990.
- [242] H. Ishibuchi and H. Tanaka, "Identification of real-valued membership functions," in *Proceedings First International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 179-182, 1990.

- [243] L. O. Hall, "Learning on fuzzy data with a backpropagation scheme," in *Proceedings NAFIPS*, (Missouri-Columbia), pp. 329–332, 1991.
- [244] R. R. Yager, "Fuzzy logic controllers using neural network," *Fuzzy Sets and Systems*, vol. 48, pp. 53–64, 1992.
- [245] K. J. Kios and N. Liu, "A machine learning method for generation of a neural network architecture : a continuous ID3 algorithm," *IEEE Transactions on Neural Networks*, 1992 (to appear).
- [246] J. C. Bezdek, E. C. K. Tsao, and N. R. Pal, "Fuzzy Kohonen clustering networks," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 1035–1046, 1992.
- [247] M. E. Cohen and D. L. Hudson, "Approaches to the handling of fuzzy input data in neural networks," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 93–100, 1992.
- [248] W. Pedrycz, "Neurocomputations in relational systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 289–297, 1991.
- [249] W. Pedrycz and H. C. Card, "Linguistic interpretation of self-organizing maps," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 371–378, 1992.
- [250] C. T. Lin and C. S. G. Lee, "Real-time supervised structure/parameter learning for fuzzy neural networks," in *Proceedings First IEEE International Conference on Fuzzy Systems*, (San Diego, USA), pp. 1283–1291, 1992.
- [251] E. Sanchez, "Genetic algorithms, neural networks and fuzzy logic systems," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 17–19, 1992.
- [252] Y. Hayashi, J. J. Buckley, and E. Czogala, "Direct fuzzification of neural network and fuzzified delta rule," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 73–76, 1992.

- [253] J. M. Keller and Z. Chen, "Learning in neural networks utilizing adaptive hybrid operators," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 85–87, 1992.
- [254] H. Ichibasi, "Hierarchical and recurrent networks of fuzzy models," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 141–144, 1992.
- [255] M. M. Gupta, "Fuzzy logic and neural networks," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 157–160, 1992.
- [256] F. Gomide and A. F. Rocha, "Neuro-fuzzy controller," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 213–216, 1992.
- [257] R. Krishnapuram and J. Lee, "Fuzzy set based hierarchical networks for information fusion in computer vision," *Neural Networks*, vol. 5, pp. 335–350, 1992.
- [258] P. J. Werbos, "Neurocontrol and fuzzy logic : connections and designs," *International Journal of Approximate Reasoning*, vol. 6, pp. 185–219, 1992.
- [259] P. J. Werbos, "Elastic fuzzy logic : a better fit with neurocontrol," in *Proceedings Second International Conference on Fuzzy Logic and Neural Networks*, (Iizuka, Japan), pp. 823–826, 1992.
- [260] C. A. Murthy and S. K. Pal, "Histogram thresholding by minimizing gray level fuzziness," *Information Sciences*, vol. 60, no. 1/2, pp. 107–135, 1992.
- [261] A. Rosenfeld, "Fuzzy digital topology," *Information and Control*, vol. 40, pp. 76–87, 1979.
- [262] A. Rosenfeld, "On connectivity properties of grayscale pictures," *Pattern Recognition Letters*, vol. 16, pp. 47–50, 1983.
- [263] A. Rosenfeld and S. Haber, "The perimeter of a fuzzy set," *Pattern Recognition*, vol. 18, no. 2, pp. 125–130, 1985.

- [264] A. Rosenfeld and R. Klette, "Degree of adjacency or surroundedness," *Pattern Recognition*, vol. 18, no. 2, pp. 169-177, 1985.
- [265] S. K. Pal and A. Rosenfeld, "A fuzzy medial axis transformation based on fuzzy disk," *Pattern Recognition Letters*, vol. 12, no. 10, pp. 585-590, 1991.
- [266] S. K. Pal and L. Wang, "Fuzzy medial axis transformation (FMAT): practical feasibility," *Fuzzy Sets and Systems*, vol. 50, pp. 15-34, 1992.
- [267] L. Kockty, "On the description of relative position of fuzzy sets," *Pattern Recognition Letters*, vol. 8, pp. 21-28, 1988.
- [268] J. M. Keller and L. Sztandera, "Spatial relation among fuzzy subsets of an image," in *Proceedings ISUMA*, (University of Maryland, USA), pp. 207-211, 1990.
- [269] R. A. Melter, "The unfamiliar world of gray geometry," in *Proceedings International Conference on Pattern Recognition*, pp. 951-953, 1984.
- [270] D. Dubois and M. C. Jaulent, "A generalized approach to parameter evaluation in fuzzy digital pictures," *Pattern Recognition Letters*, vol. 6, pp. 251-259, 1987.
- [271] A. Bogomolny, "On the perimeter and area of fuzzy sets," *Fuzzy Sets and Systems*, vol. 23, pp. 257-269, 1987.
- [272] A. Rosenfeld, "Fuzzy rectangles," *Pattern Recognition Letters*, vol. 11, pp. 677-679, 1990.
- [273] S. K. Pal and A. Ghosh, "Fuzzy geometry and image processing," in *Proceedings National Conference on Electronic Circuits and Systems (NA-CONECS)*, (Roorkee, India), pp. 97-99, 1989.
- [274] A. Ghosh, *Application of fuzzy geometry in image analysis*. Master's thesis, Indian Statistical Institute, Calcutta, India, 1989.
- [275] S. K. Pal and A. Ghosh, "Index of area coverage of fuzzy image subsets and object extraction," *Pattern Recognition Letters*, vol. 11, no. 12, pp. 831-841, 1990.

- [276] S. K. Pal and A. Ghosh, "Fuzzy geometry in image analysis," *Fuzzy Sets and Systems*, vol. 48, no. 2, pp. 23-40, 1992.
- [277] C. A. Murthy and S. K. Pal, "Bounds for membership function : correlation based approach," *Information Sciences*, 1992 (in press).
- [278] A. B. Leigh, *A fuzzy measure approach to motion frame analysis for video data abstraction*. Master's thesis, University of Houston Clear-Lake, Houston, Texas, USA, 1992.
- [279] L. Davis, *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold, 1991.
- [280] D. E. Goldberg, *Genetic Algorithms : Search, Optimization and Machine Learning*. New York: Addison-Wesley, 1989.
- [281] *Proceedings of the Fourth International Conference on Genetic Algorithms*, (University of California, San Diego), 1991.
- [282] T. Kohonen, "Self organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, pp. 59-69, 1982.
- [283] A. C. Cohen and D. B. Cooper, "Simple parallel hierarchical and relaxation algorithms for segmenting noncausal Markovian field," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 195-219, 1987.
- [284] J. Besag, "Spatial interaction and statistical analysis of lattice systems," *Journal of Royal Statistical Society*, vol. B-36, pp. 192-236, 1974.
- [285] J. Besag, "On the statistical analysis of dirty pictures," *Journal of Royal Statistical Society*, vol. B-48, pp. 259-302, 1986.
- [286] R. Kinderman and J. L. Snell, *Markov Random Fields and Their Applications*. American Mathematical Society: Providence, 1980.
- [287] F. Splitzer, "Markov random fields and Gibbs ensembles," *American Mathematical Monthly*, vol. 78, pp. 142-154, 1971.

- [288] R. B. Ash, *Real Analysis and Probability*. New York: Springer-Verlag, 1975.
- [289] H. Bourland and C. J. Wellekens, "Links between Markov models and multilayer perceptrons," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 1167-1178, 1990.
- [290] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1973.
- [291] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671-680, 1983.
- [292] H. A. Taha, *Operations Research : An Introduction*. New York: Macmilan Publishing Co., 1982.
- [293] J. J. Hopfield, "Learning algorithms and probabilistic distributions in feedforward and feedback networks," in *Proceedings National Academy of Science, (USA)*, pp. 8429-8433, 1987.
- [294] R. J. McEliece, E. X. Posner, E. R. Rodenmich, and S. S. Venkatesh, "The capacity of the Hopfield associative memory," *IEEE Transactions on Information Theory*, vol. 33, pp. 461-482, 1987.
- [295] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 76-86, 1992.
- [296] R. S. Scaler and N. Tepedelenlioglu, "A fast new algorithm for training feedforward neural networks," *IEEE Transactions on Signal Processing*, vol. 40, pp. 202-210, 1992.

List of papers by the author

1. 'Fuzzy geometry and image processing', *Proc. of the National Conference on Electronic Circuits and Systems (NACONECS)*, pp 97-99, November, 1989, Roorkee, India. (Co-author : S. K. Pal)
2. 'Feature ranking using fuzzy correlation', *Proc. of the International Conference on Fuzzy Logic and Neural Networks*, pp 561-564, July, 1990, Iizuka, Japan. (Co-author : S. K. Pal)
3. 'Self-organisation of neural network and object extraction', *Proc. of the Workshop on Signal Processing, Communications and Networking*, pp 241-246, July, 1990, Bangalore, India. (Co-author : S. K. Pal)
4. 'Index of area coverage of fuzzy image subsets and object extraction', *Pattern Recognition Letters*, vol-11, no-12, pp 831-841, 1990. (Co-author : S. K. Pal)
5. 'Neural network, Gibbs distribution and object extraction', *Proc. of the International Symposium on Intelligent Robotics (ISIR)*, pp 95-106, January, 1991, Bangalore, India. (Co-authors : N. R. Pal and S. K. Pal)
6. 'Object extraction using a self-organizing neural network', *Proc. of the International Symposium on Intelligent Robotics (ISIR)*, pp 686-697, January, 1991, Bangalore, India. (Co-authors : N. R. Pal and S. K. Pal)
7. 'Image segmentation using a neural network', *Biological Cybernetics*, vol-66, no-2, pp 151-158, 1991. (Co-authors : N. R. Pal and S. K. Pal)
8. 'Neural network, self-organization and object extraction', *Pattern Recognition Letters*, vol-13, no-5, pp. 387-397, 1992. (Co-author : S. K. Pal)
9. 'Image segmentation using fuzzy correlation', *Information Sciences*, vol-62, no-3, pp. 223-250, 1992. (Co-author : S. K. Pal)
10. 'Fuzzy geometry in image analysis', *Fuzzy Sets and Systems*, vol-48, no-2, pp. 23-40, 1992. (Co-author : S. K. Pal)

11. 'Neural computing : an introduction and some applications', *Students' Journal of the Institute of Electronics and Telecommunication Engineers* (accepted). (Co-authors : N. R. Pal and S. K. Pal)
12. 'Object background classification using Hopfield type neural network', *International Journal of Pattern Recognition and Artificial Intelligence*, vol-6, no-5, 1992. (Co-authors : N. R. Pal and S. K. Pal)
13. 'Self-organization for object extraction using multilayer neural network and fuzziness measures', *IEEE Transactions on Fuzzy Systems*, vol-1, no-1, 1993. (Co-authors : N. R. Pal and S. K. Pal)
14. 'On robustness of neural network based systems under component failure', *IEEE Transactions on Neural Networks* (revised). (Co-authors : N. R. Pal and S. K. Pal)