Natural Redundancy ~~and Coding Redundancy~~ in Information Signals

by

D. Kabilafkas, Dip. E.E, M.Sc.

Thesis submitted for the Degree of Doctor of Philosophy of the University of London

and for the Diploma of Membership of the Imperial College

LONDON , NOVEMBER 1988

IMPERIAL COLLEGE

Department of Electrical Engineering

Digital Communication Section

to the memory of my late father Σταθη.

ABSTRACT

Certain aspects of redundancy in communication which relate to the central issues handled by Shannon's Information Theory are dealt with in this thesis.

At first, aspects relating to channel coding are dealt with. A new class of codes, referred to as 1.5 Reed-Muller codes (1.5 RM codes), is introduced and described. The codes are such that they add to the range of available lengths and distances as compared with those that are currently available with the existing class of Reed-Muller codes. Consideration is also given to the question of the decoding of the new codes and it is shown that, as formulated and synthesized, the codes cannot be decomposed in such a way as to render them majority logic decodable.

The redundant nature of written (or printed) communication is then discussed and the problems of pattern recognition are outlined. The recognition is considered as a process of redundancy removal and hence, as a process that can be regarded as the inverse of encoding. The difference between the recognition of hand-printed and machine-printed characters is considered.

The problem of recognizing hand-printed characters is analysed and the process of recognition is viewed as taking place in two interrelated stages. The first stage deals with individual characters and the removal of redundancy so as to obtain only essential information which distinguishes between characters. The question of thresholding is considered and a purely automatic technique for the location of threshold positions is put forward. Following this, the question of coded representation of thin line images is examined and a new approach involving Soft Skeletonization and associated Stride Encoding is put forward and this too is examined.

In the second stage of the recognition process, characters are considered to be identified as a structured set of locally identifiable elements and it is argued that the representation of the set as a complicated multi-dimensional

probabilistic model is required for the representation of the set. The concept of modelling by pattern synthesis is presented.

An alternative approach to modelling involving the use of Markov processes is discussed and the question of how a model can be inferred from sample data is considered. Criteria for the inference are discussed and the Minimum Description Length criterion is adopted. A Multidimensional Dynamic Programming approach is put forward as a means of finding the minimum description length. A new theorem is stated and proved by which it is shown that it is possible to combine identical strings without incurring any loss in the optimality in respect to the MDL criterion. The theorem allows the computational burden involved in determining the minimum description length and the associated Markov model to be significantly reduced.

Acknowledgement

I would like to express my graditude to Professor L.F. Turner for his guidance and for giving me the opportunity to have a taste of the search for a better understanding of Information, in the way opened up by C.E.Shannon, both in Nature and espesially within ourselves, the Human Beings. I am especially indebted to him for his active and patient help, in excess of his duties as a supervisor, during the hard "writing-up", that made the presentation of this thesis possible.

For all my education since childhood, I thank my parents and all my family for their love and support. I also express my love and thanks to Efi for being by my side, albeit from afar during the very first years of our common life.

I would also like to mention the State Scholarship Foundation for the honour of granting to me a scholarship, which constituted the basic moral encouragement and financial support for the posgraduate fulfilment of my studies.

I would also like to thank all the academic staff, as well as the secretarial, computer and other staff who contributed to the operation of the College.

I also thank all friends and colleagues who had a chat with me during these years.

INDEX

PAGE

ABSTRACT 3

AKNOWLEDGEMENTS 5

INDEX 6

CHAPTER 1
INTRODUCTION 9

1.1 ASPECTS RELATING TO REDUNDANCY: NATURAL REDUNDANCY AND CODING (DESIGNED) REDUNDANCY 10

1.1.1 Introduction 10

1.1.2 EXAMPLE: The Problem of OCR: From KBits to 1 byte (or less) 12

1.1.3 Removal of Redundancy 13

5.2. REVIEW OF THE CONTENTS OF THE THESIS 15

REFERENCES 19

CHAPTER 2
A NEW CLASS OF DOUBLE INDUCTIVE BINARY ERROR CORRECTING CODES 20

2.1 INTRODUCTION 21

2.2 CONSTRUCTION OF THE NEW CODES 23

2.3 PROOF THAT THE MATRIX (2.2) GENERATES THE NEW CLASS OF CODES 27

2.4 DIMENSIONALITY CONSIDERATIONS 33

2.5 DECODING BY MAJORITY LOGIC 35

2.6 CONCLUSIONS 43

REFERENCES 46

FIGURES 47

PAGE

CHAPTER 3
THE CODING OF HANDPRINTED CHARACTERS 50

3.1 INTRODUCTION 51

3.2 REDUNDANCY CONTAINED IN CHARACTER IMAGES 52

3.3 THE CODING OF LINE IMAGES 55

3.3.1 Thresholding 55

3.3.1.1 An Annealing Technique for the Automatic location of thresholds 58

3.3.2 Hard Skeletonization 61

3.3.3 The Further Encoding of a Hard Skeleton 63

3.4 A NEW LOSSY APPROACH TO THE ENCODING OF LINE IMAGES 65

3.4.1 Introduction 65

3.4.2 Soft skeletonization using a Center of Gravity criterion 67

3.4.2.1 Examples of Soft Skeletons produced from two tone images 73

3.4.3 A Stride Encoding technique 74

3.4.3.1 The Stride encoding of the soft skeletons considered earlier 78

3.5 SOME CONCLUSIONS 80

REFERENCES 81

FIGURES 82

APPENDIX A

APPENDIX B

CHAPTER 4
ASPECTS RELATED TO THE MODELLING OF HANDPRINTED CHARACTERS 100

4.1 INTRODUCTION 101

4.1.1 The Concept of a Handprinted Character 101

4.1.2 Use of Character Synthesis in Model Development 106

4.2 FINITE STATE MACHINE MODELLING OF PATTERNS 107

4.2.1. Intoduction 107

4.2.2. Modelling 110

4.2.3 Dnamic Programming Inference of Markov Networks 113

4.3 CONCLUSIONS 127

REFERENCES 129

FIGURES 130

CHAPTER 5
CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK 139

5.1 CONCLUSIONS 140

5.2 SUGGESTIONS FOR FURTHER WORK 144

REFERENCES 146

PUBLICATIONS BASED ON MATERIAL CONTAINED IN THIS THESIS 147

Chapter 1
Introduction

INTRODUCTION

1.1 ASPECTS RELATING TO REDUNDANCY: NATURAL REDUNDANCY AND CODING (DESIGNED) REDUNDANCY.

1.1.1 Introduction

The entities of Information and Redundancy are not new. They have formed one of the basic aspects relating to human communication since earliest times. It was only in recent years, however, that these concepts have been clearly defined and given precise scientific meaning. Interestingly enough, although the ideas of Information and Redundancy were formulated as the result of studies in the field of telecommunications, these ideas now find wide application in fields of studies such as Linguistics, Genetics and Anthropology, and with the advent of Information Technology, the concepts of Information and Redundancy are common in everyday life.

Concern about aspects of the transmission of information such as data compression (saving in time or spectrum), and reliability (error correction or error detection), that are central to Information Theory, and consideration of these aspects has developed in line with the growth of modern electronic communication systems and the increase in the volume of transmitted signals. Although ad hoc solutions to related problems, such as the invention of Morse code and the use of a parity bit, for instance, existed prior to 1948, Information Theory was given exact expression and mathematical formulation by C.E.Shannon in his classic paper [Ref 1.1] of 1948. In this paper, Shannon dealt with two different, but interrelated, aspects of the subject of information. The first aspect dealt with by Shannon was concerned with defining in precise terms what Information is and how it can be measured. This part of his work deals with what is now known as "entropy measures" and "redundancy" and the application of ideas based on Shannon's original paper is now known as source coding. In the second part of his work, Shannon dealt with the question of communicating signals reliably over a noisy communication channel and he

was able to prove that a noisy communication link has a finite capacity and, with a rate not exceeding the capacity (upper limit), information can be transmitted over the channel with an arbitrarily small probability of error. Shannon, thus, showed that an essential feature of reliable communication is the introduction of redundancy into the signal to be transmitted. This process has come to be known as Channel Coding or Error Correction Coding.

Information Theory deals with stochastic aspects of the transmission of information and, since probability theory is the best framework for the formulation of stochastic phenomena, Information Theory is naturally based on it.

A central issue in applying Information Theoretic concepts is the question of how probabilistic considerations should be formulated. Looking at the communication of a signal as a probabilistic experiment (stochastic phenomenon) one may not be interested in obtaining as output an "exact replica" of the input. In the case of image transmission, for instance, it may be that only the general shape, or the position, of the object may be of significance. A different example concerns the decoding of linear codes. In this case it is the column of the standard array in which the codeword belongs that is looked for, since all entries in the column are decoded as the same code word and hence can be regarded as equivalent in this sense.

Shannon's theory provides measures of the information carrying capacity of a communication link and of the information content of a source. With regard to information measures, however, Shannon's theory is a strictly probabilistic theory and it takes no account of what is regarded as significant and what is irrelevant, from user's point of view. The decision of what is significant and what is irrelevant depends on the user and the specific application. For example, when considering handprinted text, one is usually interested in the actual letter of the alphabet (the pattern class) represented by the handprinted image. At another level, however, the same handprinted text carries information of a different kind. It can, for example, be used in a forensic application to determine the identity of the individual responsible for producing

the image (images).

Before proceeding to a discussion of different cases of redundancy removal, an example will now be given to illustrate the concepts of capacity and source entropy, in a manner that is relevant to the content of this thesis.
The example is concerned with optical character recognition (OCR).

1.1.2 EXAMPLE: The Problem of OCR: From KBits to 1 byte (or less).

Handwritten script (including handprinting) and machine printed script transmits information through images and an image, as a Chinese proverb says "is worth a thousand words".

Based on Shannon's Information Theory, it is sensible to attempt to quantify the information contained within a character image of the type referred to above. For example:

(i) How much information can the image convey?

(ii) Of the information carried by the character image, how much of it is actually considered to be information by the individual for whom it is intended?

Consider the image to be digitized so as to contain a given finite number of pixels (pels). Then if each pixel is quantized and the levels into which it is quantized are equally likely and there is no correlation between pixels, it follows from Information Theory that the information content of the image is maximum and, depending on the size of the image and the fineness of the pixel resolution, it may contain tens of KBits. A observer, or receiver, can interpret the image as a single ASCII character which usually involves a representation of 8 bits including the parity check bit. If linguistic constraints are involved then the number of digits needed to represent a character is significantly reduced from 8. . The characteristics that are responsible for the processing leading to the gradual reduction of the information content of the character image are:

(i) the human being interprets and writes characters as thin line images;

(ii) apparently different line images are considered by the human being to be

members of a particular class which, hopefully, can be represented by using representative pattern images or by a very limited number of such images.

The first factor is dealt with in Chapter 3 of this thesis and the second factor in Chapter 4 where it is explained that certain variations in the shape and/or size of the elements constituting the image are irrelevant.

Another aspect of redundancy in characters (as well as in language in general) is associated with contextual constraint. In the communication of information by text, it is the message that must be passed rather than the character(s) themselves. The character alone means even less in modern alphabets, originating from the Phoenicians, where it represents a mere phoneme (part of the syllable), instead of a word or a phrase, as in other (ancient or modern) alphabets. The contextual structure, in the level of the syllable, the word or the sentence (or even higher) has an effect in reducing even further the information context of the language and, therefore, the average information content of the characters. It is known, for example, that when linguistic constraints at the word level are taken into account, the entropy of most natural languages is approximately 2.5 bits per letter [Ref 1.2]. Although the use of the context in automatic recognition of text is likely to lead to particular effective and reliable redundancy reduction, it will not be considered in this thesis, although it constitutes an area worthy of detailed investigation.

1.1.3 Removal of Redundancy

In order to see more clearly the trade off between information and redundancy and to address certain specific problems that can be considered within the framework of Information Theory, let us begin by considering the situation in which the source of the messages, the set of signals representing them and the behaviour of the communication channel are completely and sufficiently specified and modelled. This is the situation where channel coding can be applied in order to achieve a certain level of performance when

communicating over a noisy channel.

The assumption is often made that we have a signal which is fully information carrying (it contains the message) and a decision is taken to introduce redundancy into it in order to achieve reliable communication in the presence of random noise. It is clear that a number of bits transfer information and the rest are redundant (or check bits). In this situation the exchange of information rate for gain in reliability is clear.

In the case of channel coding an efficient decoder may not exist, but the objective to be achieved by the decoder can be defined in clear terms. It has to correct errors, remove the artificially introduced redundancy and in so doing arrive at a reliable estimate of the information digits contained within the coded message.

In the case of the removal of natural redundancy, on the other hand, the problem is not nearly so well defined, nor is it amenable to mathematical formulation and analysis. The following four cases relate to natural redundancy removal and touch on problems referred to above.

The simplest case of the removal of natural redundancy results in taking advantage of the uneven statistical nature of the source and in the encoding of the source without any loss of fidelity. This topic relates directly to information lossless source coding, with an example being the well known Huffman code [Ref 1.3]. Other more complicated cases involve the redundancy contained in signals such as images and speech, and with the encoding of these subject to the introduction of some tolerable distortion. Related to this is the definition with some objective cost function or subjective criteria of fidelity. This has found formulation in the topic of rate-distortion theory [Ref 1.4]. At present, considerable difficulties are experienced in arriving at sensible cost criteria and subjective criteria of fidelity.

Another area of investigation concerned with redundancy removal is that of Pattern Recognition, in which an attempt is made to go from a highly redundant signal to an efficient representation of a pattern or a set of

patterns. This view of pattern recognition was put forward by S.Watanabe in his classic paper entitled "Pattern recognition as a quest for minimum entropy" [Ref 1.5].

One example of this approach to pattern recognition is the so-called decision theoretic approach in which a well-defined pattern is considered as THOUGH it will be observed through a channel of known characteristics.

A different and more general approach to pattern recognition is to assume that no clear definition of what constitutes the pattern exists and the "labels" associated with a particular class are the result of an external classifier, usually the human being. An important example which falls within the latter range of problems is that of the machine recognition of handprinted characters. It is important to note that the human being is the "*ultimate referee*" and "*arbitrator*"' in respect to what constitutes a successful classifier.

1.2. REVIEW OF THE CONTENTS OF THE THESIS.

Since the advent of Shannon's capacity theorem, channel coding theory has developed very rapidly, with the aim being to design codes with desirable characteristics, such as information rate and distance-to-length ratio. Such codes have important practical advantages. Although, up to the present, there is no constructive algorithm for the design of codes which achieve a rate equal to the channel capacity whilst having an arbitrarily small probability of error, many classes of codes have been designed which, nevertheless, are very useful from the application point of view. The design of codes, especially the design of the block codes, is based on algebraic theory. For reasons associated with the algebraic structure of codes, some classes of codes have a length that is a power of 2. A well-known class of codes having block length equal to a power of 2 is the class of Reed-Muller codes.

In Chapter 2, a new class of codes, referred to as 1.5 Reed-Muller codes

(1.5 RM codes), will be introduced. The codes are such that they add to the range of available lengths and distances as compared with those that are currently available with the existing class of Reed-Muller codes. More precisely, the new codes have the same length-to-distance ratio as certain members of the existing class of Reed-Muller codes, and for those cases in which the length-to-distance ratio is the same as for those members of the existing class of RM codes, the new codes are available with lengths and dimensions which lie between those values that are available with the existing class of RM codes.

Consideration will also be given in the chapter to the question of whether the new codes can be majority logic decoded. It will be shown that the codes as formulated and synthesized cannot be decomposed in such a way as to render them majority logic decodable.

It should be appreciated that the task to be performed by a decoder is well-defined and decoding can always be defined in terms of the trivial solution involving a look-up table. The exact formulation of the decoding, which can be considered as a redundancy removal task, is the feature that distinguishes it from the redundancy removal work contained in Chapters 3 and 4.

Chapters 3 and 4 deal with aspects related to the recognition of handprinted characters. There is a massive amount of literature on the topic of character recognition, and the volume of published work is disproportionate to the existing number of actual character recognition systems operating in practice. The reason for this is that the problem of the recognition of handprinted, as opposed to machine printed characters, is very much more difficult than might be imagined. This difficulty is due mainly to the very wide range and variety of characters that a human being classifies as being a particular pattern class.

Chapter 3 deals with the "image processing" stage of the character recognition system, whereas Chapter 4 deals with material related to the actual recognition of character patterns (a 3rd stage could be that dealing with the context). Although these stages are not completely separate during the recognition as performed by the human being, methodology reasons suggest this

division. The efficiency of the first part of the system can be addressed without referring to the second, based on the criteria of subjective invertibility of the process and the parsimony of the output. This matter is taken up in Chapter 3.

The first stage after scanning an image with an appropriate device is the thresholding, that is the production of a two-tone image. Recent work on the topic of thresholding, based on fuzzy set theoretic and information theoretic concepts, is reviewed in Chapter 3, and a new technique is proposed by which the computation involved in arriving at a threshold is reduced. A new pure automatic annealing-based method for the location of the threshold positions is then presented and extended for the case of thresholding multi-level images.

The question of the production of a reduced redundancy coded representation is then considered and conventional methods for skeletonization and chain coding are reviewed. Following this, a new concept which is referred to as "*Soft Skeletonization and Stride Encoding*" is set out. Soft skeletonization is based on a "Center-of-Gravity" criterion and the analytic evaluation of the criterion for constant-width strokes is given. Finally, examples for the operation of the overall scheme are presented.

Chapter 4 relates to the second stage of recognition, namely the identification or classification stage. It is argued that a handprinted character should be considered as "*a global structure of locally identifiable elements*"; and it is further argued that a complicated model, capable of expressing the interrelation between different parts of the character, is needed. The idea of validating a model by pattern synthesis is proposed and, although no work is done in the thesis, it is suggested that it forms an interesting area for further investigation.

Difficulties of analysis of the most general and valid model force one to consider less succinct models that are easier to analyse. Markov Models fall into this category and the second part of Chapter 4 deals with modelling by Markov Networks. The question of inference is then taken up in the chapter and the powerful approach of M.G.Thomason et al [Ref 1.6] is briefly outlined

and discussed. Following this, a new approach, based on the principle of Minimum Description Length is proposed. A Multidimensional Dynamic Programming method is given and the complexity of the method is dealt with. In particular, a new theorem is proved relating to the "merging of similar axis" and it is shown that such merging does not result in any reduction of the optimality of the minimum description length so found. The theorem has important consequences in that it leads to an exponential reduction in the number of computations needed to arrive at the minimum description length.

REFERENCES

[1.1] C. E. SHANNON, "*The Mathematical Theory of Communication*", Bell System Technical Journal, 47, 1948.

[1.2] D.A.BELL, "*Information Theory and its Engineering Applications*", Pitmam & Sons, 1953/56

[1.3] D.A. HUFFMAN, "A Method of Construction of Minimum Redundancy Codes", Proc. IRE, 40(10), 1952

[1.4] T. BERGER, Rate-Distortion Theory", Prentice Hall, (1971)

[1.5] S.WATANABE "Pattern Recognition as a Quest for Minimum Entropy", Pattern Recognition, Vol. 13. No 5 (1981)

[1.6] M.G.THOMASON, E.GRANUM "Dynamic Proggramming Inference of Markov Networks from Finite Sets of Sample Strings", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No 4, 1986

Chapter 2
A New Class Of Double Inductive Binary Error Correction Codes

CHAPTER 2
A NEW CLASS OF DOUBLE INDUCTIVE BINARY ERROR CORRECTING CODES

2.1 INTRODUCTION

In the introductory chapter it was explained that the redundancy associated with signals may be deliberately introduced, as in the case with error detecting and correcting codes, or it may be naturally occurring and may be the result of the process of evolution.

In this present chapter attention is focused on the first type of redundancy, namely that which is introduced deliberately for the purpose of error correction. The redundancy is added in a controlled way in accordance with certain rules, thereby ensuring a required degree of correcting capability and, hopefully, leading to relatively easy decoding.

In this chapter consideration is given to the development of a new class of codes. The codes can be considered as being an extension of the existing well-known class of Reed-Muller codes.

The Reed-Muller codes (Refs [2.1] and [2.2]) cover a wide range of rates and minimum distances. For any m and $0 \leq r \leq m$, there exists a code in the class of Reed-Muller codes for which

$$n = 2^m, \qquad k = \sum_{i=0}^{r} \binom{m}{i} \text{ and } \qquad d = 2^{m-r}$$

Such a code is usually denoted RM(r,m) and the symbols n, k and d indicate respectively the code-word length, the number of information digits (the dimension of the code) and the minimum weight or distance.
The integer r is called the "order" of the code.

The generator matrix of the RM[r,m] code contains the products of rows of an m by 2^m matrix A. The columns of A are the binary m-tuples representing the numbers from 0 to $2^m - 1$. The generator matrix of the RM(r,m) code is formed from the all ones row, the set of individual rows of A, and all products of rows of A up to the product of r rows of A.

In this chapter a new class of codes having length $n = 1.5 \cdot 2^m$; $m \geq 3$ and integer; and minimum distance $d = 1.5 \cdot 2^{m-r}$; $1 \leq m-r \leq m-2$ is introduced. The members of the class will be referred to as 1.5RM codes on account of the facts (i) that they have a double-inductive property that can be associated with RM codes, and (ii) the new codes have the same length-to-distance ratio as certain members of the class of the RM codes. The members of the class of RM codes for which the length-to-distance ratio is the same as the new class of codes are those whose generator matrix forms part of the generator matrix of the new codes.

The double inductive property referred to is the method which is used for the proof of the minimum distance property of the new codes, and which can also be used for the proof of the minimum distance property of the class of RM codes.

The generator matrix of an RM code can be expressed in terms of submatrices which are themselves generator matrices of RM codes of the same, or higher, order and smaller length. When considering the sequence of RM codes of the same order, that is, the same length-to-distance ratio, the RM(r,m) code is based on the previous code RM(r,m−1) of the sequence, that is, the code of the same r, but of half the length (and since the length-to-distance ratio is the same, the minimum distance is also halved) and on the code RM(r−1,m−1) which has half the length and the same minimum distance $d = 2^{m-r}$. This, of course, is provided both of these codes exist.

The generator matrix of the 1.5RM(r,m) code, which is a member of the new class of codes, is constructed in a similar manner from matrices that generate the RM(r,m−1), RM(r−1,m−1) and RM(r−2,m−1) codes. The new codes can be shown to have length and dimension that are 1.5 times those of the RM(r,m) code and, although the rate of the so constructed code is less than that of the RM(r,m) code, it can be shown that for rates less than 1/2 it has a higher rate than the next RM code, that is, the RM(r,m+1) code in the RM set.

2.2 CONSTRUCTION OF THE NEW CODES

As mentioned above, the notation RM(r,m) is used to indicate a Reed-Muller code of length $n = 2^m$, order r and minimum weight $d = 2^{m-r}$.

In the case of the new codes that are to be defined, the notation 1.5 RM(r,m) is used to indicate a code of length $n = 1.5 \cdot 2^m$ and minimum weight $d = 1.5 \cdot 2^{m-r}$.

Let the generator matrices (GM) of the RM(r,m) and 1.5 RM(r,m) codes be denoted by [RM(r,m)] and [1.5 RM(r,m)] respectively.

It has been found by the author that, if the rows of the usual form of the generator matrix of the RM(r,m) code are rearranged, then the generator can be derived by double induction from the four submatrices as follows[1,2]:

$$[RM(r,m)] = \left[\begin{array}{c|c} [RM(r,m-1)] & [RM(r,m-1)] \\ \hline [0] & [RM(r-1,m-1)] \end{array}\right] \qquad (2.1)$$

For example, the usual form of the generator matrix of the RM(2,4) code is

$$\left|\begin{array}{cccc} 1111 & 1111 & 1111 & 1111 \\ 0000 & 0000 & 1111 & 1111 \\ 0000 & 1111 & 0000 & 1111 \\ 0011 & 0011 & 0011 & 0011 \\ 0101 & 0101 & 0101 & 0101 \\ 0000 & 0000 & 0000 & 1111 \\ 0000 & 0000 & 0011 & 0011 \\ 0000 & 0000 & 0101 & 0101 \\ 0000 & 0011 & 0000 & 0011 \\ 0000 & 0101 & 0000 & 0101 \\ 0001 & 0001 & 0001 & 0001 \end{array}\right|$$

[1] It has subsequently been discovered that the independent finding of the above is contained in Theorem 2, Chapter 13 of Ref [2.1]

[2] see page 45

If the rows of this matrix are rearranged as follows

1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1	0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1	0 1 0 1 0 1 0 1
0 0 0 0 0 0 1 1	0 0 0 0 0 0 1 1
0 0 0 0 0 1 0 1	0 0 0 0 0 1 0 1
0 0 0 1 0 0 0 1	0 0 0 1 0 0 0 1
0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0	0 0 0 0 1 1 1 1
0 0 0 0 0 0 0 0	0 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0	0 1 0 1 0 1 0 1

then this matrix can be expressed in terms of the four submatrices referred to above. The appropriate partitioning of the generator matrix is indicated in the matrix above by the continuous lines.

The new 1.5 RM(r,m) codes are structured such that for each member of the class the generator matrix consists of nine submatrices, as follows:

[1.5 RM (r,m)] =

[RM(r,m−1)]	[RM(r,m−1)]	[RM(r,m−1)]
[0]	[RM(r−1,m−1)]	[RM(r−1,m−1)]*
[0]	[0]	[RM(r−2,m−1)]

(2.2)

The matrix [RM(r−1,m−1)]* in expression (2.2) generates the same code as the matrix [RM(r−1,m−1)], but differs from it in a manner to be

described in the next section. All other matrices in expression (2.2) are of the conventional form associated with the corresponding RM code.

The notation GM_{ij}; $1 \leq i \leq 3$; $1 \leq j \leq 3$, will be used in a conventional way to denote the vertical and horizontal position of the submatrix within the generator matrix (2.2); with for ease of reference, the position of a submatrix being defined in terms of its row and column position. This must not be confused with the actual rows and columns of [1.5 RM(r,m)] however. From the description given in the introductory section relating to the generator matrix of the RM(r,m) code it will be clear that the generator matrix of the RM[r,m−1] code contains the products of rows of an (m−1) by (2^{m-1}) matrix A. The columns of A are the binary n-tuples representing the numbers from 0 to $2^{m-1}-1$. The generator matrix of the RM(r,m−1) code is formed from the all ones row, the set of individual rows of A, and all products of rows of A up to the product of r rows of A. The generator matrix $[RM(r,m-1)] = GM_{11} = GM_{12} = GM_{13}$ of the RM(r,m−1) code can be divided into three parts with the parts being denoted by GM_{ijp}; i = 1, p=1,2,3, respectively. The first part consists of the all ones row, the set of individual rows of A and all products up to r−2 rows of A. The second part consists of the set of products of r−1 rows of A, and the third part consists of the products of r rows of A. In general, the notation $GM_{ijp}(t)$ will be used to indicate the t th row in the matrix GM_{ijp}.

The matrix $GM_{22} = [RM(r-1,m-1)]$ in (2.2) is defined to consist of two parts; the first part being identical to GM_{1j1} of the matrix [RM(r,m−1)] and the second part identical to GM_{1j2} of [RM(r,m−1)], that is;

$$GM_{221} = GM_{1j1}$$
$$\text{and} \quad GM_{222} = GM_{1j2} \; ; j = 1,2,3$$

The matrix GM_{33} in (2.2) is defined to consist of only the first part, GM_{1j1}, of the matrix [RM(r−1,m−1)]. For consistency of notation this matrix GM_{33} will be denoted by GM_{331} and hence $GM_{33} \equiv GM_{331} = GM_{1j1}$, j=1,2,3.

Consider now the question of the matrix $GM_{23} = [RM(r-1,m-1)]^*$ in (2.2).

It will be shown in the following section that if GM_{231} is identical to GM_{221} then a construction[2] (which is not in fact unique) which results in the new class of codes is that in which GM_{232} is obtained from GM_{222} according to the following rules:

(i) row t in GM_{232} is the same as row $t+1$ in GM_{222} for $t = 1,\ldots,l-1$; where l is the dimension of GM_{222} (and of course GM_{232})

(ii) row l in GM_{232} is given by a linear combination
$\alpha_0 \cdot GM_{222}(1) + \alpha_1 \cdot GM_{222}(2) + \ldots + \alpha_{l-2} \cdot GM_{222}(l-1) + \alpha_{l-1} \cdot GM_{222}(l)$
of the rows of GM_{222}; with $\alpha_0, \alpha_1, \ldots, \alpha_{l-1}, 1$; being the coefficients of any irreducible polynomial of degree l in GF(2).

Before proving that the matrix in (2.2) generates the new class of codes, an illustrative example is given in which the generator matrix of the 1.5 RM(2,4) code is constructed.

EXAMPLE:

Beginning with the generator matrix

1 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1
0 0 0 0 0 0 1 1
0 0 0 0 0 1 0 1
0 0 0 1 0 0 0 1

which generates the RM(2,3) code, then the matrix that generates the RM(1,3) code is seen to be:

---

[2]Although it is not shown in this thesis, it can be shown that the number, Fs(l), (the number Fs(l) is a function of the dimension, l, of the matrix GM_{222}) of different constructions that satisfy the Lemma given in Section 2.3 and which, therefore, can be used to generate the new codes is :

$$Fs(l) = I(l) \cdot Fsd(l) \text{ where } Fsd(l) = Fsd(l-1) + (-1)^l \cdot \frac{2^{\binom{l}{2}}}{I(l)}$$

with $Fsd(-1)=0$ and $I(l) = \prod_{i=0}^{l-1} \left(2^l - 2^i\right)$

1 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1

Now, in accordance with the description given above it follows that GM_{221} is:

1 1 1 1 1 1 1 1

and GM_{222} is:

0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1

Also, GM_{231} is:

1 1 1 1 1 1 1 1

If the irreducible[3] polynomial $\alpha^3 + \alpha + 1$ is used and the rules for combination set out above are applied, then GM_{232} is seen to be:

0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1
0 0 1 1 1 1 0 0

The final resulting matrix for the 1.5 RM(2,4) code of length 24, dimension 12 and minimum distance 6 is shown in Fig 2.1.

2.3 PROOF THAT THE MATRIX (2.2) GENERATES THE NEW CLASS OF CODES.

In this section it will be proved that the matrix given in (2.2) generates the new class of codes.

It is clear from the generator matrix (2.2) that the length of the code

[3]It should be noted that irreducibility is a sufficient but not a necessary condition. All that is necessary for the particular type of construction involved with all but the last row moving upwards by 1, is that the polynomial shall not have X+1 as a factor, that is, it must be of odd weight.

generated is $3 \cdot 2^{m-1}$, that is, the length is 1.5 times that of the RM(r,m) code.

It is now to be proved that the minimum weight, and hence the minimum distance, of the new class of codes, is also 1.5 times the minimum weight of the RM(r,m) code.

Since any codeword of the code is made up of linear combinations of rows of the generator matrix, it follows that a codeword, C, can be considered to be a juxtaposition

$$C = (C_{11}:C_{12}:C_{13}) \oplus (C_{21}:C_{22}:C_{23}) \oplus (C_{31}:C_{32}:C_{33}) = C_1 \oplus C_2 \oplus C_3$$

where C_{ij} is a linear combination of rows of the submatrix GM_{ij} and C_i, $i=1,2,3$, is defined as the juxtaposition $C_{i1}:C_{i2}:C_{i3}$. The codeword can be expressed alternatively as

$$C = (C_{11} \oplus C_{21} \oplus C_{31}):(C_{12} \oplus C_{22} \oplus C_{32}):(C_{13} \oplus C_{23} \oplus C_{33}) = \overset{v}{C}_1:\overset{v}{C}_2:\overset{v}{C}_3$$

in which $\overset{v}{C}_j$, $j=1,2,3$, is the modulo 2 sum $C_{1j} \oplus C_{2j} \oplus C_{3j}$.

Clearly, the weight, wt, of C, is

$$\mathrm{wt}\left(\overset{v}{C}_1:\overset{v}{C}_2:\overset{v}{C}_3\right) = \mathrm{wt}\left(\overset{v}{C}_1\right) + \mathrm{wt}\left(\overset{v}{C}_2\right) + \mathrm{wt}\left(\overset{v}{C}_3\right)$$

Now, as has been explained previously, each of the submatrices GM_{ij} consists of three parts, of two parts or of a single part. Hence it follows that:

$$C_{ij} = C_{ij1} \oplus C_{ij2} \oplus C_{ij3} \tag{2.3}$$

where C_{ijp} is a linear combination of rows taken from the corresponding submatrix GM_{ijp}. For those submatrices consisting of three parts, for example $[RM(r, m-1)]$, equation (2.3) applies directly. For those submatrices consisting of two parts, for example $[RM(r-1, m-1)]$ and $[RM(r-1, m-1)]^*$, C_{ij3} does not exist. For submatrices, such as $[RM(r-2, m-1)]$, which consist of one part, only C_{ij1} exists. For clarity of exposition the situation is illustrated in Fig 2.2. It is important to note that the codeword construction in terms of the nine parts $C_{11},\ldots,C_{33}$ is unique, as is the decomposition into the parts C_{ijp}.

From the way in which the codewords of the new code are constructed it follows that, in terms of the notation of above,

$$C_{11p} = C_{12p} = C_{13p} \; ; \; p = 1,2,3 \tag{2.4}$$

and also $$C_{221} = C_{231}. \tag{2.5}$$

LEMMA. $C_{222} = C_{232}$ if and only if $C_{222} = C_{232} = 0$.

Proof of Lemma:

It is known from the theory of finite fields [Ref 2.3] that if a set of 2^k elements forms a group under addition, a multiplication operation can be defined, which has a unique inverse, such that the set has the structure of the field $GF(2^k)$. Now, since any codeword is obtained by multiplying an input information vector by the generator matrix of the code, it follows that the invertible operations of multiplication and addition between any two codewords can be effected by and considered to be equivalent to applying multiplication and addition between the corresponding input information vectors prior to the multiplication by the generator matrix. This follows since the operation of encoding is invertible. Thus, operations with respect to codewords can be considered with respect to the set of 2^k input information vectors.

Using the normal polynomial representation, an input information vector $i_1,i_2,\ldots,i_k$ can be represented by a polynomial of degree $k-1$, or less, that is,

$$i_1,i_2,\ldots,i_k \longrightarrow i_1 + i_2X + \cdots + i_kX^{k-1}.$$

The addition of input vectors can be carried out by simply adding polynomials element by element, and the multiplication of input vectors can be carried out by multiplying the corresponding polynomials modulo an irreducible polynomial A(x), of degree k. Such irreducible polynomials exist for degrees ≥ 2 [Ref 2.3 p. 368].

Consider any information polynomial I(X) of degree equal to or less than $k-1$ and consider an irreducible polynomial A(X) of degree k. Consider also another polynomial B(X) of degree equal to or less than $k-1$. This can be considered to be another information vector. It can be proved that (see page 45):

$$\Big(I(X) \cdot B(X)\Big) \bmod A(X) \neq I(X) \tag{2.6}$$

Now, $\Big(I(X) \cdot B(X)\Big)$ mod A(X) forms the same space of polynomials as the

set I(x) itself, and hence the set $X \cdot I(X)$ when used in conjunction with a generator matrix GM generates the same code as the input information vector set I(X). However, the codeword generated by $X \cdot I(X)$ is different from that generated by I(X) itself. This follows directly from (2.6).

Consider now the creation of a matrix GM* which has the following property: When GM* is used as generator matrix the codeword generated by an information vector I(X) is the same as that generated by $X \cdot I(X)$ when the generator matrix GM is used.

On account of the linearity of the codes involved, GM* has the desired property if it possesses it for the basis set of input information vectors $1, X, \ldots, X^{k-1}$. Consider first the basis vectors $1, X, \ldots, X^{k-2}$. Now these information vectors, when multiplied by X, become $X, X^2, \ldots, X^{k-1}$ respectively, and when using the generator matrix GM they result in codewords which correspond to the $2^{nd}, 3^{rd}, \ldots, k^{th}$ rows of GM respectively. Therefore, it is clear that a GM* having the required property for the particular set of input polynomials being considered is a matrix which has as its i^{th} row the $(i+1)^{th}$ row of GM.

Consider now the input information vector X^{k-1}. When this information vector is multiplied by X and reduced modulo:

$$A(X) = \alpha_0 + \alpha_1 \cdot X + \cdots + \alpha_{k-1} \cdot X^{k-1} + X^k$$

the result is a polynomial $\alpha_0 + \alpha_1 \cdot X + \cdots + \alpha_{k-1} \cdot X^{k-1}$ and hence if the matrix GM* is to have the desired property, it must have as its k^{th} row the n-tuple

$$\alpha_0(\text{row } 1) \oplus \alpha_1(\text{row } 2) \oplus \cdots \oplus \alpha_{k-1} \cdot (\text{row } k) \tag{2.7}$$

where (row t) in (2.7) is the tth row of GM. This completes the proof of the Lemma. □

It now remains to be shown that the matrix (2.2) generates a code having minimum weight $1.5 \cdot 2^{(m-r)}$.

The minimum weight of the new class of codes can be determined by considering the minimum weight associated with certain subsets of codewords of

the code. These various subsets can be considered as various cases with the subsets being chosen so as to cover the whole code. Let $wt(C_{ij})$ and $wt(C_{ijp})$ respectively denote the minimum weight of the vector associated with the previously defined linear combinations C_{ij} and C_{ijp} of rows.

CASE 1: $\overset{v}{C}_1 = 0,\ \overset{v}{C}_2 = 0\ ,\ \overset{v}{C}_3 = 0$

Clearly, this case corresponds to the zero vector, therefore $wt(C) = 0$

CASE 2: $\overset{v}{C}_1 = 0,\ \overset{v}{C}_2 = 0\ ,\ \overset{v}{C}_3 \neq 0$

Thus, $C_{11} = C_{12} = C_{13} = 0$ and $C_{22} = 0$.
In this case $C_{221} = C_{231} = 0$
and by the lemma $C_{222} = 0 = C_{232}$.
Therefore the weight of $C = wt(C_{33}) \geq 2^{(m-r)+1}$,
since GM_{33} is defined as that generator matrix which generates an RM code of minimum weight $2^{(m-r)+1}$ $\left(> 1.5 \cdot 2^{(m-r)}\right)$.

CASE 3: $\overset{v}{C}_1 = 0,\ \overset{v}{C}_2 \neq 0\ ,\ \overset{v}{C}_3 = 0$

Thus, $C_{11} = C_{12} = C_{13} = 0$ and $C_{22} \neq 0$.
$C_{23} \oplus C_{33} = 0$
$\overset{v}{C}_2 \neq 0$ means that the all zero combination is excluded and hence the minimum weight of C_{22} is equal to $2^{(m-r)}$, since GM_{22} is the generator matrix of an RM code having minimum weight $2^{(m-r)}$.
Since $\overset{v}{C}_3 = 0$ it is clear that $C_{23} = C_{33}$ and hence that
$C_{331} = C_{231} = C_{221}$ and from the lemma that $C_{232} = C_{222} = 0$.
Therefore the weight, wt(C), of $C = wt(C_{221}) \geq 2^{(m-r)+1}$,
since GM_{221} generates an RM code of minimum weight $2^{(m-r)+1}$

CASE 4: $\overset{v}{C}_1 = 0,\ \overset{v}{C}_2 \neq 0\ ,\ \overset{v}{C}_3 \neq 0$

Assuming $C_{11} = C_{12} = C_{13} = 0$ and $C_{22} \neq 0$.
By the second assumption: the minimum weight of $C_{22} = 2^{(m-r)}$, as mentioned before.
In this case $wt(C_{23} \oplus C_{33}) \geq 2^{(m-r)}$, since the code generated by GM_{33} is contained within the code generated by GM_{23}; and GM_{23} is a

matrix which generates an RM code of minimum weight $= 2^{(m-r)}$.

It is thus clear that the minimum weight is

$$wt(C) = wt(C_{22}) + wt(C_{23} \oplus C_{33}) \geq 2^{(m-r)} + 2^{(m-r)} = 2^{(m-r)+1}$$

CASE 5: $\overset{v}{C}_1 \neq 0,\ \overset{v}{C}_2 = 0,\ \overset{v}{C}_3 = 0$

Thus $C_{11} = C_{22} = C_{33} \neq 0$ and $wt(C_{1i}) \geq 2^{(m-r)-1}$

From $\overset{v}{C}_2 = C_{21} \oplus C_{22} \oplus C_{23} = 0$ and $\overset{v}{C}_3 = C_{31} \oplus C_{32} \oplus C_{33} = 0$ and since neither C_{223} or C_{233} exist, it follows that $C_{123} = 0$ and in consequence

$C_{113} = C_{133} = 0$.

Since $\overset{v}{C}_2 = 0$ and $C_{23} = 0$ it follows that $C_{12} = C_{22}$. Therefore, $C_{121} = C_{221}$; $C_{122} = C_{222}$ and at least one of C_{121} $(=C_{221})$ and C_{122} $(=C_{222})$ are non-zero.

Now, using the fact that $\overset{v}{C}_3 = 0$ it follows that $C_{131} \oplus C_{231} \oplus C_{331} = 0$. That is, $C_{131} \oplus C_{231} = C_{331}$. But, by virtue of the way the matrix (2.2) is structured, it is clear that $C_{131} = C_{231}$ and hence $C_{131} \oplus C_{231} = 0 = C_{331}$.

Again, by virtue of the structure of the matrix (2.2), $C_{132} \oplus C_{232} = 0$ and $C_{132} = C_{232}$. From above we have that $C_{122} = C_{222}$ and from equality (2.4) $C_{112} = C_{122} = C_{123}$. Thus $C_{222} = C_{122} = C_{132} = C_{232}$. But by the lemma $C_{222} = C_{232}$ implies that $C_{222} = C_{232} = 0$.

Hence; $C_{112} = C_{122} = C_{132} = C_{222} = C_{232} = 0$ which also means that (see above) $C_{121} \neq 0$ or $C_{111} = C_{111} = C_{131} \neq 0$ and it thus follows that the weight, $wt(C)$, $= wt(C_{111})$ and from before this is seen to be $\geq 2^{(m-r)+1}$.

CASE 6: $\overset{v}{C}_1 \neq 0,\ \overset{v}{C}_2 = 0,\ \overset{v}{C}_3 \neq 0$

It is proved above that when $\overset{v}{C}_2 = 0$ then $C_{113} = C_{123} = C_{133} = 0$ and so $wt(C) = wt(\overset{v}{C}_1 : 0 : \overset{v}{C}_3) =$

$$wt(\overset{v}{C}_1) \oplus wt(\overset{v}{C}_3) \geq 2^{(m-r)} + 2^{(m-r)} = 2^{(m-r)+1}$$

CASE 7: $\overset{v}{C}_1 \neq 0,\ \overset{v}{C}_2 \neq 0,\ \overset{v}{C}_3 = 0.$

By the same arguments as before it is clear that since $\overset{v}{C}_3 = 0$ $C_{113} = C_{123} = C_{133} = 0$ and therefore

$$wt(C) = wt(\overset{v}{C}_1:\overset{v}{C}_2:0) = wt(\overset{v}{C}_1) + wt(\overset{v}{C}_2) \geq 2^{(m-r)} + 2^{(m-r)} = 2^{(m-r)+1}$$

CASE 8: $\overset{v}{C}_1 \neq 0,\ \overset{v}{C}_2 \neq 0,\ \overset{v}{C}_3 \neq 0.$

Since $\overset{v}{C}_j \neq 0$; j=1,2,3; $wt(\overset{v}{C}_j) \geq 2^{(m-r)-1}$. Therefore,

$$wt(C) = wt(\overset{v}{C}_1:\overset{v}{C}_2:\overset{v}{C}_3) = wt(\overset{v}{C}_1) + wt(\overset{v}{C}_2) + wt(\overset{v}{C}_3) \geq 2^{(m-r)-1} + 2^{(m-r)-1} + 2^{(m-r)-1} = 1.5 \cdot 2^{(m-r)}$$

This last case, which completes the proof, also shows that the minimum weight is not greater than $1.5 \cdot 2^{(m-r)}$. It is interesting to note that the last case is the only case in which the minimum weight is equal to $1.5 \cdot 2^{(m-r)}$, and all the other cases have a minimum weight of $2^{(m-r)+1}$ which has implications relating to the weight distribution of the code.

2.4 DIMENSIONALITY CONSIDERATIONS

The new 1.5 RM codes can be considered within the sequences of RM codes of constant length-to-distance ratio n/d. Fig 2.3 shows the sequence for three different length-to-distance ratios, n/d, with, for each length-to-distance ratio, the code dimension, k, being plotted as a function of codeword length, n. The axes of the figure are plotted to a logarithmic scale and the half-rate locus is shown, as a heavy dotted line. The loci of other constant rates are lines parallel to the 1/2 rate locus, with rates $< 1/2$ being to the right of the 1/2 rate locus and values in excess of 1/2 being to the left of the 1/2 rate locus.

It is important to consider the dimensionality of the new class of codes and its relation to the dimension of members of the RM class of codes.

The dimension, k_m, of the RM(r,m) code is $k_m = \sum_{i=0}^{r}\binom{m}{i}$ and the dimension, k_{m+1}, of the RM(r,m+1) code which has the same length-to-distance ratio as the RM(r,m) is

$$k_{m+1}= \sum_{i=0}^{r}\binom{m+1}{i}.$$

Using the relation $\sum_{i=0}^{r}\binom{m+1}{i} = 2\cdot\sum_{i=0}^{r}\binom{m}{i}-\binom{m}{r}$

it follows that

$$k_{m+1} = k_m - \binom{m}{r}$$

The corresponding rates for the codes are

$$\frac{\sum_{i=0}^{r}\binom{m}{i}}{2^m} \quad \text{and} \quad \frac{2\cdot\sum_{i=0}^{r}\binom{m}{i}-\binom{m}{r}}{2^{m+1}}$$

respectively. It is thus clear that, for a given length-to-distance ratio, the rate of an RM code decreases by a value of $\binom{m}{r}\cdot\frac{1}{2^{m+1}}$ as the length is doubled.

In the case of the dimension, $k_{1.5m}$, of the new codes it is clear from the way they are constructed that:

$$k_{1.5m}= k_m + \sum_{i=0}^{r-2}\binom{m-1}{i}= 1.5\cdot k_m - 1/2\cdot\left(\binom{m}{r}+\binom{m-1}{r-1}\right)$$

Thus , as the length of the RM(r,m) code increases by a factor of 1.5, the rate

decreases by $\frac{\frac{1}{2}\cdot\left(\binom{m}{r}+\binom{m-1}{r-1}\right)}{1.5\cdot 2^m}$ and therefore the same general

pattern of behaviour is observed as with the RM class of codes.

Comparison of the rates of the RM(r,m) and the 1.5 RM(r,m) codes shows that, for a given length-to-distance ratio, the rate of the 1.5 RM(r,m) code is less than the rate of the RM (r,m) code. A similar comparison of the values of the rate of the RM(r,m + 1) code with that of the 1.5RM(r,m) code shows that the difference, ΔR, in rates is

$$\begin{aligned} \Delta R &= \text{Rate of RM(r,m+1)} - \text{Rate of } 1.5 \cdot \text{RM(r,m)} \\ &= \frac{1}{6 \cdot 2^m} \binom{m}{r} \left(2 \cdot \frac{r}{m} - 1 \right) \end{aligned}$$

The various points raised above are illustrated in Fig 2.3. From the figure it is clear that for both the original class of Reed-Muller codes, and the new class, as the codeword length is increased, for a given length-to-distance ratio, the rate of the code decreases. This is shown by the fact that the locus for both the original RM codes and the new class of codes starts to the left of the 1/2 rate locus and crosses it as n increases.

For a given length-to-distance ratio, the new class of codes have the property that the rate of the next longer RM code is less than that of the (immediately previous) new code, if both rates are less than 1/2. The converse is true if the rates are greater than 1/2. The relative rates of the codes can be obtained by measuring the perpendicular distance of the code (the point) under consideration from the 1/2 rate locus.

2.5 DECODING BY MAJORITY LOGIC

The new class of codes described in the previous sections of this chapter are linear codes and hence can be decoded using the familiar well known standard array technique. This technique is, however, inefficient and effectively involves a look-up procedure, the complexity of which grows exponentially with the dimension of the code, or with the codeword length.

An alternative and very efficient approach to the decoding of the class of Reed-Muller codes is that known as majority logic decoding (Ref [2.4]). In this

section consideration will be given to the question of whether it is possible to decode the new class of codes with majority logic decoding (MLD).

In the discussion given earlier (Section 2.2) it was argued that the rows of the generator matrix of the RM codes can be re-arranged so that the matrix can be partitioned in the manner illustrated in matrix 2.1. The resulting parts of the matrix generate the RM codes of length and order shown. Now, a particularly valuable alternative way of rearranging the rows which leads to a simple explanation of the MLD process is the following. Let the rows of the generator matrix of the RM code be rearranged so that the row with a one in the leftmost position is placed first, the row with a one to the next leftmost position is placed next, and so on. This rearranging process is illustrated in Fig 2.4 for the RM(3,5) code. With this rearrangment it is possible to successively partition the code so that each part generates a half-length code relative to the previous partitioning and the order of this code is equal to, or one less, than the order of the code associated with the previous partitioning. Since the parts of the so partitioned matrix are of the same form as the overall matrix, this process, which is illustrated in Fig 2.4, can be repeated until a matrix of a code of zero order is reached, at which point the matrix can only be further partitioned by a vertical partition into two parts.

By using the above induction approach to the GM it is easy to explain how the received version of a codeword transmitted can be majority logic decoded to obtain an estimate of the last digit (bit) of the k-digit message word. It will be recalled that, quite generally, the encoding process amounts to the association of every message word with a sum of the rows of the generator matrix; with the rows used in the summation corresponding to 1's in the k-digit message word. A message word can be divided into two parts, an "upper" and a "lower" part, which can be associated respectively with the upper and lower parts of the GM. Consider now a codeword, and let it be divided into two equal parts. It is clear that each of the parts is the sum of the rows of the corresponding submatrices of the rearranged matrix which is illustrated in Fig 2.4. Now it will be remembered that the rows corresponding to the upper part of the message word are the juxtaposition of two identical halves and the rows corresponding to the

lower part have a first half which is equal to the all-zero vector and a second part which is equal to a sum of rows taken from the generator matrix of the RM(r−1,m−1) code and hence if the two halves of a codeword are added together, then those elements arising from the two equal upper submatrices cancel each other, and since the lower left submatrix is zero, the result of the addition depends only on the lower right matrix. In short, the result of the addition corresponds to a codeword of an RM code with half length and order reduced by 1, that is, it corresponds to a codeword of the RM(r−1,m−1) code. Moreover, this codeword arises from the lower part of the k-digit message word. From the inductive nature of the code it is clear that if the process is repeated r times, then the resultant vector is a codeword of zero order, that is, is an all-1 or all-0 codeword, depending upon whether the last message digit(bit) of the message word was 0 or 1 respectively.

When the two halves of a codeword of length $n = 2^m$ are added together, every digit of the sum is made up by the addition of two digits, one taken from each of the two halves of the codeword and hence the ith digit, $0 \leq i \leq 2^{m-1}$, of the sum is equal to the modulo-2 sum of the ith digit and the $(2^{m-1}+i)$th digit, that is, is the sum of all digits in positions j, with j being such that it is equal to i when it is taken modulo-2^{m-1}. After the above procedure has been repeated r times, the ith digit of the resultant word, the word length being 2^{m-r}, is equal to the modulo-2 sum of all those digits in positions j, with j being such that it is equal to i when j is taken modulo-2^{m-r}. Thus, 2^{m-r} sums are formed and each sum corresponds to one of the digits of a zero-order codeword. In the absence of noise all digits must be identical and equal to the last digit of the message word. Apart from the fact that the position of the row in the GM of the code, and therefore the position of this digit in the message word, has changed, the above procedure is the same as the conventional majority logic decoding of this particular digit .

Consider now a received word V and let this be the sum of the transmitted codeword C and an error word e of weight t, i.e. t errors have occurred. Since the "adding-the-halves" procedure described above is a linear operation, its effect can be evaluated separately for the codeword and for the error word. On

applying this procedure once to a codeword, it results in a codeword of half length and order reduced by 1. This resulting word belongs to a code which has the same minimum distance as the original code. It is clear that the application of the process to the error word cannot result in an increase in its weight (the number of errors) since in applying the procedure, each and every digit of the codeword is involved in the formation of one and only one of the digits of the final 0-order word. This means that every codeword resulting from the "adding-the-halves" procedure is drawn from a code that has the same minimum distance as the original code, and since the weight of the error word is not increased, the error correction capability of the original code is maintained with respect to the reduced length and order codeword. At the final stage, after r "adding-the-halves" operations, an all-1's or an all-0's codeword of length 2^{m-r} is obtained with at most t errors embedded in it. If $t \leq 2^{m-r-1}-1$, then the errors can be corrected, and if $t=2^{m-r-1}$ then error detection can be carried out, as guaranteed by the minimum weight of the original code.

Suppose now that, by using the above procedure, the last digit of the message word has been determined (estimated), which means that the contribution of the last row of the generation matrix to the formation of the transmitted codeword has been estimated. Now, provided that the number of errors does not exceed the error correcting capability of the code, the above estimation of the contribution of the last row of the RM will be correct and the effect of the last row of the generator matrix can thus be removed by subtracting from the original codeword the last row of the GM, in the case when the last message digit is 1, or by leaving the codeword unchanged if the last message word is 0. In both cases the result is a codeword, C', of the same RM(r,m) code, bearing the same errors as the original codeword word C. Furthermore, the codeword C' is a sum of the first $k-1$ rows of the generator matrix, with the rows used in summation corresponding to the 1's of the first $k-1$ digits of the message word. (Remembering that the generator matrix is in the rearranged form described earlier). Now, after $r-1$ applications of the "adding-the-halves" procedure to the codeword C', a codeword of the RM(1,m−r+1) code is obtained and this codeword consists of two identical halves, with each half belonging to an RM(1, m − r) code. These two halves are

the result of summing up those rows of the matrix corresponding to the existence of 1's in the m−r+1 digits immediately before the last message bit. By applying the same "adding-the-halves" procedure to each of these codewords (the halves), two new vectors of order 0 are also obtained. These two vectors are again both all-1 or all-0, depending on whether the (k−1)th digit of the message word is 1 or 0. The minimum weight of the RM(1,m−r) code to which the two codewords (the halves) belong, is half of the minimum weight of the original code, that is, is 2^{m-r-1}, and the length of the 0-order vector derived from the above "adding-the-halves" procedure is also 2^{m-r-1}. However (in the absence of errors), when the resultant vectors are put in juxtaposition, an all-0 or all-1 word of length 2^{m-r} is obtained. As before, since the procedure cannot increase the number of errors, the corresponding (k-1)th digit of the message can now be decoded correctly provided that the number of errors occurring is equal to or less than the number of errors guaranteed correctable by the original code and, as with the kth message digit, the contribution of the (k−1)th row resulting from the (k−1)th message digit can now be removed from the original received word.

The procedure of "adding-the-halves" of the codeword in order to find the digits of a zero-order and the use of this codeword to obtain an information digit and the subsequent subtraction of the corresponding row from the original matrix is similar to that described conventionally as majority logic decoding in which the digits of certain groups of positions are added and a majority vote among their sums is taken and the so extracted rows are subtracted from the received version of the original codeword. The interpretation of MLD in terms of the "adding−the−halves" process is particularly convenient in that it is easy to relate the technique to the inductive nature of the class of the RM codes and this in turn leads to a simple way of investigating the applicability of MLD to the new class of 1.5 RM codes which are themselves inductively related to the class of RM codes.

As it was explained in Section 2.3, every row of the generator matrix of an 1.5 RM code and hence every codeword of the code, can be divided into three equal-length "thirds". And, every codeword can also be considered to be the

sum of three parts, "upper", "middle" and "lower", with each of the parts being a sum of rows from the corresponding parts of the generator matrix.

With a view to investigating the applicability of MLD to the new class of codes, consider first how, in the absence of errors, the contribution of the RM(r−2,m−1) to a codeword, C, that is, the contribution C_{33}, can be derived from the three parts, $\overset{v}{C}_1, \overset{v}{C}_2, \overset{v}{C}_3$ of the codeword C. From the construction of the generator matrix of the 1.5 RM codes it is easy to see that, in terms of the previously used notation:

$$C_{33} \equiv C_{331} = \overset{v}{C}_1 + \overset{v}{C}_3 + f(\overset{v}{C}_2) + f(\overset{v}{C}_1) \tag{2.8}$$

where f stands for the mapping $GM_{22} \rightarrow GM_{23}$ defined in Section 2.2, and since C_{33} is a codeword belonging to the RM(r−2,m−1) code, MLD can be applied in order to decode the received codeword. In the presence of noise, however, the situation is different and it is necessary to examine whether C_{33} can in fact be derived from a noisy codeword C when the number of errors is equal to or less than the guaranteed error correcting capability of the code.

Now, since the error correcting capability of the RM(r−2,m−1) code is greater than the error correction capability of the 1.5RM(r,m) code associated with the matrix (2.2), it is necessary to ask whether it is possible to process the received version of the transmitted codeword in such a way that, given that the number of errors in the received codeword is equal to or less than the error correcting capability of the 1.5RM(r,m) code, the noisy version of C_{33}, with C_{33} being denoted as in (2.8), contains a number of errors within the error correcting capability of the RM code.

The received word, V, can be considered to be the sum of the transmitted codeword C and the error vector e, that is V = C+e. If V (and hence e) is divided into three equal-length "thirds", then the three thirds of V can be written as:

$$V_1 = \overset{v}{C}_1 + e_1,\ V_2 = \overset{v}{C}_2 + e_2,\ V_3 = \overset{v}{C}_3 + e_3 \tag{2.9}$$

where e_i, i=1,2,3 represents respective parts of the error vector. In the manner given in (2.8) a noisy version of C_{33} can be formed as:

$$V_1+V_3+f(V_2+V_1)=C_{33}+e_1+e_3+f'(e_1)+f'(e_2) \tag{2.10}$$

where f' is some extension of f in the vector space of the binary (2^{m-1})-tuples, that is, f': $V_{2^{m-1}} \Longrightarrow V_{2^{m-1}}$, is some linear mapping from the space of the code onto itself that is consistent with the effect of the function f in its own domain (the code). It is clear that what matters is the weight of the error pattern $e_1+e_3+f'(e_1)+f'(e_2)$ since it is this that must be equal to or less than the correction capability of the RM(r−2,m−1) code. Clearly the choice of the extension f', of the function f, influences the weight of the resulting error pattern.

The general question is whether a mapping f' exists which is such that

$$wt\left(e_2\right) \geq wt\left(f'(e_1)+f'(e_2)\right) \tag{2.11}$$

Now, it can be shown that no mapping f' satisfying (2.11) exists which is such that each codeword (element) of the RM(r−1,m−1) code is mapped into another element of the same code. This shows that expression (2.11) cannot be satisfied and hence certain patterns or errors of weight equal to or less than the number guaranteed correctable by the 1.5 RM code, cannot be MLD'd.

With a view to answering the question of whether the new class of codes can be partially decoded using majority logic decoding, the following argument can be used.

If a mapping could be shown to exist for the codes which is such that

$$wt\left(e\right)=wt\left(f'(e)\right) \tag{2.12}$$

for any vector e, then it can be shown that MLD can be used to correct most, but not all, the error patterns that are of weight equal to, or less than, that guaranteed correctable by the code, i.e. that the majority of error patterns of weight $\leq 1.5 \cdot 2^{m-r-1}-1$ can be MLD'd.

Consider an error vector e of weight $wt(e) \leq 1.5 \cdot 2^{m-r-1} - 1$. Now if the first part, e_1, is of weight equal to or less than 2^{m-r-2}, then it can be shown that

$$2 \cdot wt(e_1) + wt(e_2) + wt(e_3) \leq 2^{m-r} - 1 \tag{2.13}$$

Now,

$$wt\Big(e_1 + e_3 + f'(e_1) + f'(e_2)\Big) \leq wt\Big(e_1\Big) + wt\Big(e_3\Big) + wt\Big(f'(e_1)\Big) + wt\Big(f'(e_2)\Big)$$

and hence, provided (2.12) is satisfied, it follows that

$$wt\Big(e_1 + e_3 + f'(e_1) + f'(e_2)\Big) \leq wt\Big(e_1\Big) + wt\Big(e_3\Big) + wt\Big(f'(e_1)\Big) + wt\Big(f'(e_3)\Big)$$
$$= 2 \cdot wt\Big(e_1\Big) + wt\Big(e_2\Big) + wt\Big(e_3\Big).$$

which means, by virtue of (2.13) that

$$wt\Big(e_1 + e_3 + f'(e_1) + f'(e_2)\Big) \leq 2^{m-r} - 1$$

and hence the error patterns are correctable by MLD.

Since the proportion of the correctable error patterns having a number of errors in e_1 exceeding 2^{m-r-2} is small, it follows that, provided a mapping can be found which satisfies expression (2.12), the majority of the error patterns can be corrected by MLD. An example illustrating the above argument concerns the 1.5 RM(2,4) code. For this code a mapping can be found which satisfies (2.12), and it can be shown that 272 out of 300 correctable error patterns can be handled successfully by MLD.

It is important to note that not all the codes in the new class are such that a mapping satisfying (2.12) can be found. For example, in the case of the 1.5 RM(3,5) code it is easy to see that no such mapping does in fact exist. The matrix GM_{222} of this code generates one, and only one, vector of weight 10. This vector is the sum of all row vectors of GM_{222}, that is the vector 0001 0111 0111 1110. Therefore, it is impossible to map this vector, using any mapping satisfying (2.12) onto another vector of the code. Thus it is clear that the procedure for this code which is associated with eq. (2.10) increases the weight of some error patterns to a point at which the RM(r−2,m−1) code is unable to correct them.

The following is a specific illustration of this effect of increasing the weight of some error patterns and concerns one of the 28 non-correctable error patterns mentioned above in relation to the 1.5RM(2,4) code (see Fig (2.1) for the generator matrix). The question of whether or not the 12th message digit (and hence the 12th row of the generator matrix) participates in the formation of a codeword, can be answered by examining the 0-order vector C_{33}. The 8 digits which constitute this vector are obtained respectively as the modulo-2 sum of those digits in the following 8 groups, where the integer i; i=1,2,...,24 indicates the position of the digit in the codeword:

(9,17), (3,2,11,18), (6,3,14,19), (8,4,16,20),

(2,5,10,21), (4,6,12,22), (5,7,13,23), (7,8,15,24)

Although there are more "voters" than the minimum weight (=6) , not all patterns of 2 errors can be corrected since, for example, two errors in the first part, e_1, affect the decisions arising from 3 or 4 of these sets. For example, if e_1=0100 0001 and f is extended to f' by keeping all the other basis vectors unchanged (considering as basis the rows of the GM of the trivial RM(3,3) code), then the 2nd, 4th, 5th and 8th sets are contaminated, resulting in an error pattern 0110 1001 being added to C_{33} in the procedure associated with Eq (2.10).

2.6 CONCLUSIONS

In this chapter consideration has been given as to how redundancy can be added to a communication signal in a well understood algebraic way in order to improve the reliability of communication over a noisy channel.

A new class of codes, referred to as 1.5 RM codes, has been developed and described. The codes are such that they add to the range of available lengths and distances as compared with those that are currently available with the

existing class of RM codes. More precisely, the new codes have the same length-to-distance ratio as certain members of the existing class of Reed-Muller codes, and for those cases in which the length-to-distance ratio is the same as for those members of the existing class of RM codes, the new codes are available with lengths and dimensions which lie between those values that are available with the existing class of RM codes. This feature is illustrated in Fig 2.3.

Consideration is also given to the question of whether the new codes can be majority logic decoded. It is shown in the chapter that the codes as formulated and synthesized cannot be decomposed in such a way as to render them majority logic decodable.

An interesting question, worthy of further investigation, is whether it is possible, using the inductive ideas set out in the chapter, to synthesize the new codes in such a way that they can be decomposed into lower order 1.5 RM codes in the manner illustrated in matrix (2.1) for the existing class of RM codes. If such a set of codes could be found, then it might well be possible to decode overall by considering the reduced length codes and applying some relatively easy decoding technique, such as, for example, a simple look up procedure, to these reduced length subcodes. (see also pages 144 and 145b)

A basic aspect of error correction in general and error correcting codes in particular is the use that is made of redundancy. With error correcting codes, the redundancy is added in a controlled manner and as much use as possible is made of algebraic structure in order to render the process of decoding, which amounts essentially to the processing of a noisy version of a signal containing redundancy in order to obtain an estimate of the message signal.

In nature, extensive use is also made of redundancy, with natural languages involving aspects such as speech and writing (printing) being important familiar examples. In cases such as these, a fundamentally different aspect exists in comparison with the redundancy associated with the error correction codes etc. When dealing with naturally occurring redundancy, such as in speech and writing (printing), no clearly defined structure can be associated with the redundancy. The redundancy may take a variety of forms and it is, more often than not, very difficult to define what part of the pattern (speech, printing etc)

is actually signal and what is redundancy. In the next chapter aspects associated with the processing of two-dimensional images are considered. The process of information retrieval (often referred to as data compression) is considered as a process of retrieval by successive entropy reduction, with care being taken at each stage to retain the essential characteristics of the information signal. Specifically, in the chapter which follows, attention is focused on redundancy removal and the encoding of an image prior to the recognition stage in a pattern recognition system.

Supplement to page 23.

When $r=0$, the second row of submatrices of (2.1) does not exist and when $r=m$, (that is, when the code is the whole space), then the matrix $[RM(r-1, m-1)]$ is used in places where the matrix $[RM(r-1, m-1)]$ is referred, in (2.1).

Proof of (2.6)

Assuming $I(X)\neq 0$; $B(X)\neq 0,1$ and given that the degree of $I(X)$ is less than that of $A(X)$

$\big(I(X).B(X)\big) \bmod A(X) = I(X) \Rightarrow \big(I(X).B(X) - I(X)\big) \bmod A(X) = 0$

which $\Rightarrow \Big(\big(B(X)-1\big).I(X)\Big) \bmod A(X) = 0.$

And, $\Big(\big(B(X)-1\big).I(X).I(X)^{-1}\Big) \bmod A(X) = 0 \Rightarrow$

$\big(B(X)-1\big) \bmod A(X) = 0$ which $\Rightarrow B(X)=1$ and this is a contradiction

(where $\Rightarrow$ denotes imply)

REFERENCES

[2.1] F.J.McWILLIAMS, N.J.A.SLOANE, *The Theory of Error-Correcting Codes, Vols I and II, North-Holland, Amsterdam*, 1977.

[2.2] W.W.PETERSON and E.J.WELDON, *Error-Correcting Codes*, M.I.T. Press, Cambridge, Mass 1972.

[2.3] J.B.FRALEIGH, *A First Course in Abstact Algebra*, Addison-Wesley, Reading, Mass, 1967

[2.4] R.E.BLAHUT, *Theory and Practice of Error Control Codes*, Addison-Welsey, Reading, Mass, 1983

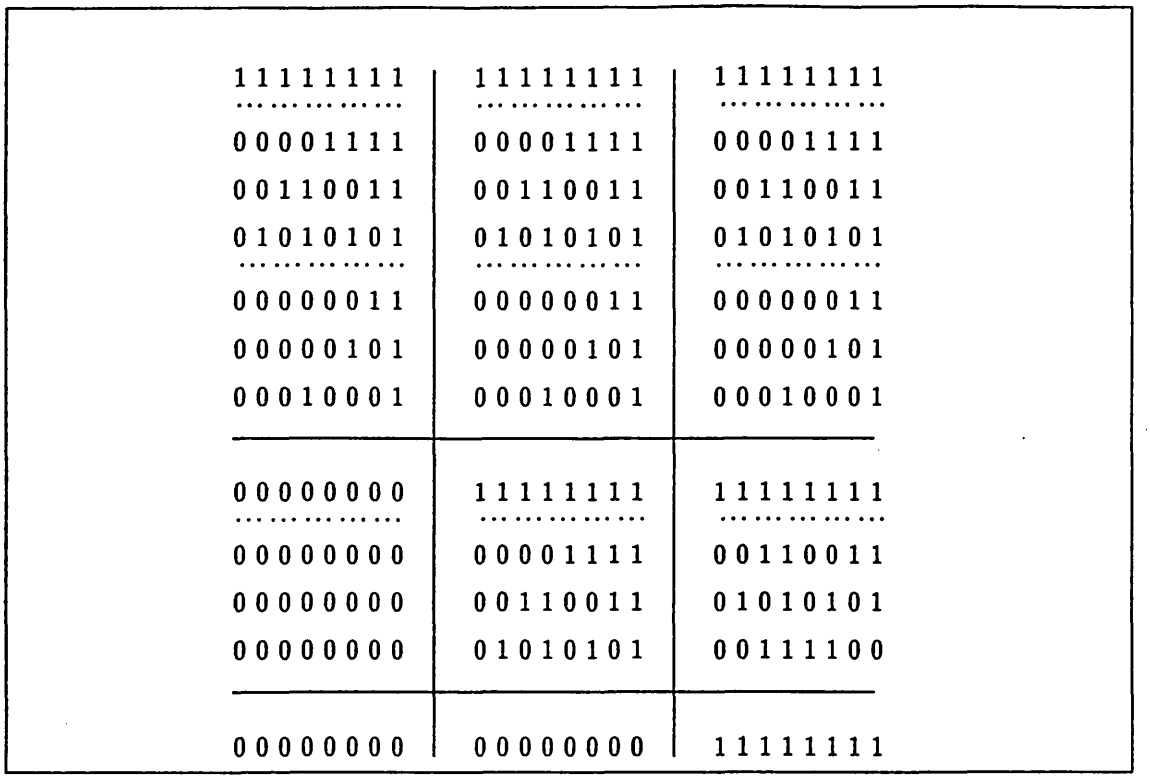


Fig 2.1

GENERATOR MATRIX OF THE 1.5 RM(2,4) CODE

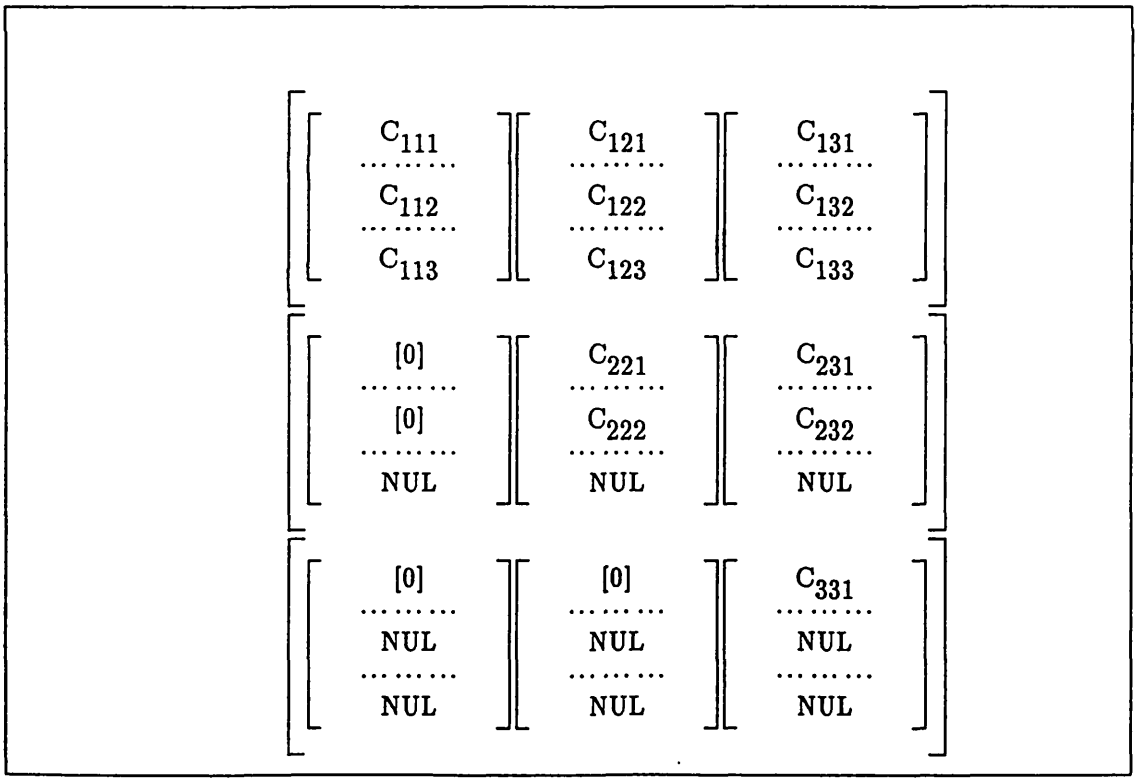


Fig 2.2

DECOMPOSITION OF THE GENERATOR MATRIX OF AN 1.5 RM CODE INDICATED BY THE CORRESPONDING PARTS C_{ijp}

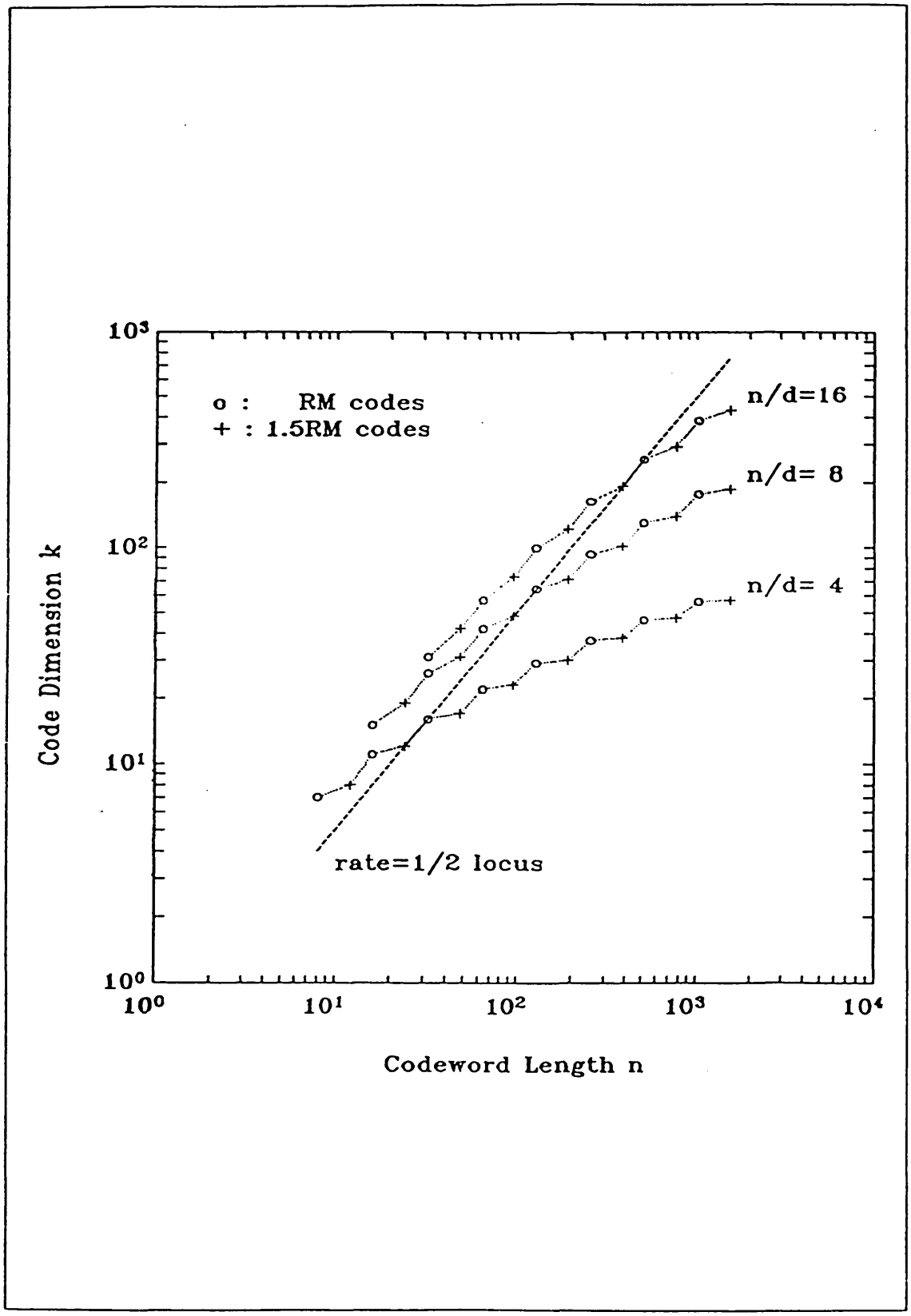
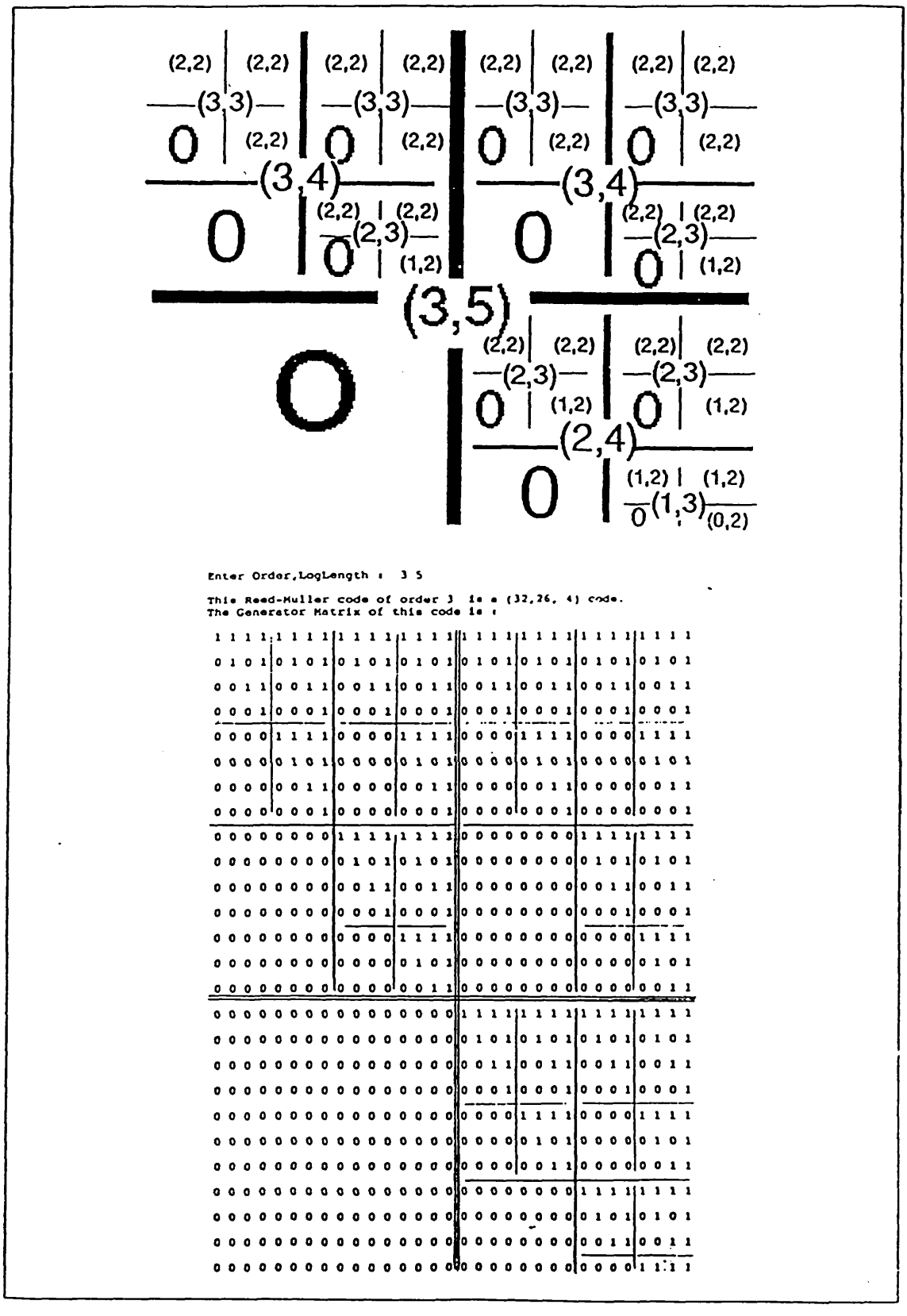


Fig 2.3

DIMENSION VS LENGTH ILLUSTRATION OF RM AND 1.5 RM CODES OF LENGTH UP TO 2^{10} AND $1.5 \cdot 2^{10}$ RESPECTIVELY AND ORDER 2,3,4 (LENGTH/DISTANCE 4,8,16).




```
Enter Order,LogLength : 3 5

This Reed-Muller code of order 3 is a (32,26, 4) code.
The Generator Matrix of this code is :
```


Fig 2.4

GENERATOR MATRIX OF THE RM(3,5) CODE IN "INDUCTIVE FORM" AND SUCCESIVE PARTITION OF IT INTO GM'S OF RM CODES OF HALF LENGTH AND SAME OR LOWER ORDER

Chapter 3
The Coding Of Handprinted Characters

CHAPTER 3
THE CODING OF HANDPRINTED CHARACTERS

3.1 INTRODUCTION

In the previous chapter the situation was considered in which redundancy was deliberately introduced into a signal with a view to improving the reliability of communicating that signal over a noisy communication channel. The redundancy was added in a controlled manner, using algebraic structure, so as to achieve a guaranteed degree of protection against transmission errors.

In this chapter redundancy is again the main focus of attention, but with the aim being to remove rather than add redundancy. The "signals" considered are the two-dimensional signals that are the alphabetic characters of the English Language, and the redundancy involved is that which is introduced naturally in the course of writing, or printing, the characters.

An important topic which is likely to play a key role in developing information technology systems is that of the automatic (machine) recognition of characters. This is a topic that has received considerable attention for more that 30 years and it is today gaining importance on account of developments in information technology, and the associated need to improve man/machine interaction.

The essential task in character recognition is to go from a two-dimensional image of a character to a unique representation (usually in the form of a binary word) of the character. A well known example of a character recognition system is that involving a human being, who reads a sequence of characters, one at a time, and then produces some coded version, such as an ASCII code word for each of the characters read.

The redundancy involved in respect of the set of characters that form the English alphabet is complex and difficult to define, and this difficulty is at the heart of the character recognition problem. The redundancy can take the form

of unimportant variations between members of the class. For example, variations in style, size, orientation etc. of a letter of the alphabet; and variations in line thickness etc.

In this chapter the inherent redundancy in the characters which form the English alphabet is considered. The removal of this redundancy, and the obtaining of a set of features, is an essential step in the character recognition process. The character recognition process can be considered, in its simplest form, as a two-stage process: The first stage is concerned with the removal of the redundancy and the extraction of features, and the second stage with the classification (the recognition) of the input from the extracted features.

At a higher level, the character recognition process can be considered to involve a third stage in which the natural redundancy contained within the language (context) is used to aid the classification process. For example, it may be that in the second stage, the classification stage referred to above, errors are made, or a soft decision approach is involved. In situations like this context may then be used in the next stage to complete the recognition process.

In this chapter questions relating to the processing of images so as to remove redundancy, and further processing to obtain a coded representation of the characters for use in the recognition stage are considered.

3.2 REDUNDANCY CONTAINED IN CHARACTER IMAGES

The information relating to a character is contained in an image that can be interpreted as a function defined over a two-dimensional domain. In the case of hand-printed characters the domain is continuous. The first stage of processing such an image generally involves sampling with the result that a discrete two-dimensional image is obtained. The elements of the two-dimensional image are generally referred to as picture elements or pixels, and it is not unusual to find resolutions ranging from 20x20 up to 128x128 pixels per character. The intensity of a pixel is again, in general, continuous in that it may

take any value within certain range limits. Such continuous pixel intensities are usually converted into a discrete form through the process of quantization. If an image pixel can take only one of 2^l levels, then it can be represented by l bits. In summary, if an image is represented as an NxN pixel image and each pixel is quantized to l bits, then the image is represented by N^2l bits. This N^2l bit representation is capable of denoting any change in size, position, orientation, or intensity of an image; and these variations may in fact be of considerable significance. For example, an individual can be identified from samples of his/her hand printing. In the case of character recognition, however, the aim is much more "coarse" in the sense that the object is to identify individual characters that belong to a particular class (or subclass) irrespective of variations between members of the class. In short, the aim of the recognition process is to represent a particular class, or subclass, of images by a unique binary digit representation. For example, the set of letters of the English Language can be represented in ASCII code using less than one byte per letter. This process of going from N^2l bit representation of a character to the simple word representation of the class constitutes a large compression, and it can be considered as a major reduction in redundancy. The process involved is, however, information lossy in that it is not possible to go back from the short length word appearing at the output of the recognition system to the N^2l image present at its input.

In order to consider the question of redundancy removal and deal with the various stages in the process it is helpful to consider in a little more detail how, disregarding variations in character style etc., the redundancy arises. In respect of hand and machine-printed material it is now common practice to represent characters as a set of N^2 (or more often NxM) binary pixels, that is, images are represented in black/white form. If a count of the relative frequency with which the white and black pixels are used is carried out then it is found that, in text, approximately 80% of the pixels are white, when resolutions of the order of 64x64 are involved. If, as a first approximation, the statistical dependence between pixels is neglected then a simple entropy calculation shows the image to

contain approximately 30% redundancy (ie the entropy is approximately 0.7 bits per pixel). This is, however, an underestimate of redundancy since factors such as the dependence between pixels and the constraints associated with hand and machine produced images account for a further significant increase of redundancy. With resolutions of the order of 64x64 pixels lines tend to have a thickness of 3 or more pixels. The actual thickness of the line carries little information of value in the recognition process and hence it constitutes a further redundancy. If lines are reduced so as to have a single pixel width, then the entropy calculation reveals that redundancy is approximately 70%. Still further redundancy is involved on account of the fact that characters are comprised of straight lines and curves which means that preferred directions are involved. Thus, in situations similar to that illustrated in Fig 3.1, the probability that the pixel marked "?" will be a black is very high. Stated another way, it is clear that variations such as ripples (or random fluctuations) observed in respect of the lines and curves that constitute the character are unimportant. And, as far as the recognition process is concerned, these variations can be ignored.

A modern dot printer of medium quality uses a 7x9 matrix of dots to represent characters, and although it contains none of the variations in respect to line ripples, and although the lines are of unit width, it can still be regarded as containing significant redundancy in the sense that from the point of view of handling, processing and transmitting characters as few as 7 bits are necessary, rather than the 63 involved in the dot printer. It is, however, clear that insofar as the actual portrayal of characters in the normal visual form is concerned the dot printer operates at the limits of acceptability and therefore contains little redundancy. In this thesis the number of digits necessary to represent a class will be considered to be that number which is strictly necessary for representing the character so that it can be stored, transmitted, etc. In Section 3.3 which follows the essential steps involved in converting a character into a digital representation that can be used in the recognition process are considered. The steps examined are (i) Thresholding, (ii) Skeletonization and (iii) Chain Coding. The material presented in Section 3.3 is mainly a reconsideration of existing

techniques but, in places, new and interesting ways of looking at this material are given and some extensions of the techniques are presented. Finally in the Section 3.4 a new combined approach to skeletonization and coding is put forward.

3.3 THE CODING OF LINE IMAGES

3.3.1 Thresholding

As was explained in the previous section a common step involved in the processing of character images (or for that matter more general images) is the step involved in going from a multi-level (grey scale) representation of a pixel to a binary word representation. This process has come to be known as "thresholding".

The process of thresholding is important in that the information so derived is used in subsequent recognition, and, as the process is irreversible i.e. is information lossy, care must be taken to ensure that the process preserves adequate information for use in further processing. The thresholding technique may be designed to take advantage of the nature of the equipment involved in writing or printing, or may make use of information available about the equipment (the camera etc) used when taking signals from the "paper" and converting them into electrical form for further manipulation. Thresholding is by no means a simple matter and it is difficult to treat mathematically. Study has been given to efficient methods for carrying out the thresholding process. An interesting and important relatively recent development in thresholding is that discribed by Pal et al (Ref [3.1])

In this paper Pal et al use arguments drawn from fuzzy set analysis to arrive at a function that can be used to determine the position at which a threshold should be placed. In Ref [3.1] the following indices of fuzziness are suggested: (refer to Appendix 2 for a list of symbols)

$$v_i = \frac{2}{MN}\sum_m\sum_n \left\| \mu_X(x_{mn}) - \mu_X(x_{mn}) \right\| \qquad (3.1)$$

$$v_q = \frac{2}{\sqrt{MN}}\left(\sum_m\sum_n\left(\mu_X(x_{mn}) - \mu_X(x_{mn})\right)^2\right)^{1/2} \tag{3.2}$$

$$H(X) = \frac{1}{MN\ln(2)}\sum_m\sum_n Sn\left(\mu_X(x_{mn})\right), \tag{3.3}$$

(where Sn is Shannon's entropy function)

and the results presented relating to the thresholding of binary images indicate that for the images considered in Ref [3.1] there appears to be little difference in the threshold arrived at using the three indices referred to above.

Further insight into the automatic implementation of the technique proposed by Pal et al can be gained by considering the general form of the equations above. The consideration, of course, extends to other indices as well. Equations 3.1 to 3.3 above can be expressed in the general form

$$I(X) = \underset{X}{E}\left(F(\mu(X))\right)$$

where X is the image, $F(\mu(X))$ is the functional (the index of fuzziness in Ref [3.1]) and $\underset{X}{E}$ indicates the averaging of the function $F(\mu(X))$ over the image. In Ref [3.1] the function μ is the standard S function of fuzzy set theory and it can be defined to be $\mu(l) = S(l,l_0,\delta l)$ where l is the intensity level of a given pixel l_0 is the threshold (that is b in the notation of Ref [3.1]) and δl is the bandwidth (bandwidth is defined in Ref [3.1] to be $\Delta b = c - a$) and hence

$$\begin{aligned}
\mu(l) &= 0 && \text{for} \quad 0 \le l \le l_0 - \frac{\delta l}{2} \\
&= 2\left(\frac{l-l_o}{\delta l}\right)^2 && \text{for} \quad l_0 - \frac{\delta l}{2} \le l \le l_0 \\
&= 1 - 2\left(\frac{l-l_o}{\delta l}\right)^2 && \text{for} \quad l_0 \le l \le l_0 + \frac{\delta l}{2} \\
&= 1 && \text{for} \quad l_0 + \frac{\delta l}{2} \le l \le l_{max}
\end{aligned}$$

In Ref [3.1] it is suggested that I(X) can be evaluated without the need to refer directly to the histogram of intensities, and the form given in Ref [3.1] for I(X) suggests that it can be evaluated by considering each of the NxM pixels constituting the image and forming the appropriate summation. If, however, use

is made of the histogram then the number of computations can be significantly reduced. The way in which this can be achieved is to realize that

$$= \underset{X}{E}\Big(F(\mu(X))\Big) = \frac{1}{MN}\sum_{m}\sum_{n} F\Big(\mu(X_{mn})\Big) = \frac{1}{MN}\sum_{l=0}^{l_{max}} \sum_{\substack{m,n \\ x_{mn}=l}} F\Big(\mu(l)\Big) \qquad (3.4)$$

where $\sum_{\substack{m,n \\ x_{mn}=l}}$ indicates the summation over all pixels within the N by M pixel image that have intensity l and hence it is clear that Equation 3.4 can be written as

$$I(X) = \frac{1}{MN}\sum_{l=0}^{l_{max}} HIST(l)\cdot F(\mu(l))$$

where $HIST(l)$ is the number of pixels in the histogram of the image which have intensity l.

But, from above $\mu(l)$ *is* equal to the S function

$$\mu(l) = S(l, l_0, \delta l)$$

which can be written as

$$\mu(l) = S(l, l_0, \delta l) = S_{\delta l}(l - l_0) ;$$

since for a given δl, S and hence $\mu(l)$ is a function of the difference $l - l_0$ between the intensity and the threshold l_0 considered. For a given δl, the combined operation of weighting with accordance to the S function and then using this within the F function can be considered as an overall "windowing" denoted as $W_{F,\delta l}(l - l_0)$ which is the convolution $W_{F,\delta l} = F * S_{\delta l}$ and hence, finally, we have

$$I(X) = \frac{1}{MN}\sum_{l=0}^{l_{max}} HIST(l)\cdot W_{F,\delta l}\Big(l - l_0\Big) \qquad (3.5)$$

In those cases in which the window is not symmetrical

$$I(X) = \frac{1}{MN} \sum_{l=0}^{l_{max}} \text{HIST}(l) \cdot \hat{W}_{F,\delta l}(l - l_0) \quad \text{where } \hat{W}(l) = W(-l)$$

The overall windowing functions corresponding to the fuzzy indices put foward in Ref [3.1] and defined in Eq. 1 2 3 Ref [3.1] above are shown plotted in Fig 3.2. The windows are drawn for a normalized bandwidth $\delta l=1$ and the effect of changing bandwidth is simply to horizontally scale the window. The significance of the Equation (3.5) is (i) by using the histogram the number of computations is reduced since all that is necessary is for the number of pixels having a given particular intensity (between 0 and l_{max}) to be multiplied by the windowing function and for this to be done for each of the levels $0,1,\ldots,l_{max}$. (ii) standard computational packages for evaluating convolutions are available and hence simply by obtaining the windowing function and convolving this (using existing programs) with the histogram the function I(X) can be obtained (iii) the approach developed shows that the technique of thresholding developed by Pal et al can be considered within the wider context of windowing and this opens up the possibility of studying thresholding (and its relation, the subjective acceptability of the images after thresholding) in a new light.

3.3.1.1 An Annealing Technique for the Automatic location of thresholds

In Ref [3.1] a fuzziness index was obtained as a function of threshold point l_0 and the positioning of thresholds considered to be located at, or close to, the positions of a broad deep minimum.

It would be of value to use the fuzziness index technique and to combine it with some purely automatic method for finding the position of the thresholds (the number of which may be a-priori unknown). A technique for doing this, which is based on similar ideas to those contained in the annealing schemes that have been put forward (Ref [3.2]) for the refining of hypothesis testing, is as follows: (1) Compute the I(X) function (using some given window function) for

an initially high number of threshold positions. (2) Convert the distribution I(X) (distance with respect to the position) to a function I(X)=k−I(X) with k being a constant which is chosen so that $k-I(X)=0$ for the largest value of I(X) that is possible. The reason for doing this is that the function I(X) in which minima are associated with the desired positions of the thresholds is converted to a function in which maxima are associated with the most probable positions of the thresholds. (3) The set of distinct I(X)'s are then split and divided, with neighbouring points being combined so as to obtain a number of discrete I(X)'s that are of equal magnitude, but which are positioned at the center of gravity of the merge points. (4) Compute I(X) for these points and go to (2)

This is illustrated in Fig 3.3. It will be noted from Fig 3.3 that there is a tendency towards congregation of points close to the maxima of I(X) ie close to the minima of the original discrete I(X) function. The steps 2 and 3 are repeated until sufficiently close packing of the thresholds is achieved in respect of a particular broad minimum in the original I(X) function.

Clearly the extent of the convergence depends on the number of iterations involved and the bandwidth of the windowing function. The shape of the window function is also important in that by appropriate choice it is possible to obtain enhanced detection of minima above and beyond that which can be obtained simply by using a reduced bandwidth.

Application of the annealing technique is shown in Figs 3.3, 3.4, 3.5 and 3.6. The histogram data used in Fig 3.3 was taken from Ref [3.1] and that of Fig 3.4 was obtained by digitizing a newspaper image. The later was digitized to 8 bits resolution. Both Figs 3.3 and 3.4 deal with the situation in which the histogram of intensities contains a single local minimum. In applying the annealing technique to the data contained in Fig 3.3 a window was used which consists of three identical entropy-based windows similar to that shown in Fig 3.2. The three-unit window was used to discriminate between local minima and broad flat regions of the histogram. The window structure is shown in Fig 3.3 with the associated intensities $I_0(X)$, $I_+(X)$, $I_-(X)$ being the product of the corresponding window by the histogram.

The annealing intensity function, I(X), used was $I(X) = I_0(X) - \frac{I_+(X) + I_-(X)}{2}$. Each component of the window used in connection with the Fig 3.3 had a bandwidth that was arbitrarily chosen to be 5. The window function used in connection with Fig 3.4 was the same as that used in connection with Fig 3.3, except that the window width of each unit was initially set at 48 and then reduced to 32,16,8 and 4 after each successive 10 iterative steps. From Figs 3.3 and 3.4 it is clear that there is a convergence towards a local minimum. In Fig 3.5 the newspaper image is shown after it has been quantized to two levels when using (i) the threshold arrived at and shown in Fig 3.4 and (ii) the threshold in the middle of the range of intensities (0 to 255). The original 8-bit image is also shown in the figure. Inspection of the Fig 3.5 shows that subjectively more acceptable results are arrived at when using the threshold obtained from annealing.

The technique described above was also applied to an image having a multi-modal intensity histogram and it was found not to work well, since certain local minima were not detected. In order to overcome this problem a slightly modified intensity function, I(X), was used in conjunction with the previously described window function. The intensity function

$I(X) = I_0(X) - \left(I_+(X) \cdot I_-(X)\right)^{1/2}$ was used.

The technique was applied to the histogram material presented in Ref[3.1], with a bandwidth of 11 being adopted for each unit of the window. The results of the application are shown in Fig 3.6 from which it is clear that the technique appears to work well. Of the initial 39 seed points 9 converged to the obvious lowest local minimum, 9 converged to the middle local minimum and 19 converged on to the highest obvious local minimum. One seed point converged to the low end of the histogram and one to the high end. These latter two convergences can be considered as very weak and can be discarded. It may be that had the number of iterations been increased, these points would also have converged to the nearest obvious local minima. A total of only 20 iterations were involved in arriving at the results shown in Fig 3.6.

It is clear that the tripple window with geometric mean performs better for multi-mode images.

3.3.2 Hard Skeletonization

An essential step towards the recognition of line-based patterns is that of line thinning, or skeletonization, as it is now commonly called. Clearly, from the point of view of recognition, the width of the line involved is of little significance.

The process of skeletonization consists essentially of successive deletions of black points along the edges of the pattern, with the process continuing up to the point where further continuation would result in changes in connectivity. C.J.Hilditch (Ref [3.3]) proposed the following criteria for a candidate black point to be removed:

(a) the point must be an edge-point

(b) the point must not be an end-point

(c) the point must not be a break-point i.e. one whose removal will affect connectivity

and (d) the point must not be such that its deletion will result in excessive erosion, either of the whole pattern or some significant part of it.

Hilditch did not propose any specific algorithm for the removal of points in accordance with a set of rules, but it is clear that little basic difficulty exists with respect to operating in accordance with rules a,b and c. However, in the case of rule d, significant difficulties exist and the usual approach is not to delete a point at once but to "flag" it for deletion and then to examine the effects of its deletion on subsequent iterations of the skeletonization of the image. Stefanelli and Rosenfeld (Ref [3.4]) proposed an algorithm for the skeletonization operation according to the rules of Hilditch. However Naccashe and Shinghal (Ref [3.5]) showed that the Stefanelli-Rosenfeld algorithm can result in the erosion of whole strokes and they proposed a new algorithm based on a modified criterion for a break point and on testing in advance whether a candidate point will become a break point after the deletion of an already flagged neighbour.

It can be shown, however, that even this modified algorithm of Naccashe-Shinghal introduces certain problems of its own insofar as subsequent processing of the image is concerned. For example, if the letter A (taken from Ref [3.4]) is processed according to the algorithm proposed by Naccashe and Shinghal then the result shown in the Fig 3.7a is obtained. It is clear from the figure that significant ripples exist in the resulting skeleton and that serifs, that generally are unimportant in the recognition process, are retained, and, in addition, extra "tails" are introduced (loops may also be introduced). The ripples and extra features can lead to a significant increase in the "length" of the description to be used in the next stage of the pattern recognition process.

Another important aspect relating to the algorithm is its sensitivity to the so called "starting point" i.e. the point (or corner) of the image at which the skeletonization process begins. Variations in the skeleton resulting from such changes are important in that they indicate how sensitive the algorithm is to rotational effects and this in turn has an influence on the variational nature of the features subsequently extracted and used in recognition.

A test was carried out using the image given in Fig 3.7 and the resulting skeletons obtained when starting at the four different points (corners) indicated in Fig 3.7b are given in the figure. The changes in the starting position are equivalent in this case to rotating the image by 90° 180° and 270°. It will be noted from the figure that significant variations in the skeleton exist, and in particular different "tails" are introduced.

An attempt has been made during the current work to improve the performance of the algorithm by changing the direction of application of the algorithm by 90° after a complete iteration; that is, a complete set of points has been removed. The results obtained are shown in Fig 3.7c. It is clear from the results contained in the figure, however, that little or no advantage is obtained. A possible reason for this is that in the case being considered a large majority of the total number of points removed are taken away during the first two iterations and thereafter the results are biased towards those obtained when beginning from the two associated directions. Some evidence in support of this

conjecture can be seen from the similarity of Figure 3.7c to the skeletons obtained when staring from the top left (the first iteration) and from the top right (the second iteration).

A recent approach to skeletonization is the so called Morphological Skeleton representation (Ref [3.6]). This technique is based on the use of a structural element (possibly a circle or a square) centered on a sequence of points that are considered to constitute the skeleton. The technique differs from the ideas discussed previously in that the image can be fully recovered from the skeleton. Illustrations showing the application of the technique are given in Fig 3.8. The figure shows the resulting minimum skeleton obtained when certain strokes giving rise to corners or juxtapositions and T-junctions are involved. The strokes involved are considered to be of constant width throughout the length of the stroke but the width can vary from one stroke to another. The results contained in parts a to e of Fig 3.8 clearly indicate that significant difficulties exist with the technique and that spurious tails etc are contained in the resulting skeletons. An illustration of the way in which the image can be reconstructed from the morphological skeleton is given in Fig 3.8a.

From the above discussion it is clear that the question of efficient skeletonization, even today, presents problems. In Section 3.4 of this chapter the matter is taken up again when a new approach based on soft skeletonization is presented.

3.3.3 The Further Encoding of a Hard Skeleton

After having obtained an appropriate hard skeleton, that is, the obtaining of a one pixel-width, the next stage of processing, either for the purposes of efficient storage, or as a further step towards recognition, is that of the efficient encoding of the information contained in the skeleton. A technique of encoding that has become popular (Ref [3.6]) is that of Chain Coding in which a three-bit word is associated with a pixel and the three bits are used to indicate the position of the next connected black pixel. The way of using the three bits is

indicated in Fig 3.9. An indication of the efficiency of the technique can be gained by considering the black/white image of the letter "A" showing in Fig 3.7a. A count of the relative frequency of occurrence of black and white shows that the probability of the pixels being black is 0.151 and hence the entropy is .612 bits/pixel. This means that by entropy coding it is possible to encode the image using 2508 bits. On the other hand, by using the 3-bit coding approach it is possible, for the 5 stroke image, to encode it using $3 \cdot 104 + 5 \cdot 12 = 372$ bits.

The encoding technique just described takes no account of the fact that character images are generally made up of straight-line sections, and curves, which, relative to the pixel resolution, can be regarded as approximating to piecewise-linear sections with approximately constant angles between the various sections which constitute the approximation to the curve. Clearly, advantage can be taken of this and the idea of run-length encoding, which has been used extensively as a method of reducing redundancy in areas such as facsimile encoding etc, can be applied.

Suppose a section of an image consists of number of pixels for which the chain coding representation is identical, ie the direction to the next black pixel is the same from one pixel to the next. A sequence of k such pixels can be considered to constitute a run of length k. If all runs up to some maximum length M are allowed and higher runs are considered as being broken down into subruns whose lengths are in the range 1 to M (clearly it is most beneficial to break a run down into a sequence of longest possible subruns) and p_k is the probability that a pixel belongs to a run of length k; $1 \leq k \leq M$, then it can be shown that there is advantage to be gained through the use of run-length encoding if (see next page for proof):

$$\left(3 - \sum_{k=1}^{M} \left(\log_2(M) + 3\right) \cdot \frac{p_k}{k}\right) > 0 \text{ , that is, if}$$

$$\sum_{k=1}^{M} \left(3\left(1 - \frac{1}{k}\right) - \frac{1}{k}\log_2(M)\right) p_k > 0$$

This expression can conveniently be written as $\sum_{k=1}^{M} A_k p_k > 0$

with $A_k = 3\left(1-\frac{1}{k}\right) - \frac{1}{k}\log_2(M)$

The values of the coefficients A_k are given in Fig 3.10 for various values of M. The figure shows which coefficients are positive and which are negative. Positive values of A_k favour run-length encoding.

It is worth noting that, if instead of using the absolute angle (3 bits) to specify the direction in going from one pixel to the next the difference in angle is used, then a further significant data reduction can be obtained from the use of run length encoding, though the encoding then becomes more sensitive to noise.

Proof: The encoding of a run of length k requires $\log_2(M)$ bits to express the value of its length plus 3 bits to exprees its direction. These bits are shared by the k pixels of the run. The length per pixel is averaged over all the pixels and compared with the 3 bits/pel required by the simple chain enconding.

3.4 A NEW LOSSY APPROACH TO THE ENCODING OF LINE IMAGES

3.4.1 Introduction

It should be noted that the coding techniques described in the previous subsection, both in respect of per-pixel encoding and the run-length encoding of angles and differences between angles, are information lossless (distortionless) and slight variations such that might be associated with any ripple introduced during the skeletonization process will be encoded accurately. Clearly, these variations are generally of no significance and more efficient encoding can be achieved if they are discarded. In this subsection a new approach to encoding skeletonized images is put forward. The technique is lossy in that some distortion is introduced, but if advantage is taken of preferential line directions, (and the fact that certain small variations such as ripples are of little significance), then the encoding can be made very efficient in terms of the number of bits actually used. The question to be addressed can be expressed in the following general manner: What is the set of points, relative to the current point, that represents a line sufficiently accurately but with reduced number of bits being employed in the encoding?

The arguments can be explained by considering Figs 3.11 and 3.12. It will be remembered that with normal chain coding a 3-bit (usually) word is used to indicate the direction of the next pixel which is actually adjacent to it in that it touches the current pixel. This can be considered as locating the next pixel which is at a distance 1 from the current pixel. Now since lines show preferential directions, that is, the direction to the next pixel tends to be within the sector $\pm 45^{\circ}$ relative to the current pixel, it should be possible to use the number of bits more effectively and to jump from one pixel to the next with the distance involved being >1; while hopefully the distortion is kept within acceptable limits. This is illustrated in Fig 3.11a. The technique can be further generalized as illustrated in 3.11b with non-constant distances being involved. This is particularly suitable for encoding lines that are effectively straight. A final generalization envisaged is that shown in Fig 3.12. In the figure an area enclosing the most likely next positions is shown. Next points located closest to the perimeter of the area indicate large steps, whereas those closer to the present point indicate smaller steps (strides).

The question of how to implement such an encoding technique will now be considered. In particular, the question will be addressed as to how encoding using contours similar to those shown in Fig 3.12 can be implemented.

The lossy skeletonization technique referred to above can be applied to two-tone (black/white) images, to an original grey-scale representation of an image, or to a regenerated pseudo-grey-scale version of the image which is regenerated from the two-tone image, or from the grey-scale image itself. In the description which follows a technique for regenerating a pseudo-grey-scale image (called a Soft Skeleton) from a binary black/white image will be described and then the encoding technique (the so called Stride Encoding technique) of skeletonization will be described. The reasons for generating the soft skeleton were (i) the images available for processing were in the form of black/white images (ii) the proposed techniques for the generation of the soft skeleton can form a smoothing function which tends to fill in gaps etc in either the two-tone or in the original grey-scale image.

3.4.2 Soft skeletonization using a Center of Gravity criterion.

(i) Basic ideas

Assume that an image is available, either in two-tone or in grey-scale form and that the image is M pixels wide by N pixels high and that it is desired to produce a soft skeleton, the so called the pseudo-grey-scale image referred to above. One criterion for generating such a skeleton is based on the idea of center of gravity. A soft skeleton based on this criterion can be defined as; the image which is such that every pixel within it has a value (a significance/intensity) that is dependent on the distance of the pixel from the center of gravity of the set of pixels located within some specified area centred on the pixel under consideration. The closer the pixel to the center of gravity the more significant it is considered to be. The generation process leading to a soft skeleton can be described formally as follows: Consider the image as indicated diagrammatically in Fig. 3.13 and assume that a window area is placed on the pixel under consideration as shown and that the window height has the general profile shown in Figs 3.13 and 3.14. Then the significance, or pseudo intensity, $\mathfrak{I}$, associated with the pixel under consideration is taken to be

$$\mathfrak{I}=\frac{\left(\left(\sum_i\sum_j \mathrm{i}\,M_{ij}w_{ij}\right)^2+\left(\sum_i\sum_j \mathrm{j}\,M_{ij}w_{ij}\right)^2\right)^{1/2}}{\left(\sum_i\sum_j M_{ij}w_{ij}\right)\left(\sum_i\sum_j M_{ij}w_{ij}\right)^n} \tag{3.5}$$

where $-k_{\min}\le \mathrm{i}\le k_{\max};\ -l_{\min}\le \mathrm{j}\le l_{\max},$ with:

k,l being the extremities of the window relative to the center of the window,

M_{ij} is the intensity of the pixel with co-ordinate i,j within the window;

w_{ij}, is the window weight of the element located at co-ordinates i,j within the window and n is some positive "normalization" exponent.

The pseudo image, or soft skeleton, is built up by considering all elements, pixels, of an image bounded as shown in Fig 3.13. Detailed examination of Eq 3.5 shows that if the term $\left(\sum_i\sum_j M_{ij}w_{ij}\right)^n$ is removed then the remainder

corresponds essentially to a normal center of gravity and hence this function has a small value when the window covers an area of substantially uniform intensity and it is not possible to distinguish between such areas, even if the intensity is widely different from one area to the next. One way of overcoming this is to include the term $\left(\sum_i \sum_j M_{ij} w_{ij}\right)^n$ in the equation, with the result that $\mathfrak{I}$ becomes large for white areas and small for black areas. Another technique is to examine the intensity $\sum_i \sum_j M_{ij} w_{ij}$ which is found to be substantially uniform and to set $\mathfrak{I}$ equal to the expected (or ultimately determined) $\mathfrak{I}$max if the value of the $\sum_i \sum_j M_{ij} w_{ij}$ term is below some predetermined threshold value. A further way which is actually used in the first experiment reported later in this chapter is simply to examine the actual intensity of the pixel under consideration and if it is found to have an intensity below some threshold (corresponding to white) to set its pseudo intensity equal to $\mathfrak{I}$max.

Since it is clearly desirable that the centre of strokes, which is most closely associated with the familiar hard skeleton, should be emphasized, it is necessary to use a transformation of the pseudo-intensity given by Eq 3.5. One such simple transformation is to use as pseudo intensity, $\mathfrak{I}_{ij}'$, where

$$\mathfrak{I}_{ij}' = 1 - \frac{\mathfrak{I}_{ij}}{\mathfrak{I}\text{max}}$$

In Eq 3.5 above a general window element weight w_{ij} is assumed, and a general window area (shape) is also assumed. In the theoretical arguments to be presented later in this chapter regarding the soft skeleton of certain important commonly encoded strokes, it is assumed that the window height is $w_{ij} = 1$ for all i,j and that the window shape is a circle. In the experimental work reported on in this chapter the window weight is again taken to be $w_{ij}=1$ for all i,j, but on account of the discrete nature of the image domain a window of shape and actual pixel dimension as shown in Fig 3.14 was used. This shape is the best approximation to a circle for the given resolution and radius.

(ii) Analysis of the Center of Gravity based skeletonization of constant width strokes.

In this subsection consideration will be given to an analysis of the center of gravity based soft skeletonization of certain constant width strokes. In the analysis, black/white images will be used as source material to be subjected to soft skeletonization. Also, in the analysis a circular masking window will be assumed. The purpose behind the analysis is to provide some indication of the significance of the radius of the masking window relative to the width of the image strokes involved. In addition the results obtained provide some information as to how useful the center of gravity technique might be as a means of obtaining a skeleton from a image made up from constant width strokes. In the analysis, for convenience, the radius of the masking circle will be set to a value of unity and the width of the strokes will be taken to be some value W. Clearly, the important parameter is the ratio of the circle diameter to the stroke width. In the analysis three basic stroke shapes will be considered; these will be the straight line, the constant radius of curvature arc, and the interaction of two constant and equal width lines (see Fig 3.15). The direction of motion of the masking circle is shown in Fig 3.15 for each of the strokes. It is thus clear that, with respect to the axis perpendicular to the direction of motion, the image (the stroke) is symmetrical and hence the Eq 3.5 simplifies to the following:

$$\mathfrak{I} = \frac{\left\| \sum_i \sum_j i M_{ij} \right\|}{\sum_i \sum_j M_{ij}} \qquad (3.6)$$

where it is assumed that the summation $\sum_i \sum_j$ corresponds to a summation perpendicular to the axis of the movement of the masking circle. It is clear that Eq 3.6 is, as one would expect, the familiar one-dimensional center of gravity of the image.

Case 1: Straight Lines.

Consider the center of unit radius circle to be located at some arbitrarily chosen axis. Let X_L be the normilized distance of the left hand edge of the stroke from this origin,

normalized
and X_R be the distance of the right hand edge from the origin. Distances are taken as being positive or negative depending on whether the edges are to the right or left respectively of the origin. Let x be the distance measured from the middle of the stroke to the origin, with this distance being negative when the center is to the right of the origin. The situation of the center of the stroke being to the left of the origin need not be considered on account of the symmetry of the situation. Thus, it can be shown that if the distance X is expressed as $x=X\cdot W$, ie is measured in units of stroke width, Eq (3.6) becomes

$$CoG = 1 - \frac{2}{3}\cdot\frac{\left\|\left(\sqrt{1-X_R^2}\right)^3 - \left(\sqrt{1-X_L^2}\right)^3\right\|}{\left(\cos^{-1}(X_R) + X_R\sqrt{1-X_R^2}\right) - \left(\cos^{-1}(X_L) + X_L\sqrt{1-X_L^2}\right)} \quad (3.7)$$

where $X_L = \text{sign}(X_l)\cdot\min\left(\|X_l\|,1\right)$; $X_R = \text{sign}(X_r)\cdot\min\left(\|X_r\|,1\right)$

and where the min(a,b) is taken to mean that the minimum of a and b is used. This expression is perfectly general in that it places no restriction on the ratio of radius to stroke width. Some of the results obtained from Eq (3.7) are shown in Fig 3.16 where the intensity is plotted as a function of the ratio of the distance of the center of the circle to the width of the stroke. The figure shows the results obtained when several values of the diameter-to-width ratio are used. From the figure it is clear that, if the technique is to be used to obtain a clearly defined hard skeleton, then a sufficiently large diameter-to-width ratio has to be employed (approximately in the range 1.5 to 2.0). Another interesting feature displayed in the Fig 3.16 is that for various values of $D/W \leq 2$ the intensity value has the same value (a value of 0.576) at the edge of the stroke. This provides a reliable and relatively simple means of determining the stroke edges and, since a smoothing effect is involved the application of the technique results in a "cleaning up" of the edges. An actual illustration of this is shown in Fig 3.17.

Case 2. Curves of constant mean radius.

The situation being considered is shown in Fig. 3.15c. Although it is not difficult to find the effective pseudo . _ for a given position of the center of the circle relative to the stroke, it has not been found possible, when the diameter of the circle is greater than the stroke width, to obtain an analytical expression for the position giving the maximum value of the intensity which corresponds to the situation in which the center of the circle is located at the center of gravity of the area measured relative to the center of the circle. It has in fact been found possible, however, to obtain upper and lower bounds to the position of the maximum.

It can be shown that the distance, O_E, of the position of the maximum intensity measured from the middle of the stroke is bounded as follows

$$\sqrt{R_c^2 - \tfrac{1}{4}\left((2R)^2 - 1\right)} - R_c < O_E < \sqrt{R_c^2 + \tfrac{1}{4}\left((2R)^2 - 1\right)} - R_c$$

where R = radius of the circle, $R_c = \frac{R_2 + R_1}{2}$, and all distances are normalized so the stroke width $(R_2 - R_1)$ is unity. However, from further geometric arguments it can be seen that the center of gravity cannot lie to the right of the middle of the stroke, and hence the following tighter bounds are obtained.

$$\sqrt{R_c^2 - \tfrac{1}{4}\left((2R)^2 - 1\right)} - R_c < O_E < 0$$

Results illustrating the values of these bounds are given in Fig 3.18. From these results it is clear that, if the ratio of the diameter of the masking circle to the width of the stroke is slightly greater than 1, then the position of the largest value of the center of gravity is very close to the middle of the stroke. It is also clear from the figure that the bounds converge and approach the middle point if the ratio of the mean radius $\left((R_2 + R_1)/2\right)$ to stroke width is large. The bounds are also seen to be quite tight when the diameter of the masking circle is close to and slightly greater than the stroke width.

Case 3. Angles formed by the intersection of two straight lines of equal width.

The situation analysed is shown in Fig 3.15b. It is not a simple matter to evaluate the pseudo intensity but the problem is greatly eased if the situation is examined by considering the effective moments of the area encompassed by the masking circle as associated by each of the two lines X_{OUTER} and X_{INNER} as shown in Fig 3.15b. The effective moment (and the corresponding area) is obtained by appropriate subtraction of moments and areas.

Consider one of the two lines, say the outer line, and denote the intersection of the line with the line of motion of the axis of the masking circle by X_e. The areas involved in evaluating the center of gravity are A1 A2 A3. It should be noted that when X_e is within the circle $A3 = 0$. Then M1 M2 and M3 denote the associated first moments. It can be shown that when X_e is inside the masking circle the areas and the associated moments are:

$$A1 = \frac{1}{2}\left(R^2(\pi - \theta_n) + X_n \cdot a_n\right) \qquad M1 = -R^3 a_n^3$$

$$A2 = \frac{1}{2}\left((X_e - X_n)\, a_n\right) \qquad M2 = \frac{1}{2}\left((X_e - X_n)\, a_n\right)\left(\frac{2X_n + X_e}{3}\right)$$

$$A3 = 0 \qquad M3 = 0$$

and when X_e is outside the masking circle the corresponding areas and moments are:

$$A1 = \frac{1}{2}\left(R^2(\pi - \theta_n) + X_n \cdot a_n\right) \qquad M1 = -R^3 a_n^3$$

$$A2 = \frac{1}{2}\left((X_p - X_n)(a_p + a_n)\right) \qquad M2 = \frac{1}{2}\left((X_p - X_n)(a_p + a_n)\right)\cdot\ldots$$

$$\ldots\cdot\left(\frac{(a_p + 2a_n)X_n + (2a_p + a_n)}{3(a_p + a_n)}\right)$$

$$A3 = \frac{1}{2}\left(R^2\theta_p - X_p \cdot a_p\right) \qquad M3 = R^3 a_p^3$$

where $\theta_n, \theta_p, a_p, a_n, X_p$ and X_n are as indicated in Fig 3.19. Some of the results obtained are given in Fig 3.20. In the previous cases, the curves shown in the figures demonstrated the existence of a common intensity level at the

"edges" (see in the Fig 3.15b for an indication of what is considered in this case to be an edge) for a range of values of masking circle. It should be noted from the results presented in Fig 3.20 that in the case of angles formed by the intersection of lines the maximum does not occur at a point of equal distance from the edges but closer to the inner edge. This means in effect that the center of gravity technique tends to produce a soft skeleton (and the same would be true for a hard skeleton derived therefrom) with "rounded" (less acute) corners.

3.4.2.1 Examples of Soft Skeletons produced from two tone images

The center of gravity technique described in the Subsection 3.4.2 has been applied in order to demonstrate its effectiveness in producing the soft skeletons from two-dimensional images. The two dimensional images considered were quantized to a resolution of 64x64 pixels. The images used are shown in Figs 3.21a, b and c. Fig 3.21a shows a well formed letter A, Fig 3.21b shows a more poorly formed A and Fig 3.21c an image which corresponds to a letter S formed by a set of bars (or lines). A masking window as shown in Fig 3.14 was used. This corresponds to a circle of radius 6 pixels and pixels whose coordinates i,j relative to the center of the circle were considered to be within the circle if $\sqrt{i^2+j^2}$ when rounded to the nearest integer was ≤ 6. The results of the process are shown in Fig 3.22a, b and c. From the figures it is clear that the intensity is greatest near the centers of the strokes involved and that there is a significant tendency towards the reduction of serifs. It will also be noted that the characteristic referred to in the immediately preceding subsection regarding the rounding of the acute angles and the movement of the maximum intensity toward the lower edge of an angle can also be observed in Figs 3.22 a, b and c.

Although the center of gravity technique of soft skeletonization described above may be of importance in its own right, its main significance is in its use in conjunction with encoding techniques to obtain a low bit representation of an image. A new encoding technique to be used in conjunction with the soft skeletonization technique is described in the following subsection.

3.4.3 A Stride Encoding technique

The conventional chain coding process described in an earlier subsection operates on a hard single-pixel wide skeleton to produce an efficient representation in the form of a set of binary digits which represent small segments of the skeleton.

A new alternative approach to encoding is put forward and examined in this Subsection. The technique is based on the use of a soft skeleton of the type described in the previous subsection and the technique differs fundamentally from a conventional encoding technique in that a choice exists as to the next pixel which is considered to form an element of the skeleton. The choice of this pixel, and the number of pixels used, clearly determines the effectiveness with which a meaningful skeleton can be reconstructed from the pixels and it also has a bearing on the accuracy of the recognition process. The technique to be described in this subsection is referred to as a "Stride Encoding" technique since the resulting coding representation of the stroke is not connected in the sense of being a set of connected pixels. The stride encoder generates a sequence of "next points", together with information about bifurcations and end points, and this information, taken together, forms the encoded representation of an image. The essential details of the stride encoder process will now be described.

(i) Selection of the first point.

The choice of starting point in any encoding process relating to images is fairly arbitrary, apart from the fact that some extremity, usually an end point, is used. There is no particular reason to start at, say, the top -left, the top-right or bottom corners as far as most images are concerned. In the work reported in this section, a bottom-left corner was taken as the position from which the scanning operation was started. In effect a 45° line was moved up and hence across the image and the first pixel encounted whose intensity was in excess of some predetermined threshold was taken to be the starting point, or the first pixel.

(ii) The generation of a sequence of next points which represents an image

The generation of a sequence of next pixels involves the generation of next points, one after the other, with each such point being relative to the preceding point. The stride encoder decides on a next point by considering a number of possible contender points. A decision is then made in favour of one of the points in the contender set. The process then steps on the so selected point and the process is repeated. The elements involved in the stride encoder operation are similar in concept to those described earlier when considering extensions of the conventional Chain Encoder scheme. An area of consideration containing a number of contended points, which are positioned relative to the origin within the area of consideration, is placed on the image to be encoded, with the origin being located at the last selected point. The operation starts with the origin being placed at the "first point" discussed immediately above. The direction of orientation, the axis or the so called "preferred" direction, of the area under consideration is determined by the immediate past history of the image (broadly speaking the expected direction of the stroke based on previous observations). For example, one way of positioning the axis is to place it along the line joining the immediately two previous points. Certain questions have to be answered in order to design a stride encoder. These are

(i) what is the size of the area of consideration that should be used?
(ii) how many contender points should be enclosed within the area?
(iii) where should the points be placed?
(iv) what weights should be associated with the various points?
and (v) how should the next point should be selected?

The questions are very much interrelated and they are extremely difficult to answer in analytical terms. However, certain guidelines can be set out as to the choice of various parameters. The most important guidelines are

(i) the size must not be such as to bridge meaningful gaps, whilst at the same time it is desirable that small meaningless gaps be bridged. Also the size must be such as to avoid the creation of false bifurcations.

(ii) As regards the positioning of the contention points, this is a matter that is clearly related to the size of the area of consideration. The points need to be sufficiently far apart to result in an adequate stride (which yields efficient encoding) and the points must be positioned predominantly in the forward direction. By examining the extent and frequency with which short segments of the image change (the change in direction between selected successive points) further insight in the positioning of the points can be obtained,

(iii) the number of points within the area is clearly related to the resolution involved, and the efficiency of encoding aimed at,

(iv) the weight associated with the points is related to their positions and the efficiency of the encoding and it is clear that advantage may be gained by assigning higher weights to points that are positioned close to preferential directions, and that the efficiency of encoding can be increased by weighting more heavily points that are at a greater distance from the origin.

From the description above it is clear that flexibility exists in the selection of the next point, unlike conventional chain coding. In the stride encoding scheme any one of the contender points within the area of consideration is a possible choice for the next point and a variety of strategies for selecting the point are possible. It is difficult to derive any optimum strategy since many factors are involved. These include the subsequent ease of recognition of an image from a stride coded representation, and the associated efficiency of the encoded operation (the degree of redundancy removal involved). In order to gain some insight into the question of selecting the next point, the following strategy was adopted for use in the experiments reported on later in this chapter.

The values of the weighted intensities of each contender point within the area of consideration are evaluated and the most likely next point is taken

initially to be the point of highest weighted intensity. The point is accepted as a next point if (i) its unweighted intensity is greater than some predetermined threshold, (ii) it is not close to some previous accepted next point which forms part of the representation of the current stroke (see Fig 3.23a) unless the length of the loop resulting exceeds a predetermined length, (iii) it is not close to a next point on a stroke joined to the current stroke by bifurcation, unless the length of the substrokes constituting the loop exceeds a predetermined value (see Fig 3.1b), and (iv) if a next point is acceptable according to rules i to iii above, then subsequent next points such as those shown in Fig 3.23c, which follow a junction, are acceptable if the length of the stroke (or loop) as measured from the junction is in excess of some predetermined threshold.

All contention points having a value over a threshold greater than that used in (i) are kept in memory as possible next points, or as points of bifurcation.

If the point of highest weighted intensity does not have an unweighted intensity greater than the threshold, then the contention point of next highest weighted value is examined in accordance with the criteria outlined.

A stroke is considered to be ended (an end point is considered to have been reached) if no possible next point satisfies the criteria defined. Having obtained the stride coded representation of a stroke the stored possible bifurcation points are then examined. A first test is carried out to determine whether a point is actually a bifurcation point. This is done by comparing the unweighted intensity of the point with a threshold somewhat higher than that used in (i) to determine the next possible point of the stroke. The reason for using a higher threshold can be explained by considering Fig 3.24. It is desired to select as next point a point A, which will have less intensity than points such as points marked X, when moving along the stroke as indicated, but to discard point B, as a bifurcation point. If a point has an intensity greater than the threshold, then a further test is carried out to determine whether there exists some previously accepted point (used in a coded representation of a stroke) which is closer to the point being considered as a bifurcation point. If such a so positioned point exists, then the point under consideration is discarded. In the

event of no such point existing, the point under consideration as an bifurcation point is accepted and the stride coding of a new stroke is commenced, with the direction being taken as that of the line joining the bifurcation point and the point giving rise to it.

3.4.3.1 The Stride encoding of the soft skeletons considered earlier.

In order to illustrate the effectiveness of the stride coding technique when used in conjunction with the soft skeletonization technique, the soft skeletons shown in Fig 3.22a, b and c were stride encoded. In the case of the soft skeleton of the well formed letter A (see Fig 3.21a) an area of consideration identical to that shown in Fig 3.14 (the masking element) was used, with the points of contention being the 32 points which form the perimeter of the area. In the experiment a uniform weight of value 1 was allocated to each contention point. The resulting set of next points which were extracted by the stride encoder are shown in Fig 3.25. It is clear that the resulting representation provides the basis for the accurate recognition of the letter by the human being and that, provided that the recognition system is designed correctly, it should also form the basis for the accurate machine recognition of the image. The stride coder actually produced the results in the form of two strokes, the first being indicated by the integer 1 and the second indicated by integer 2. It is clear from the results that the encoder functioned satisfactorily and it also tends to reduce serifs somewhat. The reason why the encoder did not follow the straight line associated with the left hand element of the A is that with the uniform weight allocated to the contention points, no preference is given to straight line direction for next point selection.

In the case of the more poorly formed A shown in Fig 3.21b the stride coding was carried out using the area of consideration and the set of contention points indicated in Fig 3.26. The weights associated with the various contention points are also shown in this figure. It is clear that a preference is built in the weighting scheme so as to favour movements along the straight line.

In the experiment, the area under consideration and the associated contention points were maintained in the absolute position shown in figure 3.26. This was done so as to ensure that each point lies on (over) a pixel of the image. If the previous point was as indicated by P1 then the weights are shown in the figure. If, however, the previous point was some point such as that indicated by P2, then the weight were rotated so as to take up the new preferred direction as shown in Fig 3.26b. The results of applying the stride coding technique to Fig 3.21b are shown in Fig 3.27. The various thresholds referred to above and adopted in the experiment are also shown in the figure. Although it is not immediately clear from the results contained in the figure, the encoder actually bridged the unnatural break where the "horizontal stroke" of the A was supposed to join the left hand leg of the A. That the encoder did actually bridge this gap can be determined from the fact that the "point X" indicated on the figure was actually generated from the bifurcation of the left end of the A. If, however, the bifurcation had started from the right leg then a point somewhat as shown and marked with "*" would have been generated. An experiment was carried out to verify this and the result is shown in Fig 3.27. It should also be noted from the figure that the unwanted serif close to the middle of the right leg is significantly suppressed by the encoding operation. The resulting line drawing obtained by linking the set of next points is again seen to constitute a good representation of the letter A. It was found that, when applied to the well formed A, the latter technique worked equally well.

In the case of the third image the same area of consideration and the same set of contention points was used as in the second example. The various thresholds used are indicated in Fig 3.28. The results obtained from the stride coding operation are shown in Fig 3.28. In this case, although the original image was made up of a set of discontinuous lines, a sensible and acceptable stride code is obtained which could be used in a recognition system.

In Figs 3.29a and b the normal chain coded representation of the images shown in Fig 3.21a and b are shown so that comparison can be made with the skeleton obtained through the use of soft skeletonization and stride coding.

3.5 SOME CONCLUSIONS

In this chapter general consideration has been given to the preprocessing of images prior to recognition. In particular the question of thresholding, (the production of a two-tone image from a grey-scale image), has been considered. An important recent work on the topic has been reviewed and a new computationally efficient technique for computing the so called indices of fuzziness has been put forward, together with a new purely automatic technique for locating the positions of thresholds developed. The question of line thinning and the generation of a coded representation of the lines that constitute the image has also been looked at and important existing work in these aspects has been reviewed. A new approach to the problem of generating a coded representation of an image has been set out. This involves two interconnected and interrelated stages. The first is referred to as a soft skeletonization technique in which either a grey-scale, or a two-tone, image is processed and a pseudo-grey-scale image is produced using a center of gravity technique. It is then explained how this image is used in a stride encoding operation to produce a low redundancy representation of the image. The results presented, although limited, indicate that the technique is itself likely to be useful as part of a system for the automatic recognition of images. The technique is more efficient (needs fewer bits) than existing schemes such as chain coding,* and it seems to be capable of removing certain unwanted features, such as small serifs, and to be capable of bridging small unwanted gaps in and between strokes.

*In the above mensioned results, for instance, approximately 30 points were employed and a codeword of 5 or 4 bits was required to express each of them. This means that between 120 and 150 bits are needed as compared with 372 for the chain code.

REFERENCES

[3.1] S.K.PAL, R.A.KING and A.A.HASHIM: "Automatic grey level thresholding through index of fuzziness and entropy", Pattern Recognition Letters 1 (1983) 141-146, North - Holand Publishing Company

[3.2] W.H.PRESS, B.P.FLANNERY, S.A.TEUKOLSKY, W.T.VETTERLING: Numerical Recipes, Cambridge University Press 1986.

[3.3] C.J.HILDITCH: "Linear skeltons from square cupboards." Mach. Intell. 4, pp 403-420 (1969)

[3.4] R.STEFANNELLI, A.ROSENFELD: "Some Parallel Thinning Algorithms for Digital Pictures", J. Ass. of Computer Machinery, Vol. 18 no 2, April 1971.

[3.5] N.J.NACCACHE, R.SHINGHAL: "An investigation into the skeletonization approach of Hilditch.", Pattern Recognition, Vol 17, No 3, pp 279-284, 1984.

[3.6] P.A.MARAGOS, R.W.SCHAFER: "Morthological Skeleton Representation and Coding of Binary Images", IEEE trans. Acoustics,Speech and Signal Processing, Vol. ASSP-34 no 5, Oct 1986

[3.7] H.FREEMAN: "Computer processing of line drawing images." Comp. Surv. 6, 57-97, (1974)

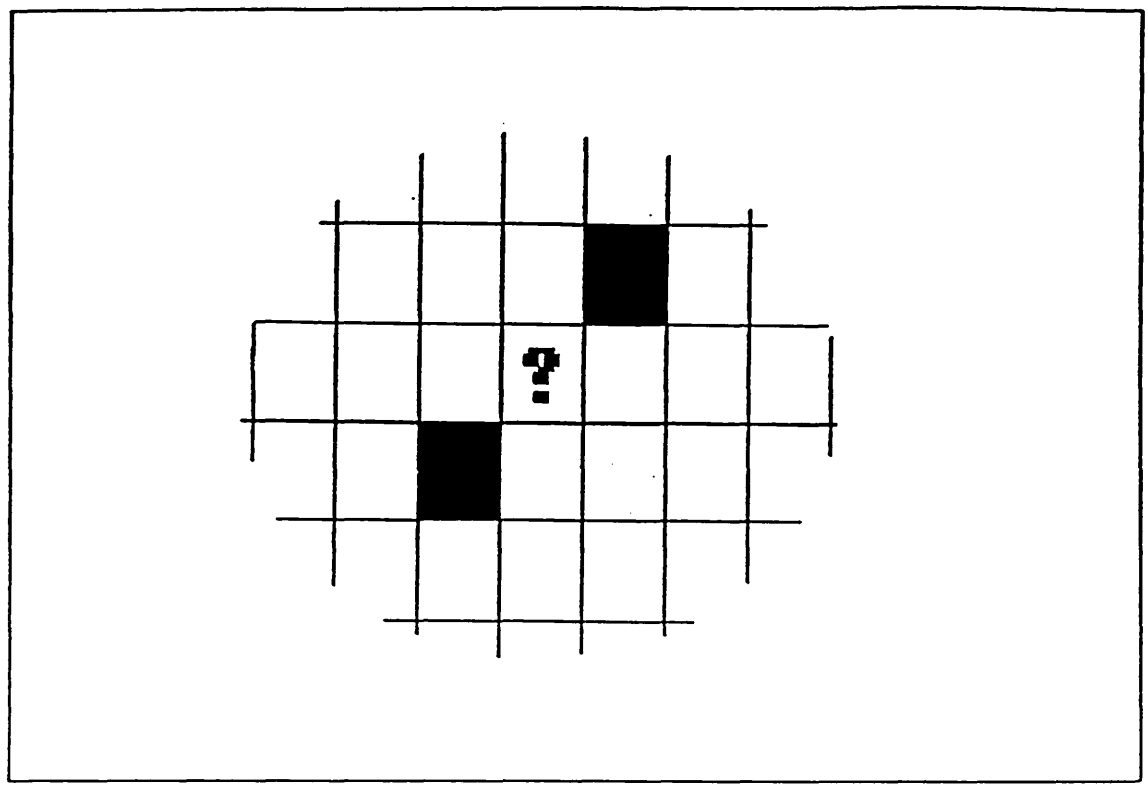


Fig 3.1
STATISTICAL DEPENDENCE BETWEEN PIXELS

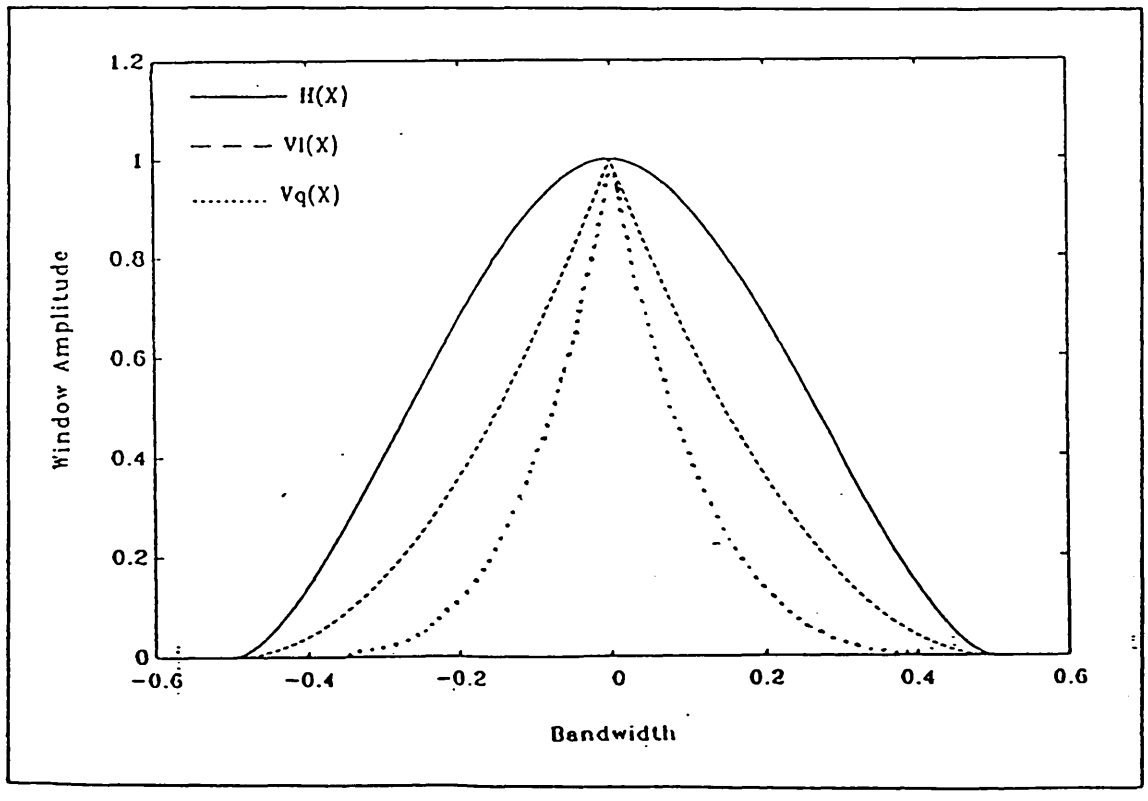


Fig 3.2
WINDOWING FUNCTIONS

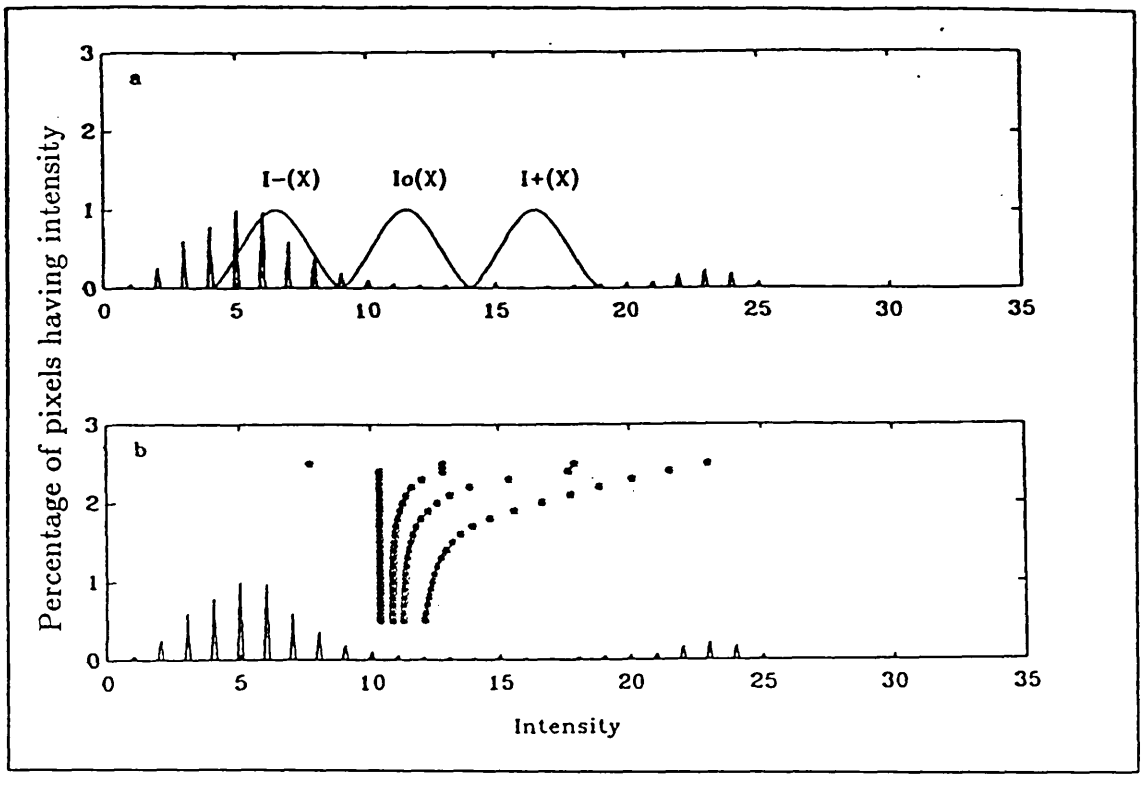


Fig 3.3
ANNEALING THRESHOLDING ON A BIMODAL HISTOGRAM

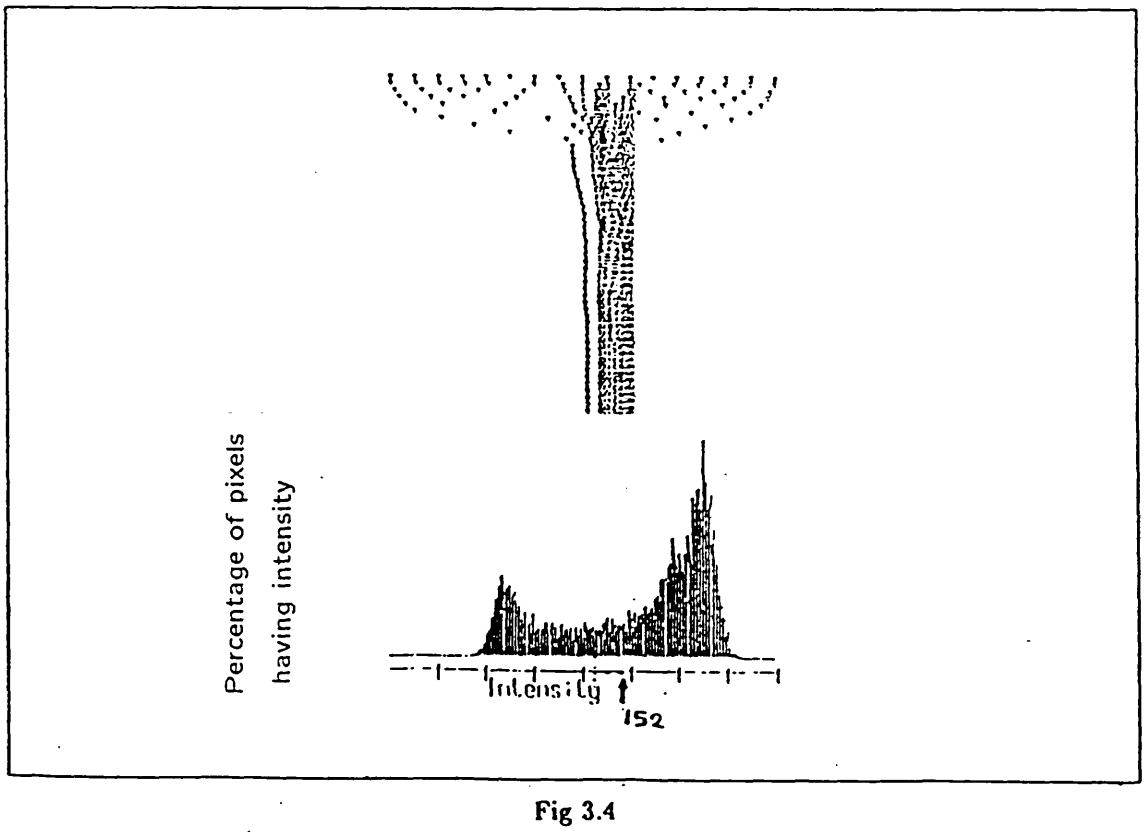
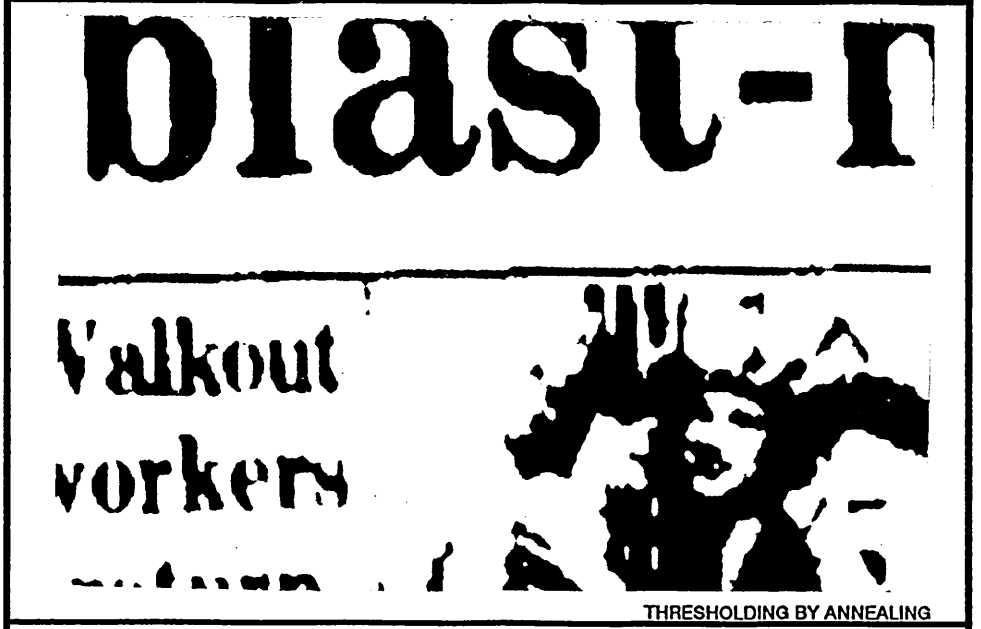


Fig 3.4
ANNEALING THRESHOLDING ON A NEWPAPER IMAGE

a



b



c

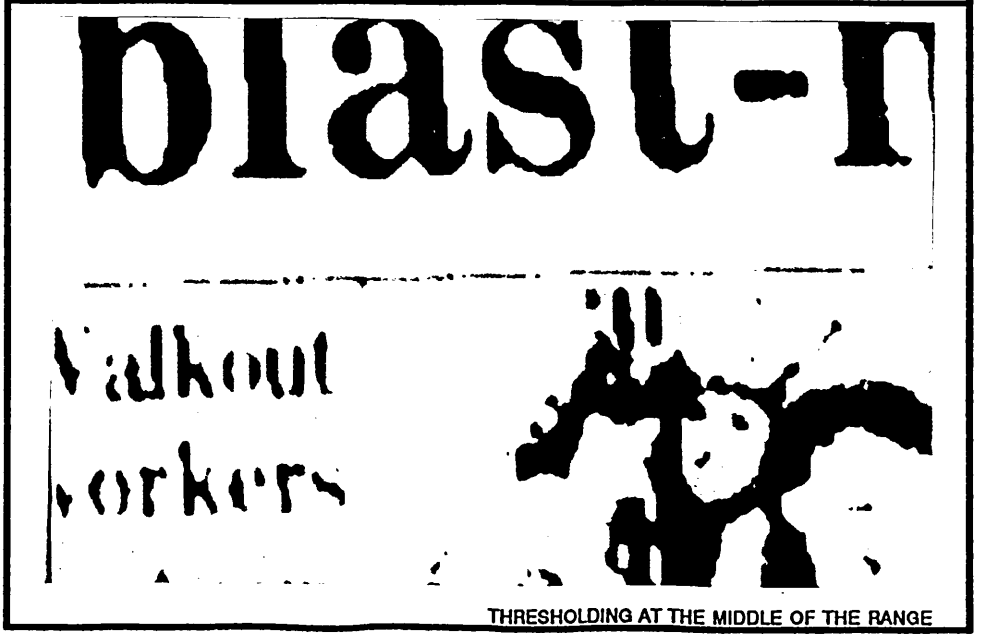


Fig 3.5 THRESHOLDING OF A NEWSPAPER IMAGE

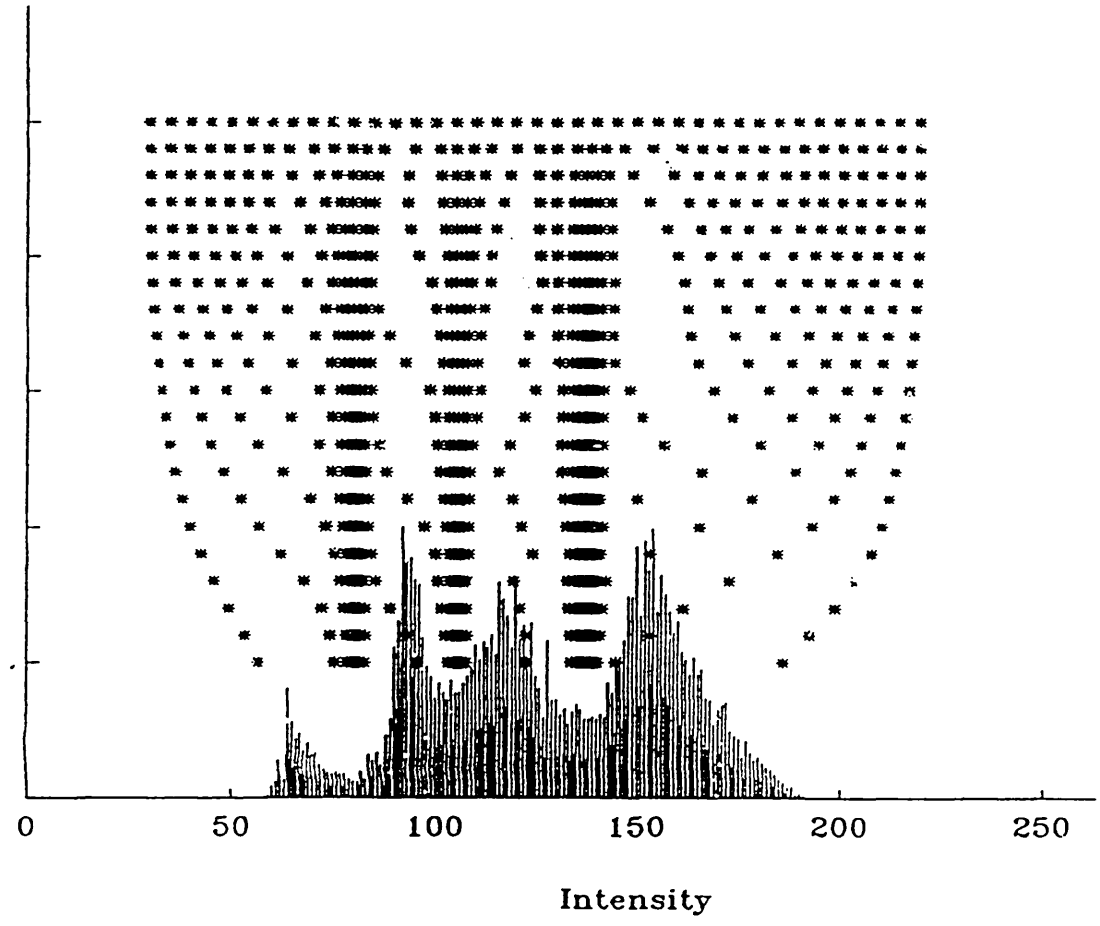


Fig 3.6
ANNEALING THRESHOLDING ON A MULTIMODE HISTOGRAM

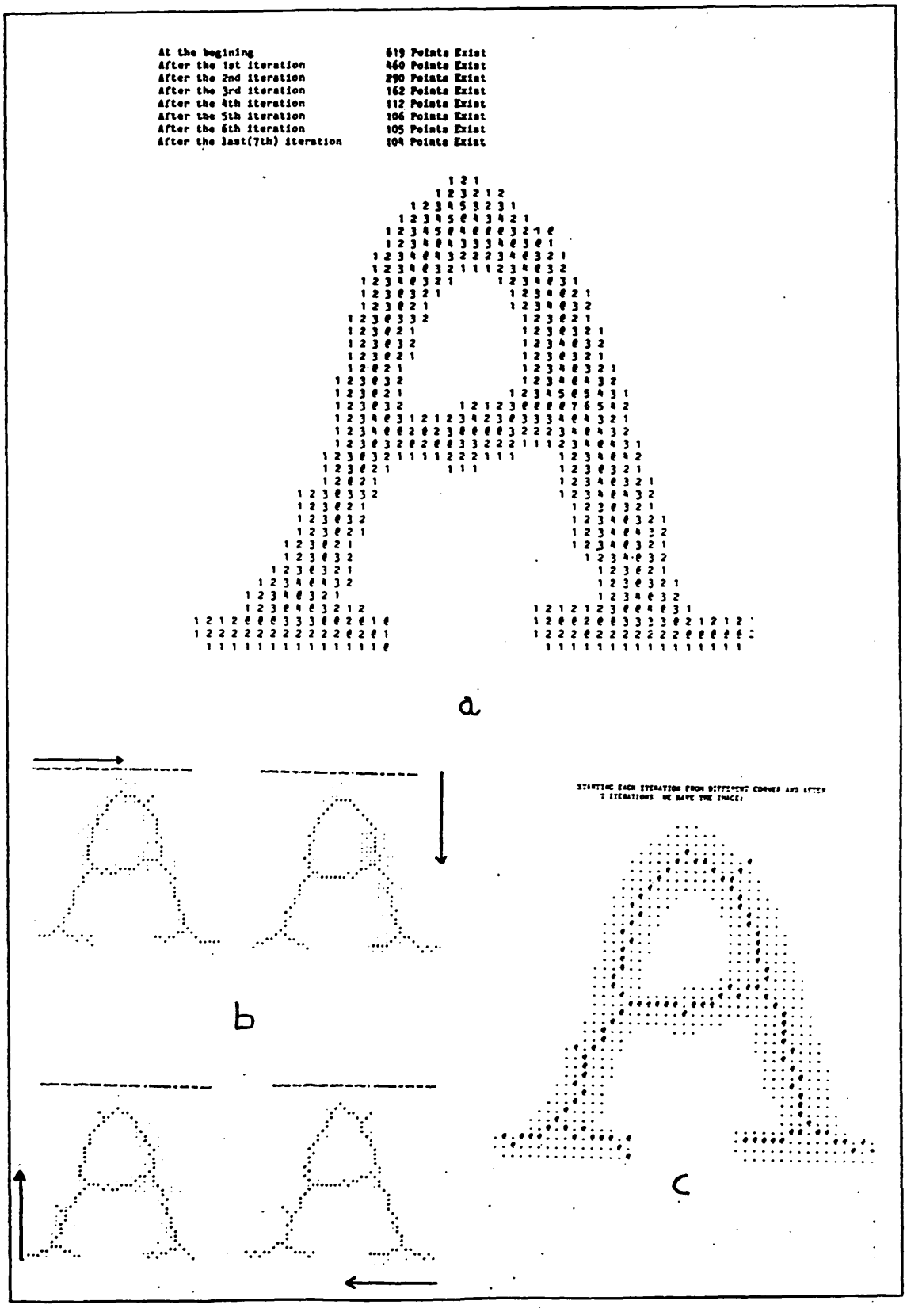


Fig 3.7

HARD SKELETONIZATION BY THE NACCASHE-SHINGHAL ALGORITHM

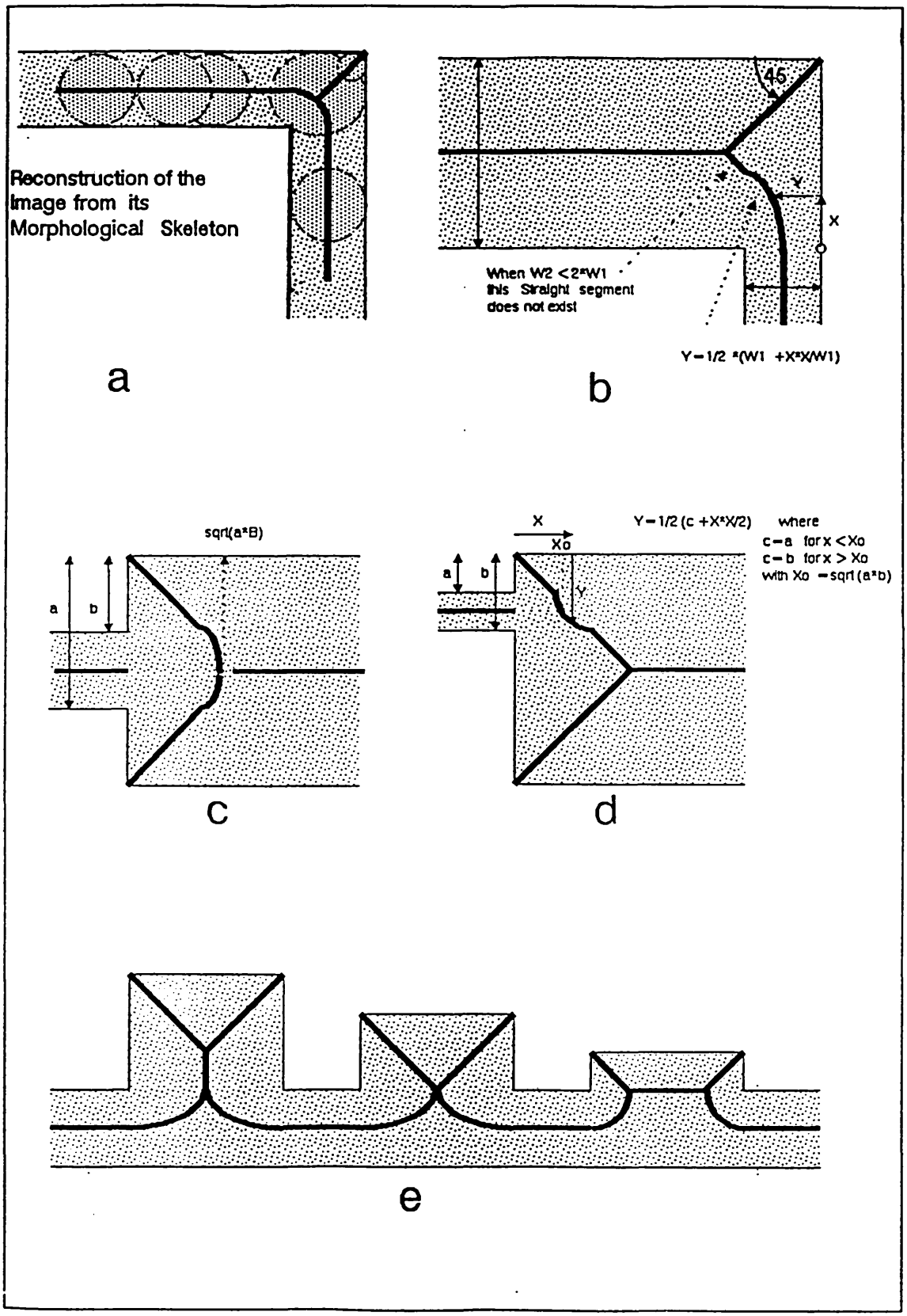


Fig 3.8

MOPHOLOGICAL SKELETONS

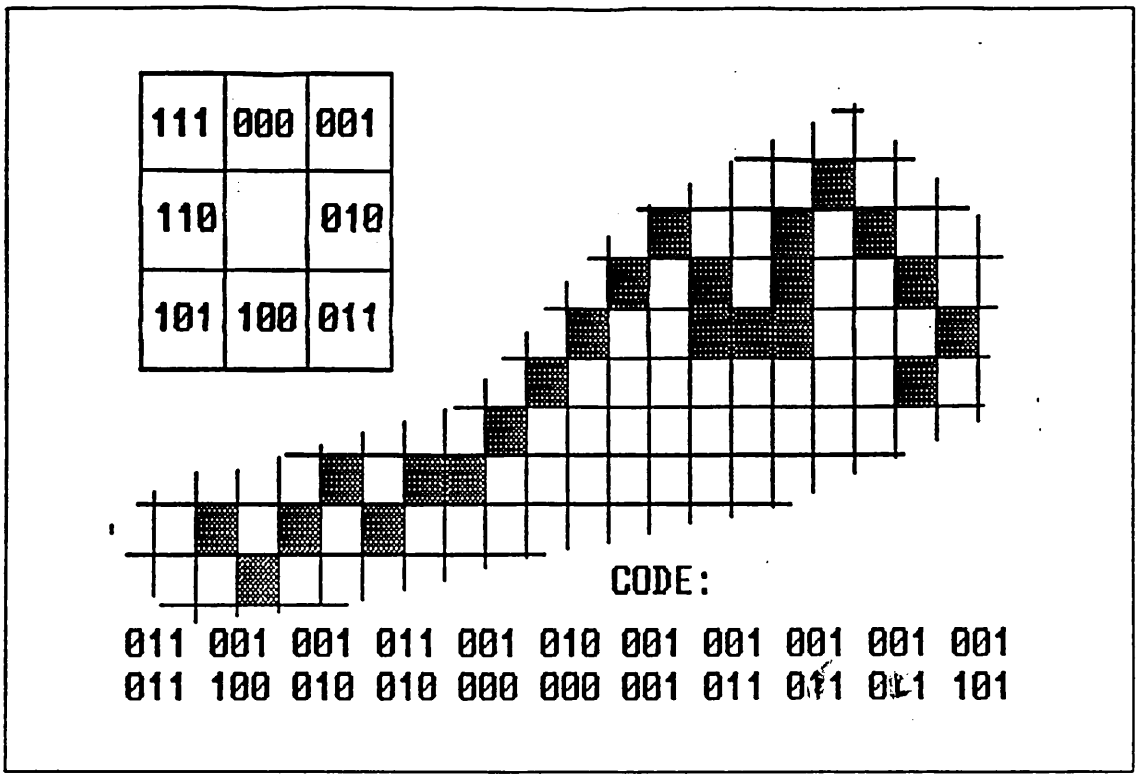


Fig 3.9

FREEMAN'S CHAIN CODE

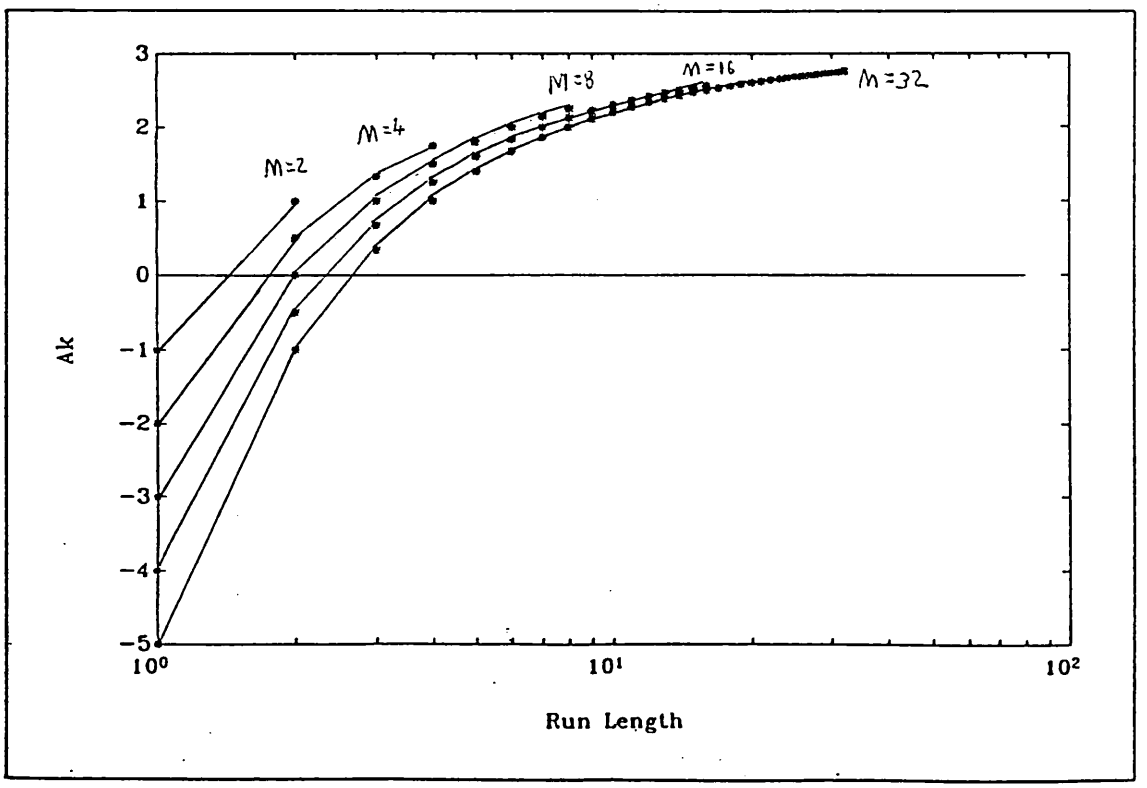


Fig 3.10

EVALUATION OF THE PERFORMANCE OF RUN LENGTH ENCODING

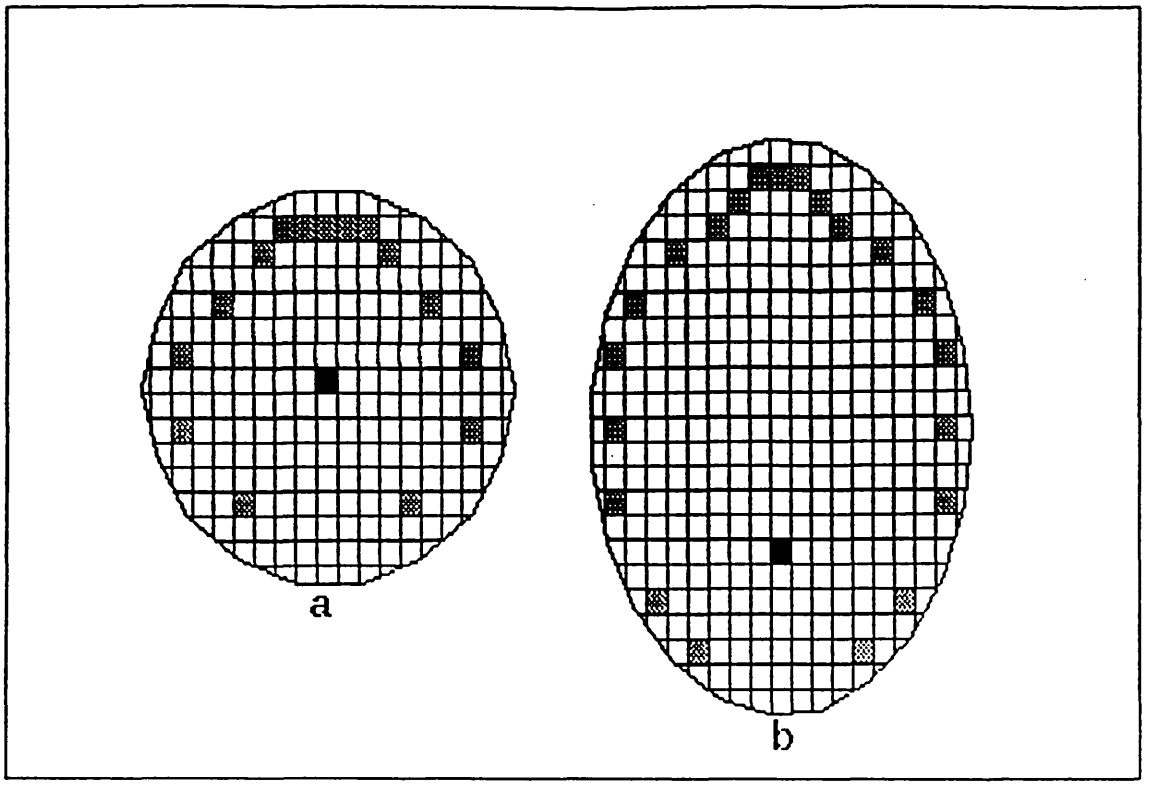


Fig 3.11

"NEXT" POINTS: NON UNIFORM DISTRIBUTION ON CIRCLE AND ELLIPSE

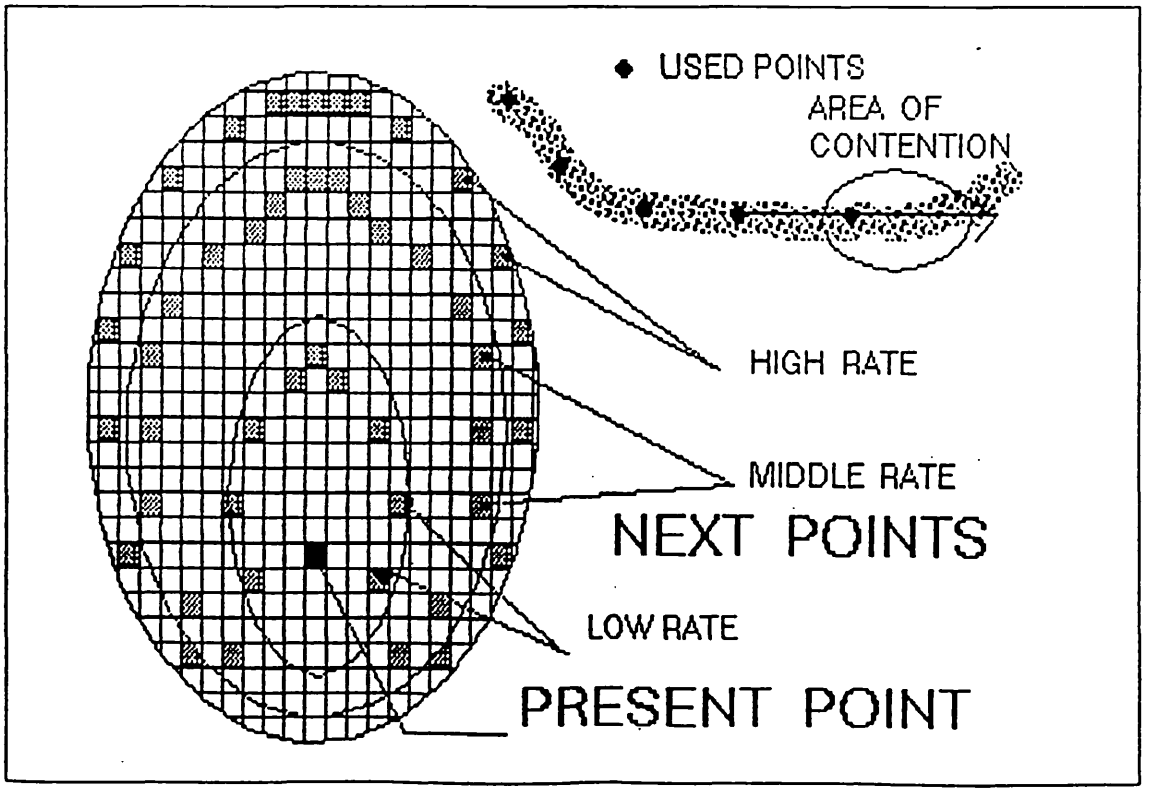


Fig 3.12

AREA OF POSSIBLE "NEXT" POINTS WITH VARIABLE SIZE OF STEP (STRIDE)

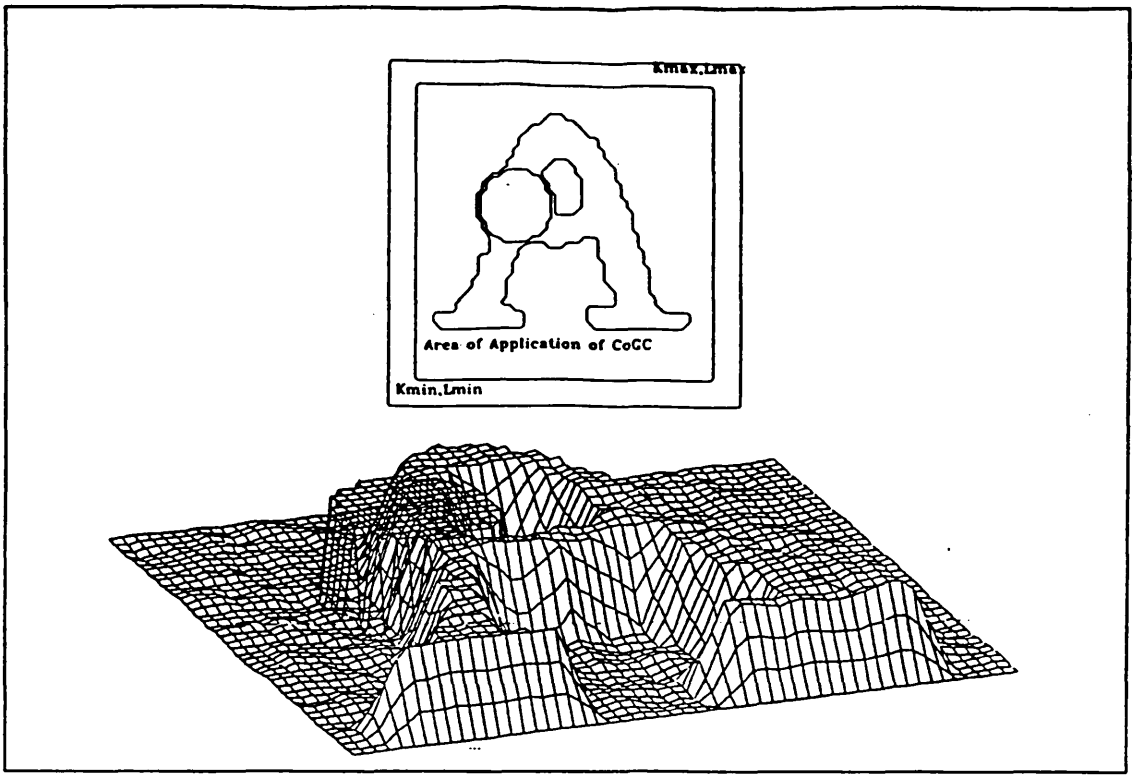


Fig 3.13

SOFT SKELETONOZATION

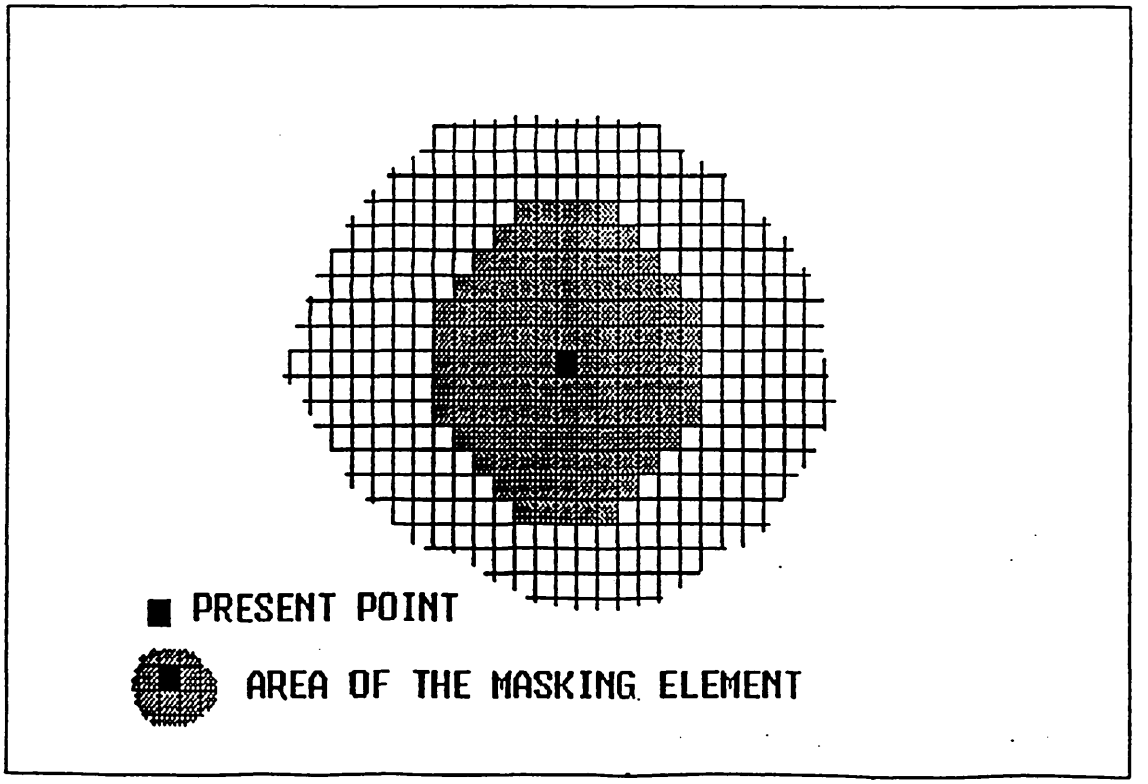


Fig 3.14

MASKING ELEMENT

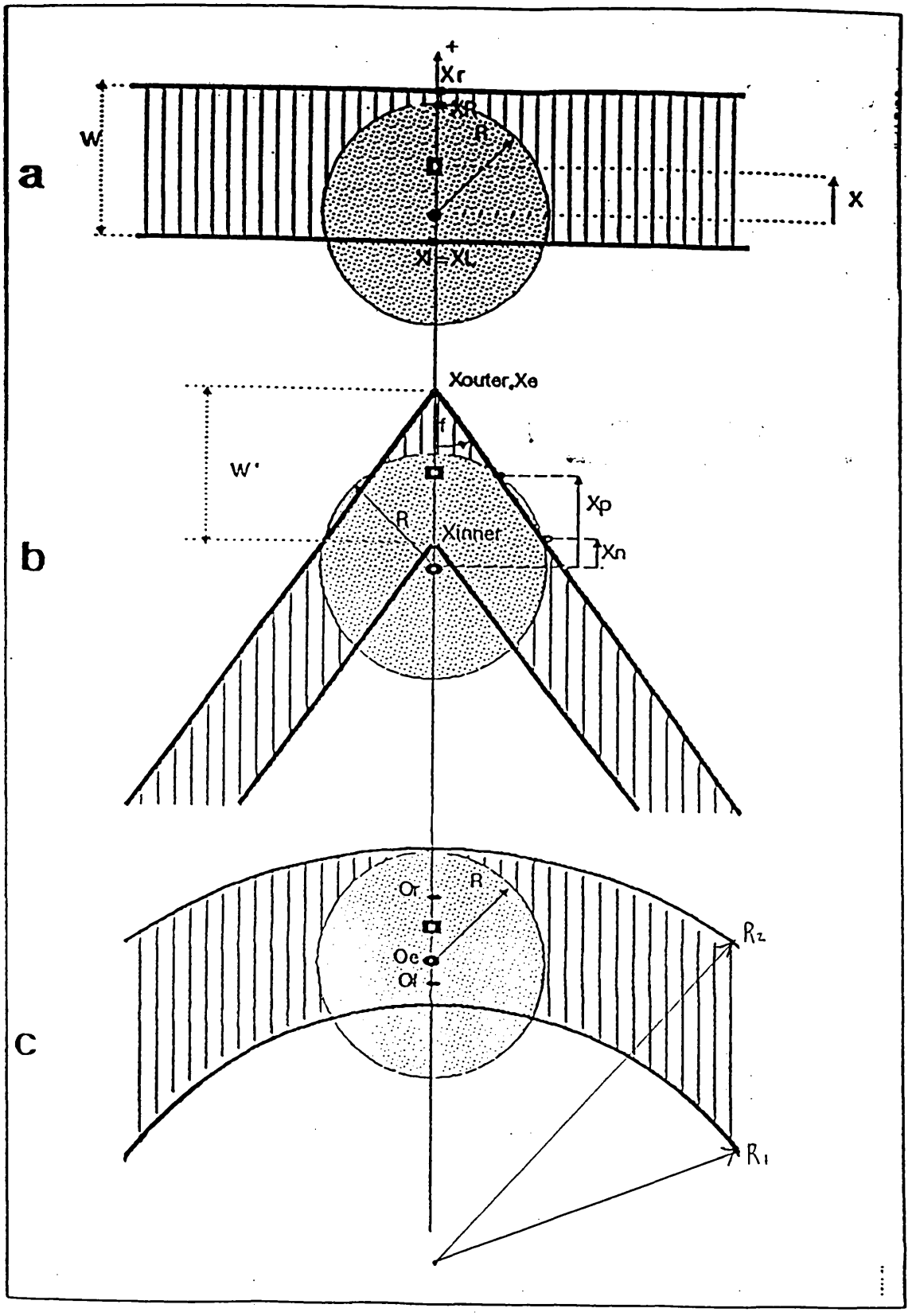


Fig 3. 15

CENTER OF GRAVITY CRITERION FOR CONSTANT WIDTH STROKES:
POSITION OF THE MASKING ELEMENT

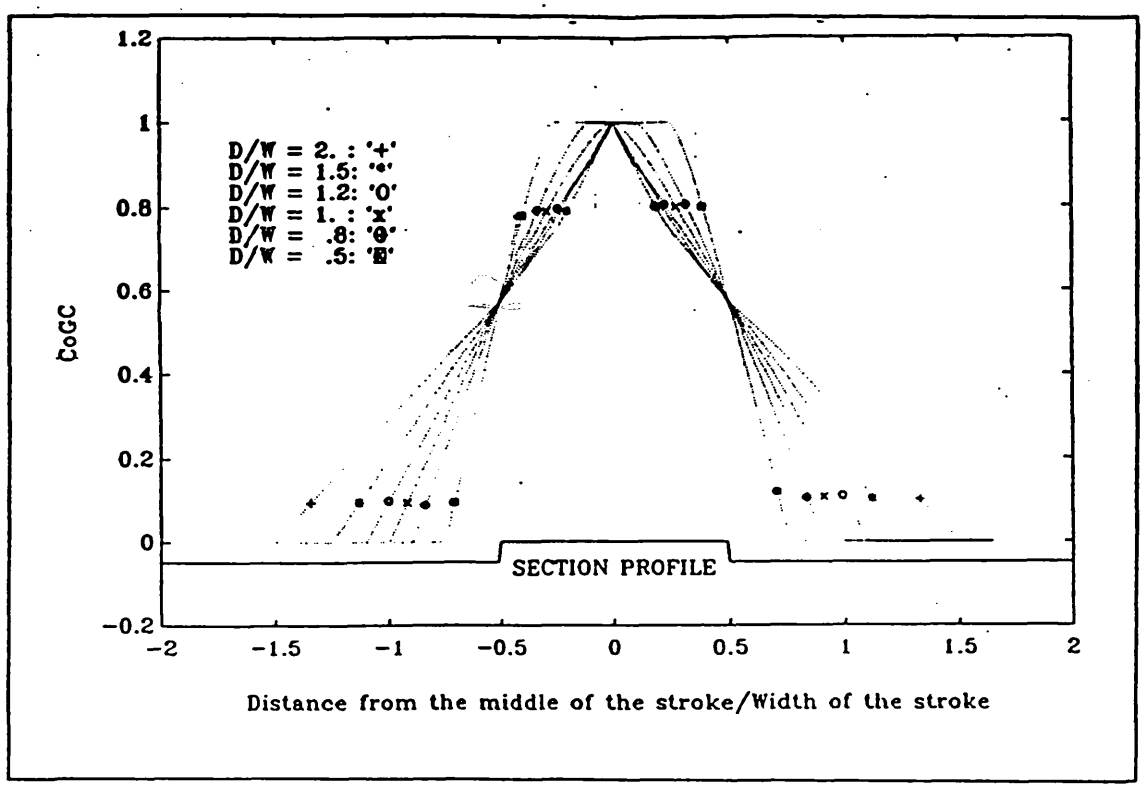


Fig 3.16

CoGC FOR STAIGHT STROKES

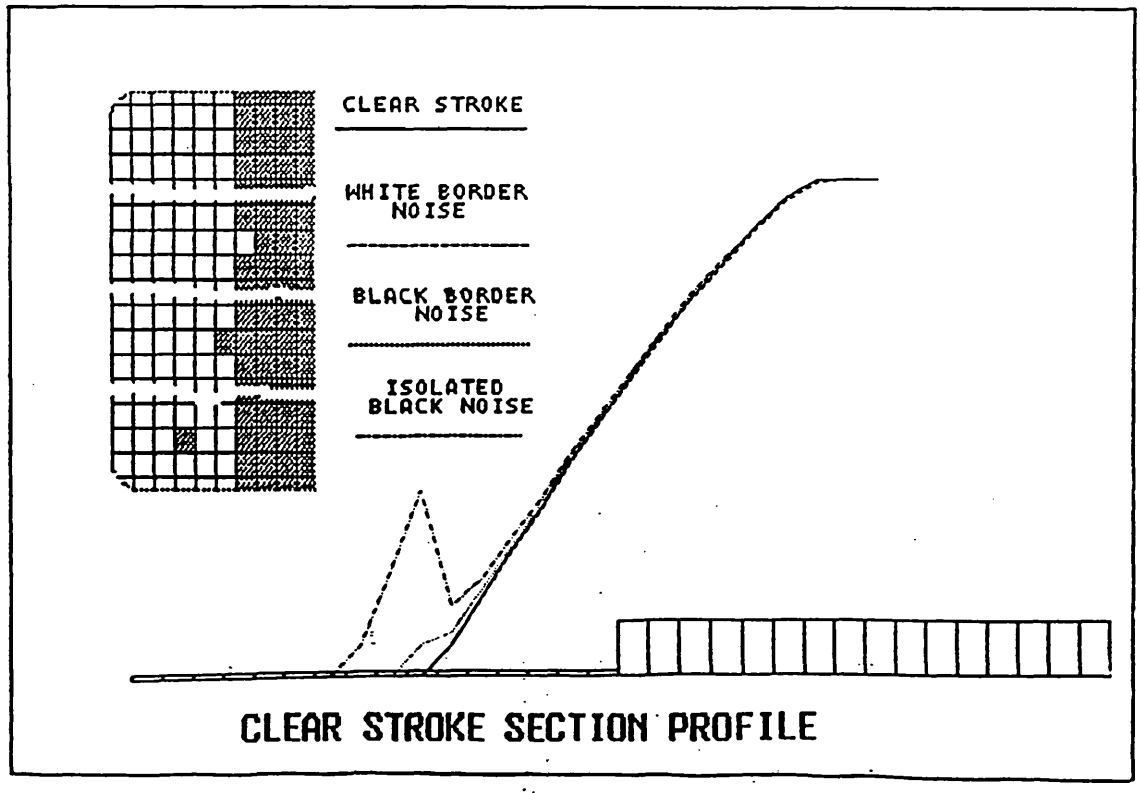


Fig 3.17

EFFECT OF NOISE ON CoGC FOR STRAIGHT STROKES

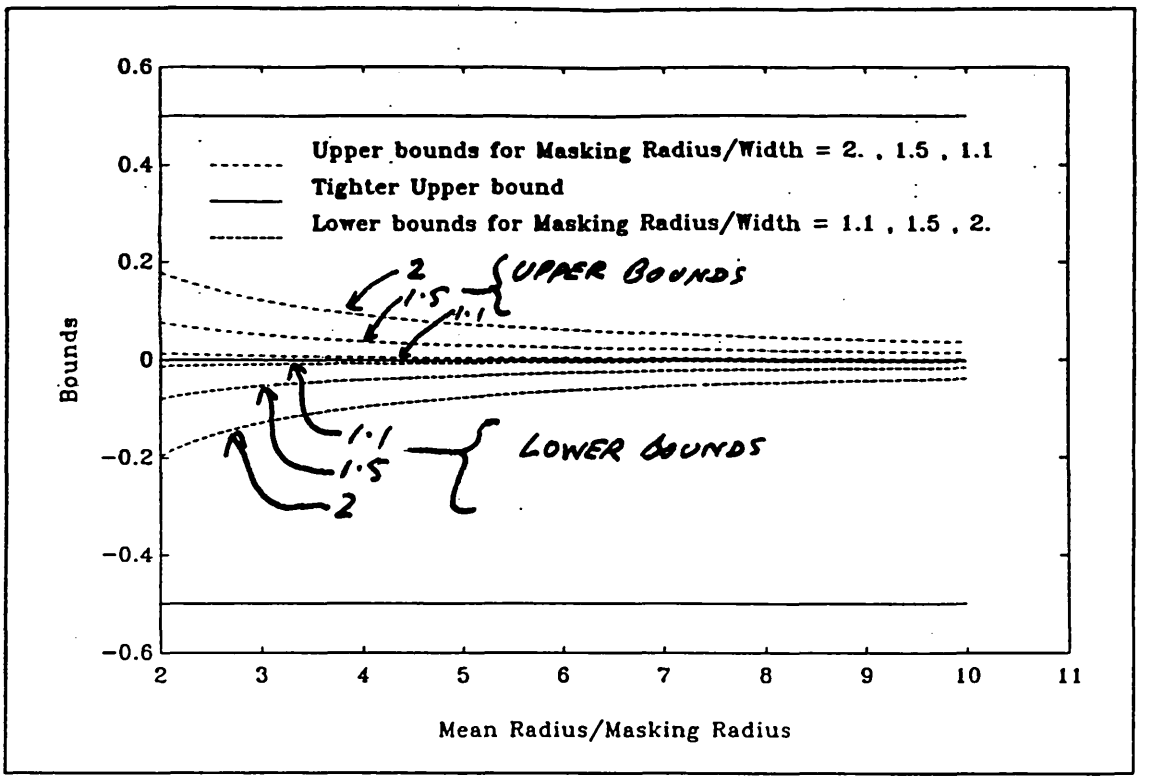


Fig 3.18

BOUNDS TO THE POSITION OF MAXIMUM COGC FOR CURVES OF CONSTANT RADIUS

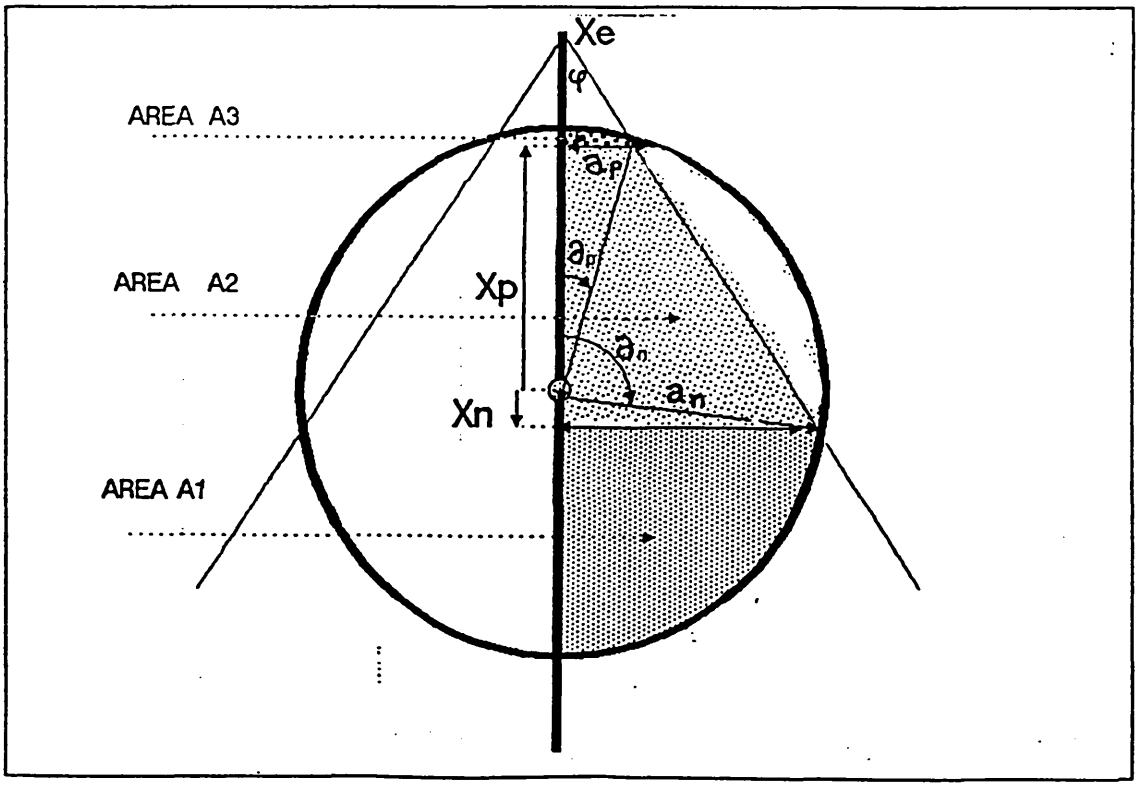


Fig 3.19

EVALUATION OF COGC FOR ANGLES

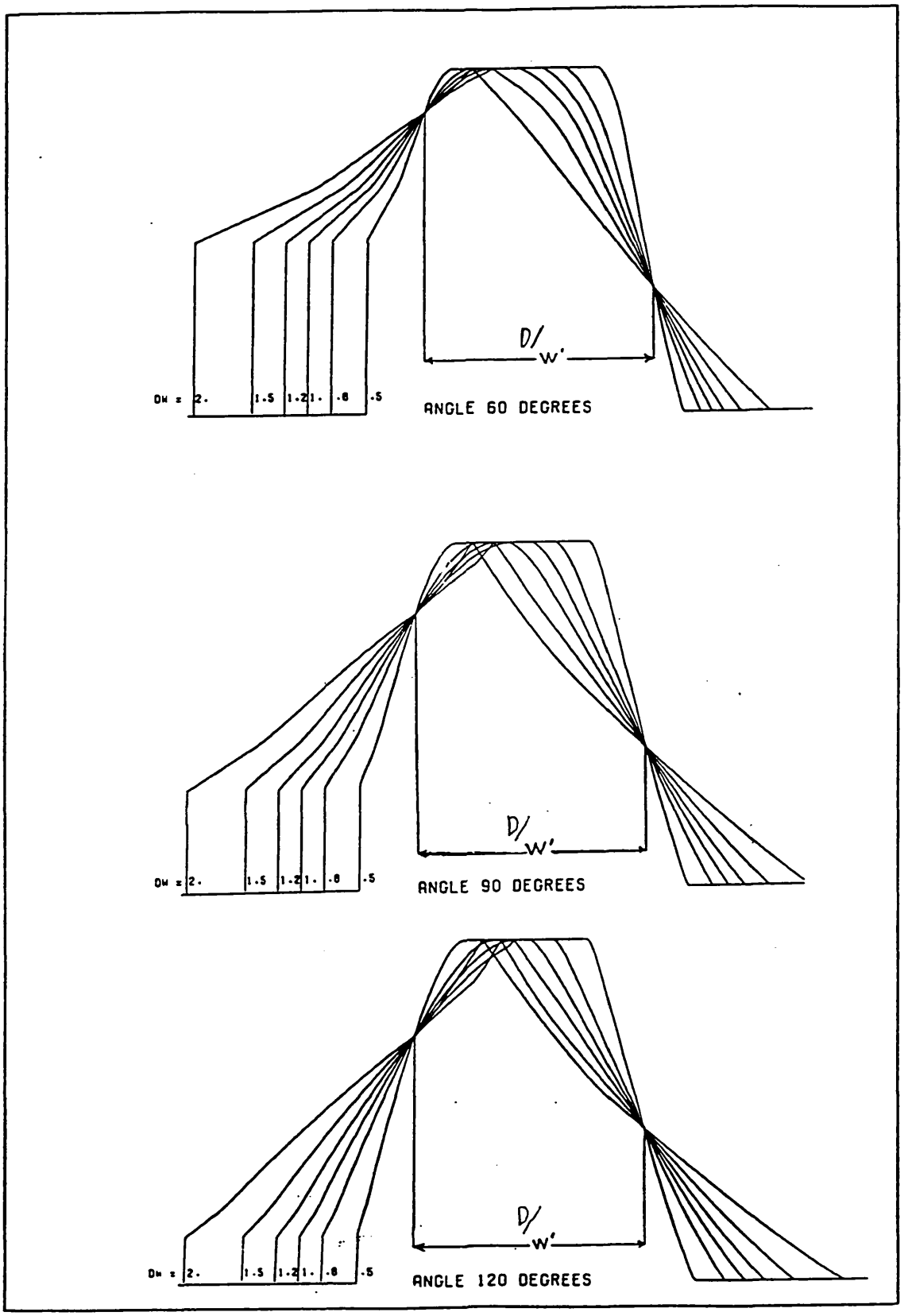


Fig 3.20

CoGC FOR STROKE INTERSECTIONS OF ANGLE 60°, 90° AND 120°

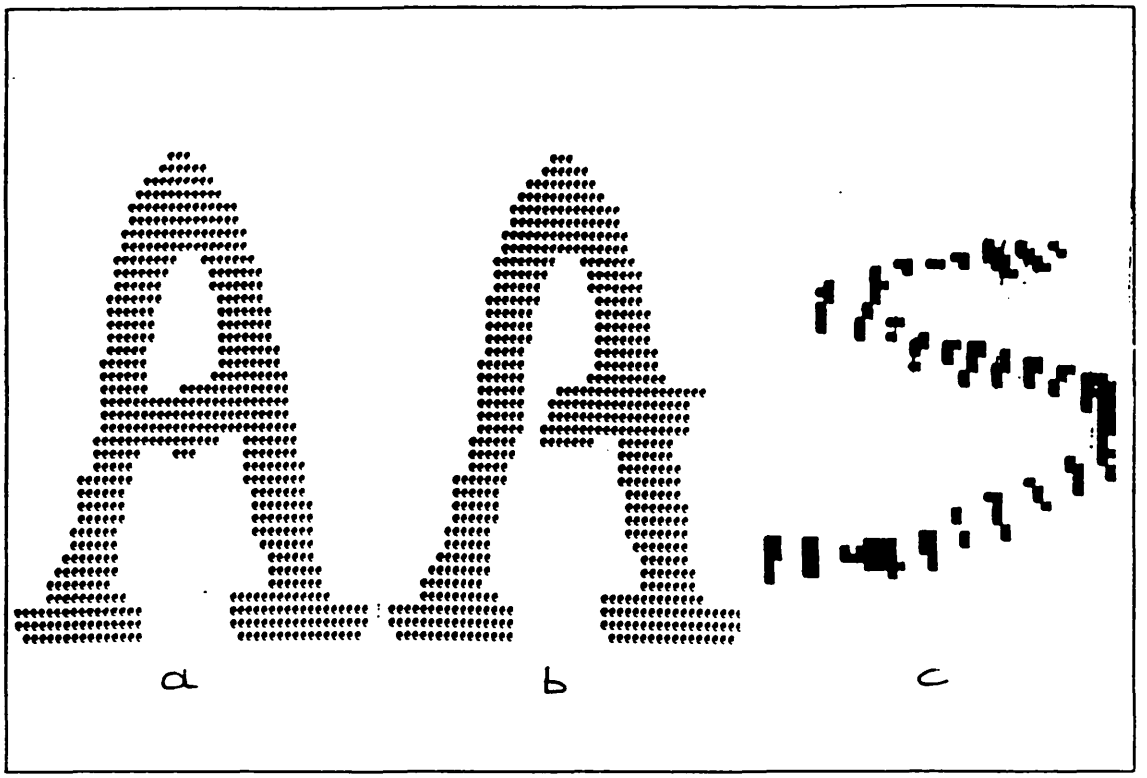


Fig 3.21

THE TWO-TONE CHARACTER IMAGES CONSIDERED IN THE EXAMPLES

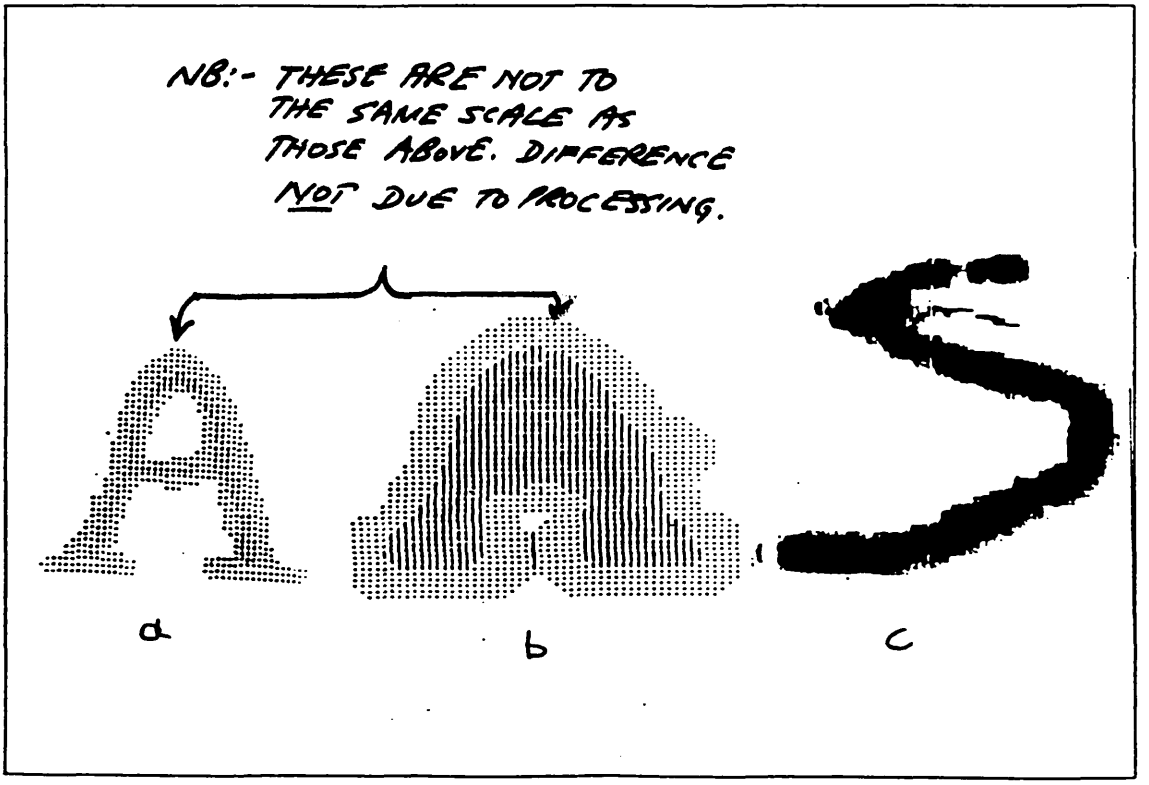


Fig 3.22

SOFT SKELETONS BASED ON COGC

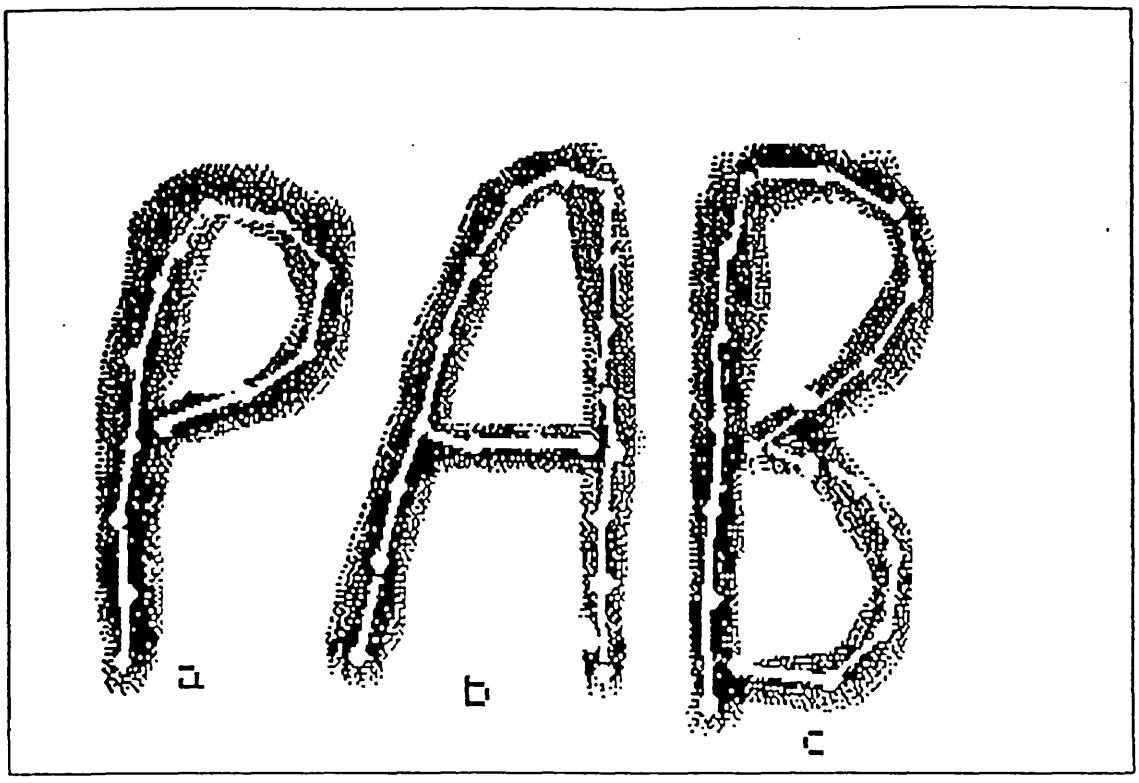


Fig 3.23

LOOPS DURING STRIDE ENCODING

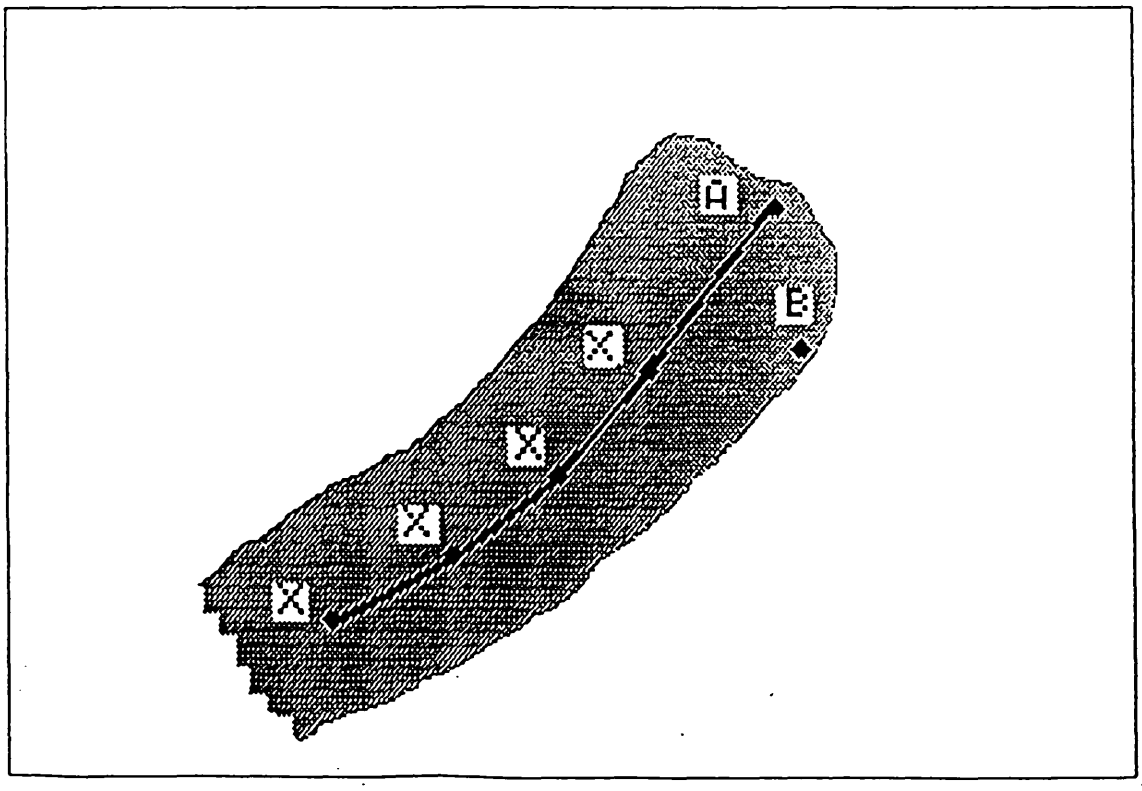


Fig 3.24

THE STRIDE ENDODER AT THE END OF THE STROKE

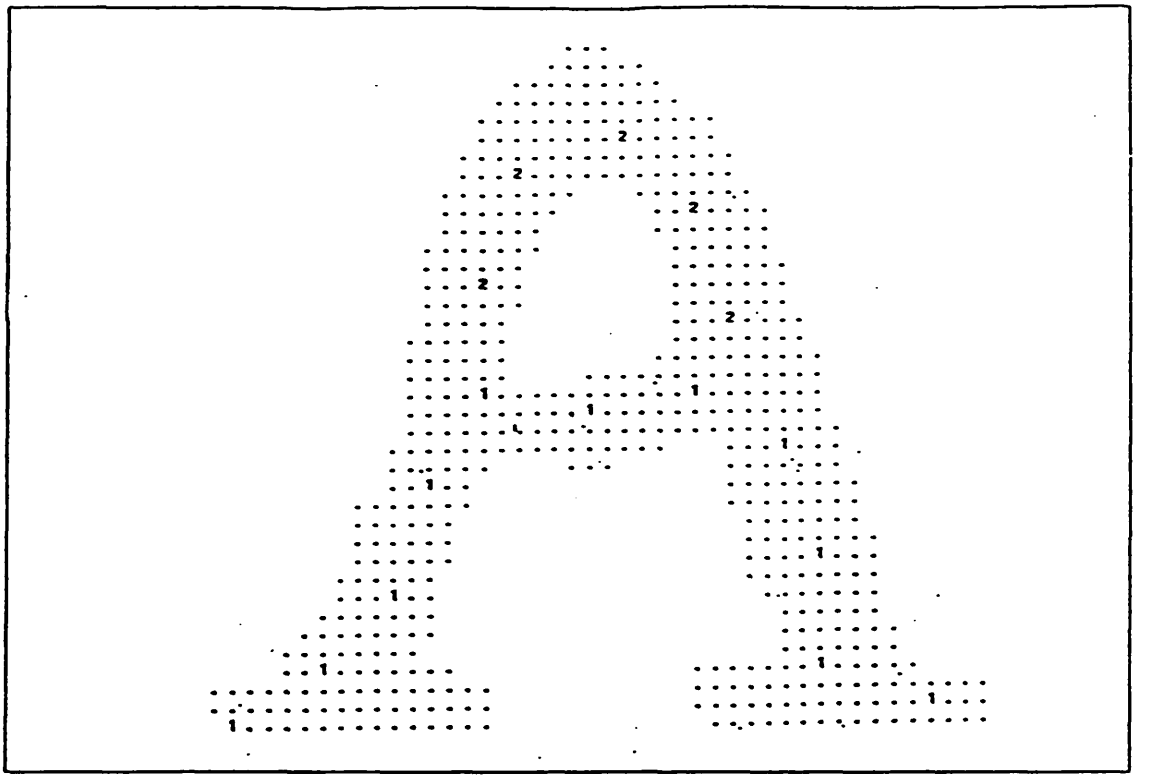


Fig 3.25

STRIDE ENDODING WITH UNWEIGHTED AREA OF NEXT POINTS

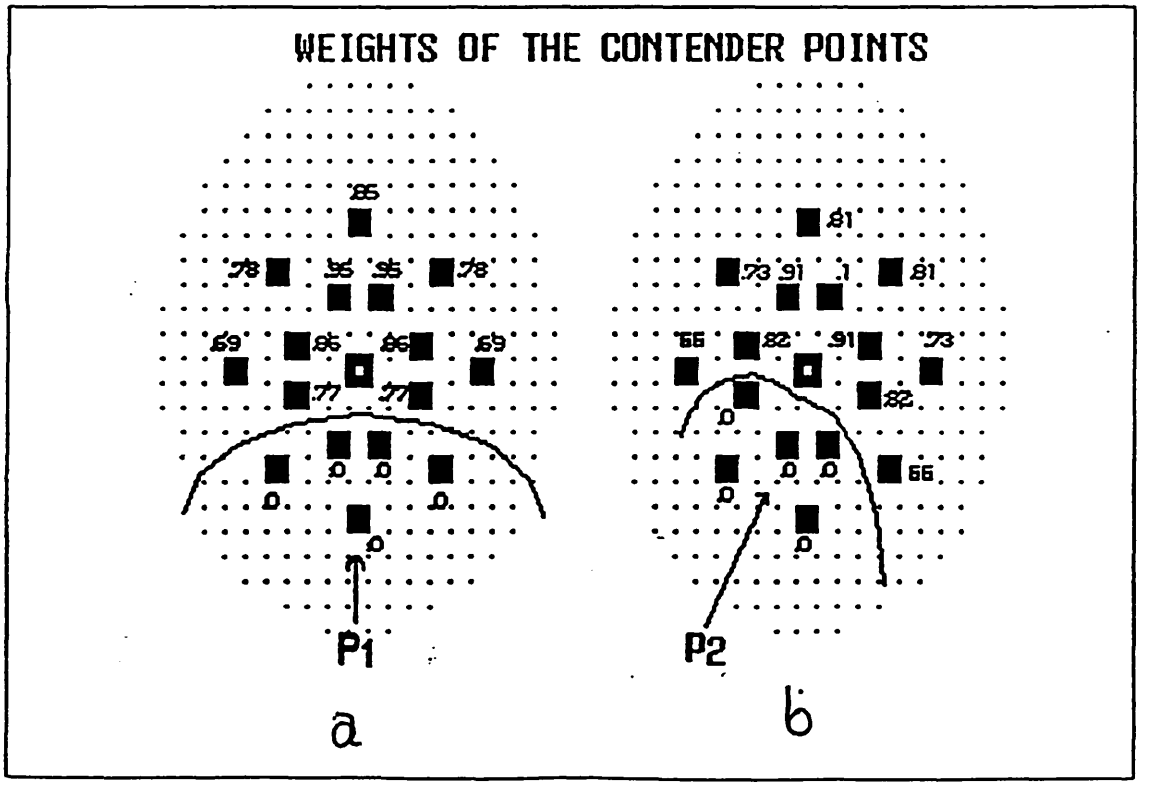


Fig 3.26

AREA OF NEXT POINTS FOR THE STRIDE ENDONDING SHOWN IN FIGS 3.27 AND 3.28

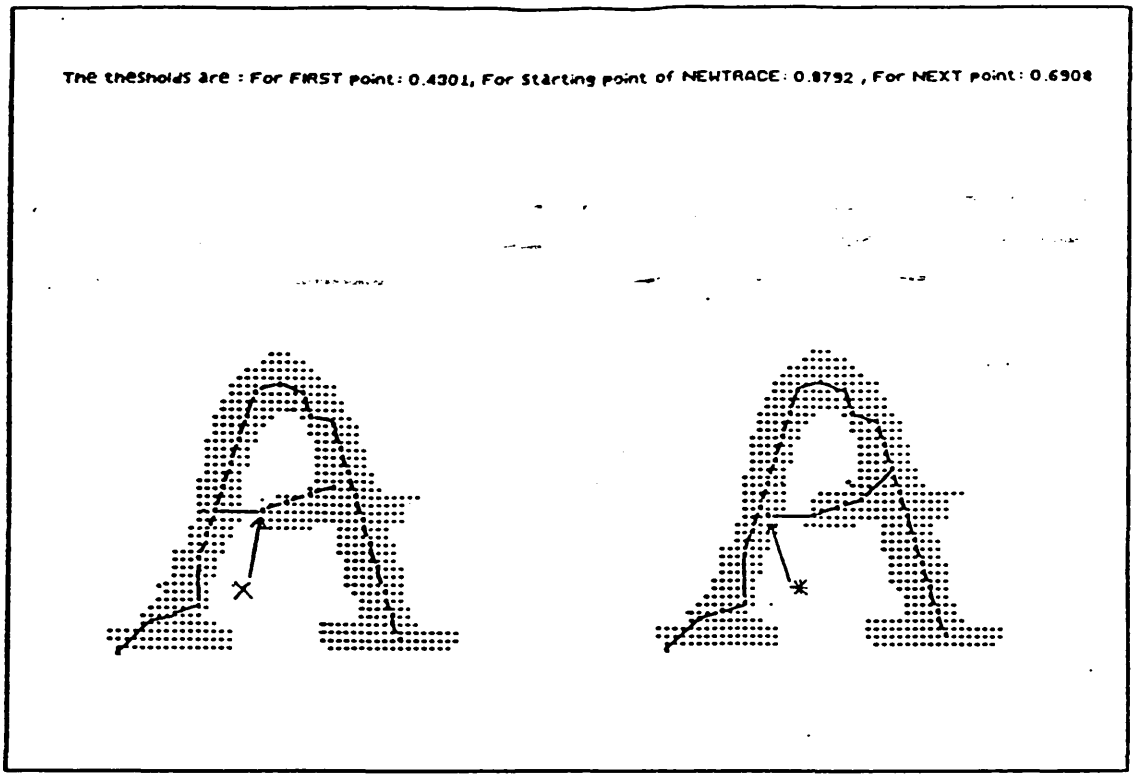


Fig 3.27

STRIDE CODE OF "POORLY FORMED A"

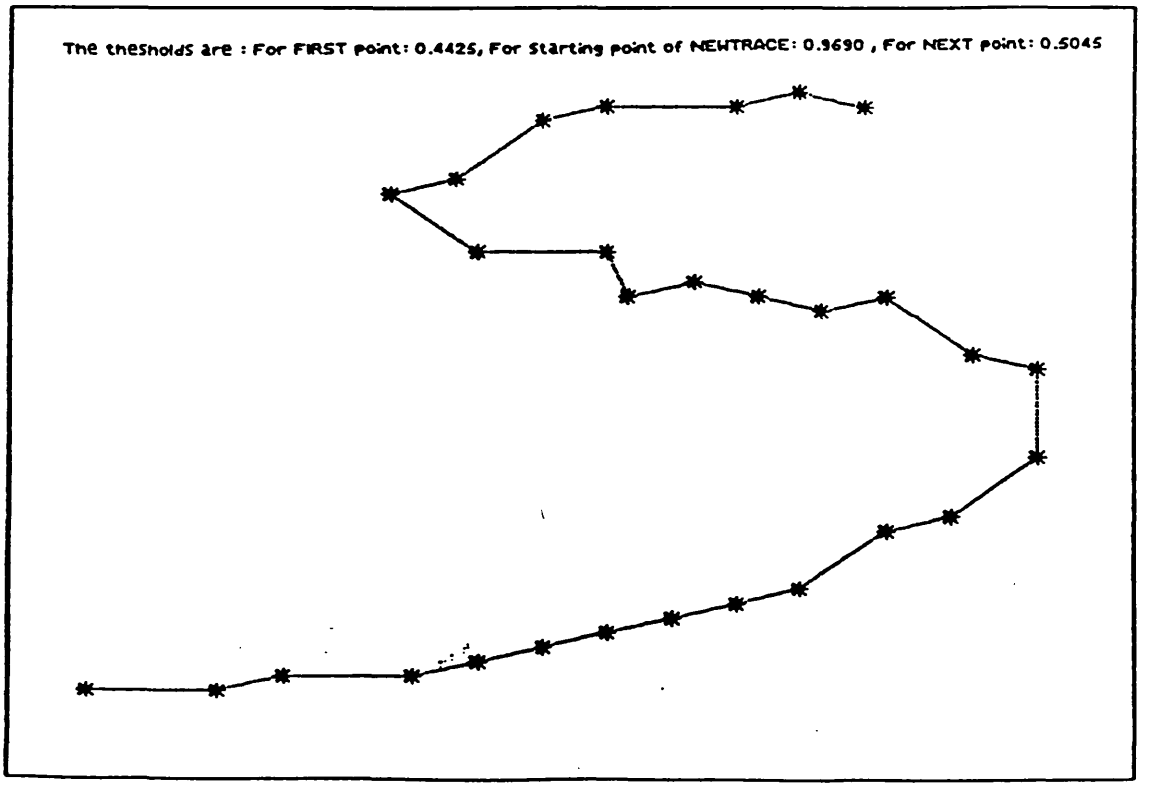


Fig 3.28

STRIDE CODE OF "DISCONTINUOUS S"

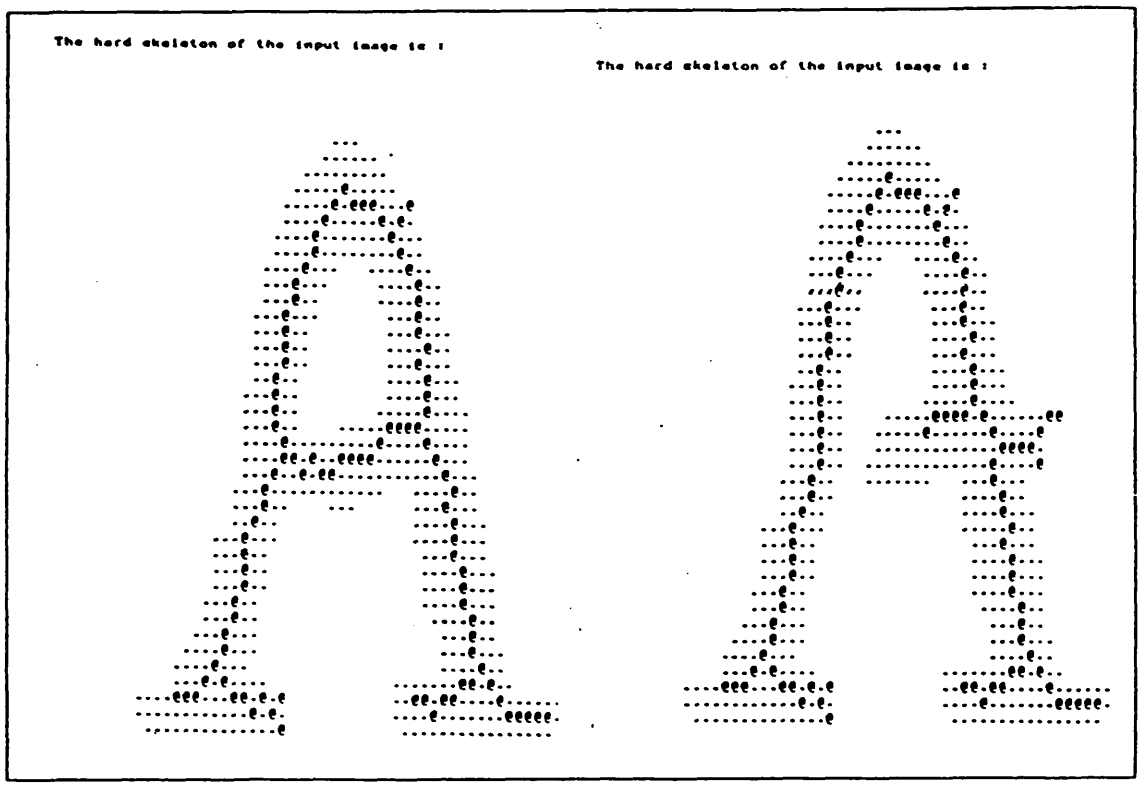


Fig 3.29

HARD SKELETONS OF "A" AND "POORLY FORMED A"
USING THE NACCASHE-SHINGHAL ALGORITHM

Chapter 4
Aspcects Related To The Modelling Of Handprinted Characters

CHAPTER 4
ASPECTS RELATED TO THE MODELLING OF HANDPRINTED CHARACTERS

4.1 INTRODUCTION

4.1.1 The Concept of a Handprinted Character

In the previous chapter the problem of the encoding of handprinted characters was considered. Each character image was treated as a separate entity and no attempt was made to identify a character with some predetermined pattern. The aim of the encoding process was to achieve a low-bit representation of a character image whilst retaining sufficient information to enable characters of a particular class to be distinguished by a machine from characters in other classes. As with the conventional chain encoder, the end product of the encoding approach, involving the stride encoding of a soft skeleton described in the previous chapter, is a one-dimensional representation of the character. The one-dimensional representation takes the form of a string of symbols taken from a specific alphabet and the string is such that, using a set of generic rules, it is possible to obtain from the information contained in the string a line image representing the original character that was encoded; that is, the encoding process is reversible to the extent that a representative image can be obtained which the human being can recognize as being the "same" as the original character.

As was explained earlier, the next stage in the machine recognition of characters is that of classification. In the classification part of the recognition system, the string of symbols referred to above is taken and a further entropy reduction is employed so that a significantly reduced set of symbols is obtained. This set of symbols, in the case of the 26 letters of the English Alphabet and the numerals 0-9, can simply be a 6-bit word. This 6-bit word uniquely distinguishes between the pattern classes, and could be used to synthesize

(print) a pattern that will be recognized as belonging to and representing a particular class of patterns.

Before considering the problem of the actual classification of handprinted characters, it is important to note that no unique objective definition exists as to what a particular handprinted character is, and that there is no obvious physical set of characteristics that can be attributed to any one character. An immediate important consequence of this is that it is difficult to arrive at a "standard" image that can be used effectively to machine recognize (classify) correctly those handprinted images that would be classified by the human being as belonging to a particular class. The idea of using a standard representative pattern for each different class is, in fact, the basis on which simple template matching works. In the absence of such a standard, the only really effective course of action is to store a sufficiently large number of patterns which represent the significant variations within a particular pattern class. In the case of machine printed characters, on the other hand, the range of patterns is significantly reduced and hence template matching becomes an implementable technique of recognition.

Summarizing, the difficulty in the recognition of loosely defined patterns such as the handprinted characters, compared with the recognition of well-defined patterns, such as the machine-printed characters, is that with the former, there is no ideal (standard) pattern (possibly with a limited number of known deviations) from which, after undergoing various potentially observable transformations, the full range of patterns constituting the class can be, or is, derived.

In the absence of a unique objective definition of a character it seems fair, at this time, to argue that a particular character is assigned to a particular class simply because the human being so classifies it; and not because it has any definable likeness to some ideal image representing the pattern class. An immediate important consequence of this is that the human being is the ultimate judge of the accuracy of the classification process performed by any other system which is designed to "recognize" images that represent characters

as defined by the human being.

The arguments set out above have significantly influenced the approach adopted in this thesis to the problem of defining characters with a view to their ultimate machine recognition. A character will be considered as a global structure of locally identifiable elements and the modelling of such a structure will be discussed. The importance of this approach is that if effective modelling could be developed then it would significantly influence the ease with which the classification process could be carried out and it would throw light on other important factors such as the synthesis of characters from rules.

Before turning to the actual modelling of characters it is necessary to give some consideration as to what is meant by the words "global", "structure", "locally", "identifiable" and "elements" when they are used in the context set out above.

— Global: The size of the "space" over which the human being recognizes characters differs significantly according to the kind of writing that is being examined and there are strong indications that the human being does not in fact read isolated characters but deals with syllabaries, or sometimes whole words. Experiments (Ref [4.1]) have shown, for example, that the speed of reading when mixed case (upper and lower) printing is used, is much lower than the speed when either upper or lower case letters alone are used. However, nearly all the research in OCR has, up to now, been concerned with the reading of individual characters. The reason for this is that the system complexity increases exponentially with the number of characters considered as a group. In the context of this thesis the term "global" will be taken to mean the recognition of isolated characters which can be considered as being contained within some rectangular "box". It is important to note that a recognition system that works entirely with isolated characters, generally, will have a performance that is inferior to the one that takes advantage of redundancy which may be in the form of linguistic or contextual constraints.

A clear parallel exists here with the material presented in the Chapter 2, in which, although it was not stated explicitly, hard decisions were assumed to

have been taken in respect to the recognition of isolated binary digits and context, in the form of redundancy associated with the error correcting code, is used to correct erroneous decisions made in respect to the isolated digits.

— Structure: A character pattern belonging to a given class is more than just an ordered set of elements (such as strokes). For example, in the case of the letter A, it is not sufficient to say that the character is a set of strokes with the first stroke being taken from the set in Fig 4.1a, the second from the set of Fig 4.1b and the third from the set of Fig 4.1c; although each of these sets contains strokes that might be a constituent element of some character A. It is necessary in describing a character to consider it as a stochastic phenomenon and to describe in probabilistic terms the properties (such as length etc) of the various elements involved and the probabilistic relations between these elements.

— Locally: Each of the elements that constitute a character is recognizable locally, within the rectangle ("box") referred to above. The size of this local area varies from element to element and from pattern class to pattern class, and it may range from being very small to being a substantial part of the whole "globe" (the rectangle). It is possible, in an adaptive system, to adjust the local area and to adjust the definition of what constitutes an element on the basis of previously gained information. For example, a previously observed sample of printed text may indicate that "strokes" below a certain length are serifs and as such, can be considered as being of little significance in machine recognition.

— Identifiable: In the case of handprinted characters it is clear that a very large range of alternatives exists within each class and that no exact statement exists as to what values (size, curvature, angle etc) define each element of the class. It seems likely that the human being does not make "hard" decisions in respect to the separate elements which may be considered as constituting a character and that soft decisions are probably involved, with some form of likelihood being associated with the various elements, which precedes the making of a hard recognition decision at some higher level. In other words, it is likely that a pattern may not have a set of unambiguous identifiable characteristics.

In any attempt to model a character class, using a model employing a finite number of elements, with an associated discrete probability distribution, it is inevitable that some approximations are involved in respect of some members of the class when the model is used to represent the class. If such a model is used to represent a class and it is then employed in the classification part of the system, then decisions will be probabilistic.

— Elements: In the discussion above reference was made to elements ("strokes", "curves", "angles"). The choice of elements for use in modelling characters is at the centre of the character recognition problem. It dictates the complexity of the model used and the subsequent ease with which characters can be recognized. Clearly, the defining of essential elements is a difficult matter since it appears that the human being recognizes characters in a hierarchical way; and at this time, there is no clear consensus as to what constitutes the basic elements and no definitive list of "elementary" parts is known to exist.

Clearly, there is a trade-off between the actual size of the elements into which a character is decomposed and the complexity of the associated model that signifies the character structure. The same trade-off exists between the effort to identify the elements and the effort involved in recognizing the whole character.

If it is decided to describe a character in terms of large-size elements (strokes and curves), then the resulting model is expressed in very simple, and probably deterministic, rules. Thin straight lines and thin curves can be recognized well by geometrically justifiable methods, but a major difficulty in using those methods in practical character recognition is associated with the fact that the strokes constituting the lines and curves are often wide relative to their length. The wide nature of strokes makes it difficult to decide on the exact shape of them. Another difficulty arises from the fact that it is often far from clear where one stroke ends and the other begins.

The second approach is to keep the elements short and incorporate them

into a complicated model. An example of such a sequence of short elements is the sequence of vectors produced by the stride encoder scheme contained in the previous chapter. Such a complicated model must be stochastic and must be able to express relations across different parts of the image. A description in which each part of the character has its own model, and in which this is independent of the model of the other parts, is clearly not likely to be appropriate, expect for fairly simple characters (shapes).

A combination of the long-element and short-element approach results in the creation of a two- (or multi-) stage model with the large elements being considered as consisting of a sequence of elements (primitive elements) connected together in terms of a parametric model. After the large elements constituting the image have been identified, they are labelled and the appropriate statistical information associated with the parameters of the model is appended. For example, after a sequence of basic elements has been identified or interpreted as a large line, the label is fixed to the long element and then statistical information regarding its length (one of the parameters defining it) is further added.

Following the identification, labelling and provision of statistical information relating to a particular element, the second stage of the model is then developed. This involves defining and providing statistical information relating to the interrelation between long elements.

4.1.2 Use of Character Synthesis in Model Development

In the previous subsection the idea of using a probability based model to represent handprinted characters was introduced and discussed briefly. It was pointed out that very little exists in the way of precise description of the pattern class involved. It was argued that the final judge as to what constitutes a particular character class is the human being. In consequence, it follows that the best way forward in formulating a model to represent a class of characters is to make use of the somewhat less than precise ideas that the human being has

as to what constitutes the human consensus as to how characters must appear. The general philosophical question is whether it could be possible to formulate an appropriate model, or models, without human intervention and, as yet, no definite answer has been provided. The question of validating a model as a representation of characters (and more general patterns) is interesting in itself. Conventionally, models in the form of feature representation of a character have been validated by designing a feature extraction system and an associated recognition system. The accuracy and effectiveness has been assessed in terms of the percentage of correct recognition achieved. This clearly provides one way for evaluating model representations of the kind introduced in the previous subsections.

An interesting alternative in validating a model is to use the model as a means of synthesizing characters (patterns) and then for a human being to decide whether or not the synthesized patterns are accurate and acceptable as representations of members of a class of patterns. The question of model design and validation will not be considered further in the thesis but the synthesis approach to validation is worthy of detailed investigation.

4.2. FINITE STATE MACHINE MODELLING OF PATTERNS

4.2.1. Intoduction

In the previous section it was pointed out that, in general, a vast variety of patterns (shapes) can exist within a pattern class and that, in the absence of a firm objective definition of a standard pattern, the best way forward is to establish a probabilistic model which can be used to represent members of a pattern class or classes. The importance of such a model in the decision making part of a character (pattern) recognition system was also pointed out. In the remainder of this chapter attention will be focused on the development of a model which might be used to represent a pattern class. No attempt will be made to become directly involved in the related decision making process, but the significance of the model to that process will be emphasized.

It is clear that by using the full range of possibilities associated with a high level computer language, for instance, it is possible, relatively easily, to program a machine so as to generate virtually any of the patterns within a particular pattern class.

EXAMPLE:

The simple high level program

```
REPEAT
        PLOTLEFT
        PLOTRIGHT
UNTIL  RAND ≤ θ
IF RAND ≤ φ/2          PLOTLEFT
IF RAND ≥ 1− φ/2       PLOTRIGHT
STOP
```

is capable of generating a large variety of symbols that would probably be classified as the Greek letter Λ.

In this high level program, the subprogram PLOTLEFT (PLOTRIGHT) plots a segment of length l, at angle $-\alpha$ (α) respectively relative to the vertical, starting from the point at which the segment finished in the previous invocation of the same subprogram; $RAND$ is a random variable, $0 \leq RAND \leq 1$, the parameter θ is a parameter which, in conjunction with the parameter l, controls the length of the strokes; and ϕ is a parameter expressing the probability of uneven length of non-equal stroke lengths (one stroke is allowed to be one unit of length, l, longer than the other).

The program is such that the length of the two strokes are identical discrete correlated random variables. By generalizing the program it is possible to arrive at a program which is capable of generating a greatly increased range and variety of the letter Λ. □

Although it is possible to program a machine in the manner discussed above so as to produce patterns that are classified by the human being as being acceptable and hence can be considered as belonging to the same class; that is, it is relatively easy to develop a model that can be used to generate a pattern

class, it is much more difficult, however, to start with an observed pattern and then to decide from which model it was most likely to have come. Clearly the conceptually trivial approach in which each model is allowed to generate the full range of patterns and the observed pattern is then compared with each different pattern belonging to a particular class is, in general, prohibitively difficult to implement in practice. The significance of the step in going from an observed pattern to the model that was most likely to have produced the pattern should be fully appreciated. If this stage could be accomplished easily, then, since a particular model would be known to be associated with a particular pattern class, the recognition process would also have been accomplished, and simply done.

It is well known (Ref [4.2]) that in the case of Stochastic Finite State (FS) machines[1] it is possible to go from an output and to answer the question of whether, and if so with what probability, a particular machine could produce the output. The ease with which this question can be answered depends on the complexity of the finite state machine and the complexity, that is, the length, of the string of symbols comprising the output. The structure of a Finite State machine is, however, not ideal as a model for representing complicated patterns, such as hand-printed characters. Such a machine is not capable of expressing in succinct terms the correlation between parts of the pattern that might be widely separated.

Despite this difficulty, the ease of analysis, and the ease in determining the probability that a particular machine might produce the output, suggests that it might be advantageous to consider the use of such a machine in pattern modelling.

The motivation for modelling has been touched on above. A more specific illustration of the significance of FS machine modelling is the following.

Consider the situation illustrated in Fig 4.2 which shows two FS machines

---

[1]The term "Finite State" machine will be used in this thesis to mean a Stochastic Finite State machine.

and a sequence, or string, of symbols that might represent a pattern to be classified. If it were possible relatively simply to estimate reliably the probability that the sequence was generated by the FS machine number 1, and similarly the probability that it has been generated by the FS machine number 2, then , in essence the recognition process has been performed provided that it is known which class of patterns each of the machines is capable of generating. For example, if it was known that machine number 1 generates patterns considered by the human being to be fully representative of the letter A and machine number 2 was known to generate patterns considered to represent the letter H, then an estimate of the probabilities that the machines generated a specific pattern that is to be classified between the classes A and H will constitute a maximum likelihood estimate and this could be used simply in the decision making part of the recognition process.

4.2.2. Modelling

A pattern (or character) can be represented as a concatenation of symbols taken from a finite alphabet. Such a concatenation can be referred to as a string of symbols or, more simply, as a string. An example of such a string may be the output obtained from a stride encoder, similar to that described in the previous chapter. The aim of the modelling process is to arrive at a description of a mechanism (that is, a FS machine in the present case) that is capable of generating strings that are plausibly the same as those contained within the ensemble that is associated with the pattern class.

The first decision to be taken concerns the constraints on the mechanism to be used and the criterion of plausibility. The imposing of constraints on the model is related directly to the desired properties of the model, such as low complexity, and to the need to be able to arrive at a model. This later point is related to the fact that, although a complicated model might be better able to generate plausibly the set of strings than a simple model, it may not, in fact, be possible to arrive at a statement of the complicated model and its parameters from a consideration of the ensemble of strings that the human being associates

with the pattern class. Another reason for imposing constraints on the nature of the Finite State machine is that it must take into account any a priori knowledge that exists as to the way in which strings are generated. For example, the fact that a FS machine capable of operating so as to produce with maximum probability the set of strings associated with a pattern class may not be the correct model if it does not, for example, take into account the fact that it is known a priori that the strings are produced one symbol at a time.

With regard to the question of plausibility it is necessary, as was suggested above, to arrive at some acceptable and useful criterion of plausibility. There are three well established criteria that have been developed which are quite general in their application, which are applied as measures of plausibility within the context considered in this section.

The first is a development of the familiar Maximum Likelihood statistics used, for example, in hypothesis testing. Let it be assumed that there exist a set S of n strings and let it be assumed that there are two FS machines, i and j, that are capable of generating all strings contained within the set S. Let it further be assumed that each of the two machines is allowed to generate n strings and let $P(S/FS_i)$ and $P(S/FS_j)$ denote respectively the probability that the sequence of strings generated by each machine is identical to the set S. The machine FS_i is regarded as being a more plausible generator of S than is FS_j if $P(S/FS_i) > P(S/FS_j)$.

The second criterion is the so called "Minimum Description Length (MDL)" criterion that was introduced relatively recently by J.Rissanen [Ref 4.4]. This idea has been taken up by Feder [Ref 4.3] who has suggested its use as a general criterion for statistical inference. This criterion can be stated in the following general terms: Given a finite set of observations (with, in the present context, a string being an observation), the optimum probabilistic model for generating the strings is that which minimizes the combined cost of the so-called self information (cost) required to represent the observations and the cost associated with the complexity of the model.

The following is an example of this approach in which the cost associated with the complexity of the model is taken to be fixed. Let it be assumed that a random variable X is capable of taking two values $X=\alpha$ and $X=\beta$ (these could be taken to represent the symbols that go to comprise a string). Let the probability of X taking the value α be p_α and the probability of it taking the value β be $p_\beta(=1-p_\alpha)$ and let it be further assumed that the system is memoryless and that the cost associated with the generation of α is $\log(p_\alpha)$ and the cost associated with the generation of β is $\log(p_\beta)$. Then the cost associated with the generation of a string containing the symbol α a total of n_α times and containing the symbol β a total of n_β times is

$$H=-(n_\alpha\log(p_\alpha)+n_\beta\log(p_\beta)). \qquad (4.1)$$

This cost (which can be related directly to the length of the ideal code needed to represent the observation) is minimized when:

$$p_\alpha=\frac{n_\alpha}{n} \text{ and } p_\beta=\frac{n_\beta}{n};$$

where $n=n_\alpha+n_\beta$, the number of symbols in the string. Notice that H, as given in Eq (4.1), is the cost associated with the generation of n symbols. The average cost per symbols is thus given as

$$\frac{H}{n}=-\frac{n_\alpha}{n}\log(p_\alpha)-\frac{n_\beta}{n}\log(p_\beta) \qquad (4.2)$$

and Eq (4.2) corresponds to the Shannon's Entropy function, in the case of a memoryless source and hence the criterion reduces to a minimum entropy criterion. M. Feder (Ref [4.3]) has proved this relationship between the minimum cost function and the entropy in more general terms and more general situations from that discussed in the above example. This minimum entropy concept could, therefore, be used as a criterion of plausibility.

J.Rissanen has shown (Ref [4.4]) that if the cost associated with the complexity of the model is neglected, then the Minimum Description Length criterion is equivalent to the Maximum Likelihood criterion discussed earlier. The relationship can be illustrated quite simply in terms of the example given above. The probability, P_γ, that the string γ contains the symbol α a total of n_α times and contains the symbol β a total of n_β times is:

$p_\gamma = p_\alpha^{n_\alpha} \cdot p_\beta^{n_\beta}$. Now, since the logarithm is a monotonically increasing function, it follows that the maximum of p_γ is the same as the maximum of

$$\max(p_\gamma) \Leftrightarrow \max\Big(\log(p_\gamma)\Big) = \max\Big(\log\big(p_\alpha^{n_\alpha} \cdot p_\beta^{n_\beta}\big)\Big)$$

$$= \max\Big(\log\big(p_\alpha^{n_\alpha}\big) + \log\big(p_\beta^{n_\beta}\big)\Big) = \max\Big(n_\alpha \log(p_\alpha) + n_\beta \log(p_\beta)\Big)$$

and since the maximum of a function is equivalent with the minimum of its negative it follows that the maximization of p_γ is equivalent to the minimization of $-n_\alpha \log(p_\alpha) - n_\beta \log(p_\beta)$. Thus, it follows that the criterion of Minimum Description Length is also a generalization of the Maximum Likelihood criterion.

4.2.3 Dynamic Programming Inference of Markov Networks.

In the previous subsection of this chapter, the need of a model capable of generating strings was explained and related criteria of plausibility were discussed.

In this section, the question of modelling is considered in detail and attention is focused on a particular set of finite state machines (Markov Networks).

An important contribution to the modelling, (or the obtaining of the appropriate Markov network) is contained in the paper by M.J.Thomason and E.Granum (Ref [4.5]). The essential relevant aspects contained in this paper will be reviewed before going on to present some new work on this topic.

The first constraint imposed on the Markov Network is that it should be capable of generating only finite-length strings. The desirability of having a machine capable of generating only finite-length strings is that it is easier to infer what the network should be, than would be in the case with a machine capable of generating infinite-long strings. The constraint that the machine

should be capable of generating only finite-length strings is equivalent to the "left-to-right move" or "no-loops" constraint imposed by Thomason et al in their paper.

Consider a particular FS machine (Markov network). At any one of a finite number of "instances", (which should not be confused with time instances or the positions in the space at which symbols appear at the output of the machine), the machine can produce one symbol drawn from a specific finite set of symbols, or it may fail to produce a symbol. This later aspect can be considered as a kind of deletion and will be referred to as "production" of a null. The fact that the machine at an instance can produce a symbol that is the same as one produced at some previous instance, and the fact that it may fail to produce a symbol at all, is at the heart of the problem that will be considered in the remainder of this chapter.

The approach adopted by Thomason et al in their paper to the question of the inference of a model from a set of strings is to consider the strings one by one and, for each string considered, to modify the FS machine (Markov Network) in order to maximize the probability that the string under consideration will be the output of the network in response to an initial trigger.

Now it can be proved by a counter example that this approach does not guarantee that the finally inferred network will be the maximum likelihood generator for the complete set of strings it is designed to model, with likelihood being taken as the sum of the probabilities that the machine will generate each string of the total set being considered. That is, if the set S contains the strings $S_1,...S_n$ and the probabilities that the finally inferred machine obtained using the Thomason's criterion of adjustment will generate the strings are $p_1,...p_n$ respectively, then the likelihood is taken to be $\sum_{i=1}^{n} p_i$. It is clear that this will be equal to or less than $\sum_{i=1}^{n} p_i'$ where p_i' is the maximum likelihood probability of generating a particular string in the model at the time it was considered and the machine inferred. This follows from the fact that the adjustment of the machine in respect of a particular string upsets the optimum setting arrived at in respect of the string presented to it immediately previously. The fact that the final

machine is not generally optimum can be proved, by the following counter example.

Consider the Markov network shown in Fig 4.3a. This represents the network after 19 substrings of elements have been presented to it. These substrings are the string $\beta\alpha\beta$ which is presented once, the string $\alpha\alpha\beta$ which is presented 13 times, the string $\alpha\alpha$ which is presented 4 times and the string $\beta\alpha\alpha$ which is also presented once. This pattern of events can be denoted by S=[#$\beta\alpha\beta$#, 13#$\alpha\alpha\beta$#, 4#$\alpha\alpha$#, #$\beta\alpha\alpha$#]. The corresponding number of times a link is traversed in the network is shown in Fig 4.3a and the associated probabilities are also shown in the figure from which it is calculated that the sum of the probabilities of generating the substrings is 0.637.

Now suppose a further (20^{th}) substring is added so that the set of samples becomes S=[#$\beta\alpha\beta$#, 13#$\alpha\alpha\beta$#, 5#$\alpha\alpha$#, #$\beta\alpha\alpha$#]. Now the sequential method of modelling suggests that the existing paths within the network established at the previous stage be kept and that they will be reinforced appropriately. If this procedure is adopted, then a Markov network as shown in Fig 4.3b is inferred. This network is the same as that shown in Fig 4.3a except that the trace of numbers (the line indicates the generation of successive symbols) is appropriately enhanced by 1. Assuming this Markov Model to be the appropriate model, it is clear that the probability that it will generate the 20^{th} string is 30/400=0.075, and the sum of probabilities of generation of the complete set of 20 substrings is 0.600. If, however, a Markov machine as shown in Fig 4.3c had been used, then the probability of generating the 20^{th} string would be 14/400= 0.035 but the sum of the probabilities of generating the complete set is now at the higher value of 0.645. The numerical details related to the above calculations are shown in Table 4.1.

The counter example given is not unrealistic and it is, in fact, indicative of what is likely to occur when using the criterion, set out in Thomason's paper, for maximizing the probability of generating the last substring. The overall important conclusion to be drawn is that Thomason's criterion may not work, in that it may fail to maximize the sum of the probabilities associated with the

generation of strings contained within the set S. In fairness to Thomason et al it should however be noted that they did not, in fact, claim that the criterion would minimize the sum of the probabilities referred to above. A new alternative approach to Markov network modelling will now be developed. The approach is based on the use of the Minimum Description Length criterion as a way of obtaining an appropriate network for the generation of the strings such as those contained within the set S. The idea of using the Minimum Description Length criterion is new.

In respect of modelling a set of n strings, let it be assumed that at an instance, the machine is capable of generating one of the symbols $a_1,\ldots,a_m$ and let it be also assumed that the costs associated with the generation of these symbols at this particular instance are $-\log(p(a_1)),\ldots,-\log(p(a_m))$, where $p(a_1),\ldots,p(a_m)$ are the probabilities that the symbols $a_1,\ldots,a_m$, respectively, will be generated. Further, let it be assumed that at the instance in question, the symbol a_i, $1\leq i\leq m$ occurs a total of n_i times ($0\leq n_i\leq n$), with respect to the complete set of n strings. Thus, the total cost associated with the generation of the symbols at the l^{th} instance is $-\sum_{i=1}^{m} n_i\cdot\log(p_i)=\Delta_l$, say. Clearly, the probability with which the symbols are generated changes from instance to instance. The total cost of generating the symbols over the total of L instances is the sum $M=\sum_{l=1}^{L}\Delta_l$, and the aim is the minimization of this cost.

From the discussion given earlier related to the Minimum Description Length criterion, it is clear that the minimization process described above corresponds to an application of this criterion to the generation of strings and, furthermore, when M is averaged over all strings and instances, this corresponds to the Entropy function familiar from the Information Theory.

The rather general description given above can be formulated in more precise mathematical terms as follows.

Given a finite set $S=\{X_1,X_2,\ldots,X_n\}$ of n strings and given that the $X_k{}^{th}$ string is $X_{k_1},X_{k_2},\ldots,X_{k_{l(k)}}$, where $X_{ki}\in T$, T is a finite set of symbols

and $l(k)$ is the length of the X_k^{th} string; then the Markov Network is sought that is "most likely" to have produced the sequence under the Minimum Description Length (MDL) criterion.

Let $\mathbb{I} = \{\mathbb{I}_1 \times \mathbb{I}_2 \times \ldots \times \mathbb{I}_n\}$ be an n dimensional discrete hyperspace, where $\mathbb{I}_k = [1,2,\ldots,l(k)]$ denotes the set of all positions in the $\mathbb{I}_k^{th}$ string, ranging from the first to the last position, $l(k)$, of this string, together with the zero position, before the string commences and the subscript k corresponds to the k^{th} string. This is illustrated in Fig 4.4 for the set of strings $[x_1x_2x_3, y_1y_2y_3y_4, z_1z_2z_3]$. The node point indicated by $\mathbf{I}=(2,4,1)$ in this figure corresponds to the generation of the three given substrings up to $\mathbf{I}$. It will be noted that each node within this n-dimensional discrete hyperspace corresponds to the generation of the set of strings up to some point.

Now, there are a large number of possible "trajectories" or "traces" that can be traversed in order to reach a given node point. For example, consider the two axes X and Y, shown in Fig. 4.5. These two axes show a two dimensional space and the point E=(1,2) denotes the instance at which the strings corresponding to these axes have been generated up to the point E (the first symbol for the X string and the second symbol for the Y string). In order to reach this point, it is possible to have traversed a number of different trajectories, two of which are shown in the figure. Physically, the first trace corresponds to a generation process in which nothing is generated in respect to the 2nd string during the first instance but a symbol α is generated in respect to the 1st string. Then at the second instance, the symbol α is produced with respect to the 2nd string and nothing is produced in respect to the 1st string. In the next (the third) instance, the symbol β is generated in respect to the 2nd string and nothing in respect to the 1st string. An alternative trace is that in which at the 1st instance the symbol α is generated with respect of both the 1st and 2nd strings, and in which thereafter the trajectory is the same as before. Each trajectory has a cost (description length) associated with it and the aim is to reach the end point at which all strings in S would have been generated and in so doing to have traversed the MDL trajectory.

In order to find out the minimum description length trajectory that has to be traversed in order to reach some point $\mathbf{l}$, it is necessary to know the minimum description length trajectories leading to all points that are allowed and are one step from the point $\mathbf{l}$, that is, all those allowed points $\mathbf{l}'$, from which it is possible to reach $\mathbf{l}$ in a single step. These are illustrated in Fig 4.4, from which it is seen that these allowed points lie on the corners of a hypercube of unit edge, with $\mathbf{l}$ being at one of the corners. Symbolically, these can be expressed as follows:

Let $\mathbb{J} = \{GF2\}^n - \{0\}$ be the set of all nonzero binary n-tuples, with this set representing the complete set of the one-step trajectories by which $\mathbf{l}$ is reached from $\mathbf{l}'$. Then the minimum description length at the point $\mathbf{l}$ is

$$M(\mathbf{l}) = \underset{\substack{\mathbf{j} \in \mathbb{J} \\ (\mathbf{l}-\mathbf{j})_i \geq 0 \ \forall i}}{\text{MIN}} \left\{ M(\mathbf{l}-\mathbf{j}) + \Delta(\mathbf{l}-\mathbf{j}, \mathbf{l}) \right\} \tag{4.3}$$

where $\mathbf{j}$ is such as to restrict the choice at the edge of the hyperspace and to prevent the trajectory from leaving the hypercube and $\Delta(\mathbf{l}-\mathbf{j},\mathbf{l})$ is the minimum cost (minimum increment in description length) associated with the last step.

In order to determine $M(\mathbf{l}-\mathbf{j},\mathbf{l})$ the complete set of one step trajectories is examined and the entropy associated with each one step trajectory is evaluated, with the entropy being defined as before. An example which serves to illustrate this is shown in Fig 4.4 on which the entropies associated with the seven one-step trajectories are shown.

The minimization of $M(\mathbf{l})$ is then carried out by evaluating Eq. 4.3 above for each of the one step trajectories leading to the $\mathbf{l}$ point. This process is then repeated until the ends of all strings in S are reached i.e. until the point marked ★ of the hyperspace shown in Fig 4.4 is reached. It will be appreciated that the process is a dynamic programming process extended to the n-dimensions and its similarity to the Viterbi process of decoding should be appreciated.

In order to clarify the arguments set out above consider the following example, the main features of which are given in the Fig 4.6. Let l be the point to be reached with the numbers (9,5,7,3) denoting the length of the strings W,X,Y,Z up to that point. The point l in the 4-dimensional space being considered can be reached from 15 different previous states l'. Each one-step trajectory can be indicated by means of a binary 4-tuple j as shown in the Figure, and the previous stages corresponding to the one-step moves to l are shown in the column headed l'. The set of symbols generated are indicated in the column headed A, with λ indicating a null response. The corresponding entropies are given in the column headed H. The fractions in this column correspond to the relative frequency with which the null output and the symbols a,b, and c are generated when traversing the corresponding one-step trajectory. For example, the 4-tuple (non-binary) a=(λ,λ,λ,c) indicates that with the particular associated one-step trajectory, the null occurs 3 out of 4 times and the symbol c 1 out of 4 times. The one step trajectory to be chosen for the final move to l has to be determined by obtaining from each of the 15 possible one-step trajectories the sum of the costs (entropy or minimum description length) occurring up to the point l' and the cost shown in the corresponding row of the column H.

It is reemphasized at this point that the aim (which has been achieved) of the approach set out above is to develop a technique which can be used to find a Minimum Description Length (highly plausible) machine for the generation of the set of strings S.

It will be noted from the description given above that the dimensionality of the hyperspace used in arriving at the Minimum Description Length machine is equal to the number of strings contained within the set S irrespective of whether or not any or many of them are identical. Now, clearly, in most pattern recognition situations it is highly likely and, indeed, from the point of view of easing the recognition process, it is desirable, that many of the strings within a set such as S will be identical. In this situation, it will clearly be advantageous in terms of the simplicity of finding the Minimum Description Length trajectory

if the dimensionality of the hyperspace could be reduced. An immediate consequence of a theorem which will now be proved is that the axes associated with identical strings can be merged into a single axis, and the Minimum Description Length trajectory in the reduced dimensionality hyperspace is still optimum, that is, it has the same cost (Description Length) associated with it as the cost associated with the optimal trajectory in the original hyperspace.

Theorem: If within the n-dimensional hyperspace with axes corresponding to strings a number of the axes (strings) are identical, then the Minimum Description Length trajectory lies in the hyperplane which is the set of points satisfying the equation X=Y

The meanings of the terms contained in the theorem are illustrated in Fig 4.7

Proof:

The proof will be given for the merging of two axes that are themselves already merged, that is, each has been formed from the merging of a number of identical axes. The proof thus applies to the merging of two or more identical axes. An axis that is the result of a merging will be called a "merged" axis.

Consider two axes X and Y, which, as stated above, are each the result of a merging process, and let W_X and W_Y be the weights, that is, the number of axes that have been merged respectively into the axes X and Y and let the number of axes after merging including the merged axes X and Y be n^* ($\leq N$). Let $\mathbb{T}$ be the set of all trajectories that can be traversed in the n^*-dimensional hyperspace and let $\mathbb{T}_B$ be the subset of these trajectories which lie on the (n^*-1)-dimensional hyperplane which is the bisector of the X and Y axis. The bisector of X and Y is the set of points which satisfy the equation X=Y.

We wish to prove that the optimum trajectory is contained within the set $\mathbb{T}_B$. Equivalently, it is to be proved that all traces that are not contained in $\mathbb{T}_B$ but which lead to the end point are not optimum (are not Minimum Description Length traces).

Let $\mathbb{T}_{\hat{B}}$ denote the set $\mathbb{T}-\mathbb{T}_B$ of traces that are not contained in $\mathbb{T}_B$.

Consider any trace $\mathcal{T}$ in $\mathbb{T}\hat{B}$. If it can be shown that, for any chosen trace, $\mathcal{T}$, in $\mathbb{T}\hat{B}$, there exist a trace, $\mathcal{T}^*$, in $\mathbb{T}B$ which is better, then this proves that the optimum trajectory is contained in $\mathbb{T}B$. It will be proved that for any trajectory $\mathcal{T}$ in $\mathbb{T}\hat{B}$, there exist within $\mathbb{T}B$ two trajectories, $\mathcal{T}X$ and $\mathcal{T}Y$, which are such that at least one of them is better than the trace $\mathcal{T}$. This will thus prove the theorem.

Expressed symbolically, we have:

$$\mathcal{T}opt \in \mathbb{T}_B \Longleftarrow \forall\, \mathcal{T} \in \mathbb{T}-\mathbb{T}B \neq \mathcal{T}opt \Longleftarrow \forall\, \mathcal{T} \in \mathbb{T}-\mathbb{T}B\ \exists\ \mathcal{T}^* \in \mathbb{T}_B$$

such that

$$M(\mathcal{T}) \geq M(\mathcal{T}^*) \quad \Longleftarrow \quad \forall\, \mathcal{T} \in \mathbb{T}-\mathbb{T}B\ \exists\ \mathcal{T}X, \mathcal{T}Y \in \mathbb{T}_B \text{ such that:}$$

$$M(\mathcal{T}) \geq \frac{WX \cdot M(\mathcal{T}X) + WY \cdot M(\mathcal{T}Y)}{WX + WY} \tag{4.4}$$

(the symbol " $\Longleftarrow$ " stands for " is implied by ")

It follows that if the condition (4.4) is satisfied, then at least one of the $M(\mathcal{T}X)$, $M(\mathcal{T}Y)$ is less than or equal to $M(\mathcal{T})$.

Consider now the determination of the two traces $\mathcal{T}X$ and $\mathcal{T}Y$. These traces are, by definition, contained in the $\mathbb{T}B$, that is they lie in the hyperplane whose coordinates satisfy the equation $X=Y$ (see Fig 4.7). Any point in the space is defined by its coordinates and hence the traces themselves are defined by a sequence of such coordinates. In fact, those contained within the hyperplane have general coordinates $(X, Y(=X), z_i)$, $i=3,4,\ldots,n^*$, where n^* is the dimensionality of the hyperspace.

Now, consider a trace $\mathcal{T}$ which does not lie on the bisecting hyperplane and let (X_t, Y_t, z_{it}) $i=3,4,\ldots,n^*$, be the coordinates of one point, the point I_t, of this trace. Consider two projections of the point I_t with the one projection being parallel to the Y axis and the other being parallel to the X axis. This is illustrated in the Fig 4.7. Where the projection process is repeated for each point of the trace $\mathcal{T}$, then the result is two traces in the hyperplane B. This is illustrated in Fig. 4.8 which shows the two traces that are generated by the

projections from the trace $\mathcal{T}$. The whole process can be expressed symbolically as:

$$P_X(\mathcal{T}X) = P_X(\mathcal{T}) \qquad P_Y(\mathcal{T}X) = P_X(\mathcal{T})$$
$$P_X(\mathcal{T}Y) = P_Y(\mathcal{T}) \qquad P_Y(\mathcal{T}Y) = P_Y(\mathcal{T})$$
$$P_Z(\mathcal{T}X) = P_Z(\mathcal{T}Y) = P_Z(\mathcal{T}) \qquad \forall\, Z \notin \{X,Y\}$$

where $P_X(\mathcal{T}X)$ is the X coordinate of the $\mathcal{T}X$ traces, likewise $P_Y(\mathcal{T})$ is the Y coordinate of the $\mathcal{T}$ trace etc.

Now, it remains to be shown that the two traces so defined satisfy the relation (4.4) above. As a consequence of the additive property of $M(\mathcal{T})$ (remember that costs were defined to be additive), it is sufficient, in order to prove (4.4), that this condition holds for any one step contained within the trace.

In order to make the method of proof clear consider again the Fig 4.6. We wish to express the cost (the Entropy H) shown in the last column of the Fig 4.6 as the Entropy of a 4-element vector, whose elements correspond to the probabilities associated with λ (null) and a,b,c, with the first element corresponding to the probability associated with λ, the second element corresponding to the probability associated with a, the third element corresponding to the probability associated with b and the fourth element corresponding to the probability associated with c. Let the row vector [0,1,0,0] represent the element a, the vector [0,0,1,0] represent the element b, the vector [0,0,0,1] represent the element c, and the vector [1,0,0,0] represent the null element. Then, the vector V representing the argument of H is formed by taking the weighted sum of the vectors I_W, I_X, I_Y, I_Z representing the contents of the W,X,Y,Z axes respectively, (that is the symbols a, a, b and c), with the weights being given by the elements contained within the first column in Fig. 4.6.

Consider the 15^{th} row of Fig 4.6. The following fixed vectors corresponding to the elements a,a,b,c are:

[0,1,0,0], [0,1,0,0,], [0,0,1,0], [0,0,0,1]

and these are then weighted respectively with the elements taken from the first column. Thus, the sum is

$$1\cdot[0,1,0,0]+1\cdot[0,1,0,0,]+0\cdot[0,0,1,0]+1\cdot\ [0,0,0,1]$$

which on element-by-element addition yields [0 2 0 1]. Now, this vector tells us which symbols are present (in this case 2 a's and 1 c) but it only provides us with indirect information about the number of nulls. The number of nulls is obtained by subtracting the sum 2+1 from 4, in order to find out that a single null is involved. Thus, the entropy of the probability associated with the step (λ,a,b,c) involved in going to the final point by the vector j in column 1 of Fig. 4.6 is H(1/4,2/4,0,1/4)=1.5.

In general, the entropy vector $\mathbb{V}$ of the one step trajectory $\mathfrak{J}$ can be expressed as:

$$\mathbb{V} = \sum_{i=1}^{n^*} W_i \mathfrak{J}_{X_i} \dot{\mathbf{j}}_{X_i} + \left(N - \sum_{i=1}^{n^*} W_i \dot{\mathbf{j}}_{X_i}\right)[1\ 0\ \ldots\ 0]$$

$$= \sum_{i=1}^{n^*} W_i P_{X_i}(\mathfrak{J}_{X_i}) + \left(N - \sum_{i=1}^{n^*} W_i \dot{\mathbf{j}}_{X_i}\right)[1\ 0\ \ldots\ 0]$$

where $P_{X_i}(\mathfrak{J})$ is the projection of $\mathfrak{J}$ on the X_i axis, N is the total number of strings, n^* is the number of strings (axes) after merging (this is the dimensionality of the hyperspace) and Wi is the weight of the axis Xi.

The reason behind this somewhat cumbersome approach is that it provides a method of finding an entropy vector for each one-step trajectory. For example for the point (9,5,7,3) considered above (see Fig 4.6), the one step trajectory leading to (9,5,7,3) via (8,4,7,2) has an entropy vector (1/4,2/4,0,1/4).

Now, if the weights of X and Y (the number of respective identical strings combined) are W_X and W_Y respectively and $\mathbb{V}_X$ and $\mathbb{V}_Y$ are the entropy vectors resulting from the projections on the Y and X directions respectively from the trace that lies outside the bisector plane, then it can be shown that the entropy $\mathbb{V}t$ of the one step (the t^{th} step) of the trajectory lying outside the bisector hyperplane is related to $\mathbb{V}_X$ and $\mathbb{V}_y$ by the expression:

$$\mathbb{V}_t = \frac{W_X \cdot \mathbb{V}_{X_t} + W_Y \cdot \mathbb{V}_{Y_t}}{W_X + W_Y} \tag{4.5}$$

It will now be shown that Eq. (4.5) holds and, further, that, by convexity, this immediately implies that the one step cost $\Delta(\mathcal{T})$ associated with the trace outside the bisector hyperplane and the costs $\Delta(\mathcal{T}_X)$ and $\Delta(\mathcal{T}_y)$ associated with the one-step projections of this onto the bisector hyperplane are related as follows:

$$\Delta(\mathcal{T})_t \geq \frac{W_X \cdot \Delta(\mathcal{T}_X)_t + W_Y \cdot \Delta(\mathcal{T}_Y)_t}{W_X + W_Y} \tag{4.6}$$

from which it can be shown that the cost associated with one or both of the trajectories projected onto the hyperplane is less than, or equal to, the cost associated with the trajectory outside the hyperplane.

Eq (4.5) will now be proved and it will be then argued that (4.6) follows by virtue of the convexity of the entropy function.

Let $\mathbb{V}$ denote,as before, the entropy vector associated with the one step trajectory outside the bisector plane. This trajectory is in a n^*-dimensional space which is formed by combining a number of identical strings to obtain a corresponding weighted axis (or a number of different weighted axes). Then from before we have:

$$\mathbb{V} = W_X P_X(\mathcal{T}) + W_Y P_Y(\mathcal{T}) + \sum_{Z_i} W_{Z_i} P_{Z_i}(\mathcal{T})$$

where $\sum_{Z_i} W_{Z_i} P_{Z_i}(\mathcal{T})$ with the sum of the projections of the one-step trajectory to the Z_i axis

Likewise, the entropy vector, $\mathbb{V}_X$, of the projection of the trace $\mathcal{T}$ in the Y direction is

$$\mathbb{V}_X = W_X P_X(\mathcal{T}_X) + W_Y P_Y(\mathcal{T}_X) + \sum_{Z_i} W_{Z_i} P_{Z_i}(\mathcal{T}_X) =$$

$$= (W_X + W_Y)P_X(\mathcal{T}) + \sum_{Z_i} W_{Z_i} P_{Z_i}(\mathcal{T})$$

and similarly the entropy vector V_Y, of the projection in the X direction is

$$V_Y = W_X P_X(\mathcal{T}_Y) + W_Y P_Y(\mathcal{T}_Y) + \sum_{Z_i} W_{Z_i} P_{Z_i}(\mathcal{T}_Y)$$

$$= (W_X + W_Y)P_Y(\mathcal{T}) + \sum_{Z_i} W_{Z_i} P_{Z_i}(\mathcal{T})$$

Therefore, it follows that: $$V = \frac{W_X \cdot V_X + W_Y \cdot V_Y}{W_X + W_Y} \tag{4.7}$$

This relation holds for all elements in the entropy vector including the first element of it which denotes the number of nulls involved. The proof given applies directly to elements corresponding to the non null symbols. A similar proof can, however, be provided for the case of the nulls.

From the relation (4.5) it is clear that V_t lies between V_{Xt} and V_{yt}. Also $\Delta(V_t)$ is, by definition, the entropy of the vector V_t. By virtue of the convexity of the entropy function it follows at once that

$$\Delta(\mathcal{T})_t \geq \frac{W_X \cdot \Delta(\mathcal{T}_X)_t + W_Y \cdot \Delta(\mathcal{T}_Y)_t}{W_X + W_Y} \tag{4.8}$$

which proves immediately that the entropy of the trace outside the bisector hyperplane is equal to or greater than the entropy of at least one of its projections onto the hyperplane. These convexity arguments are illustrated diagrammatically in Fig. 4.9 from which it should be clear that (4.8) holds. □

The theorem just proved is of considerable importance in attempts to obtain a suitable Markov Network for use as a model for a set of strings. This importance stems from the fact that by merging identical axes (strings) the computational problem involved in obtaining the network is significantly reduced. In fact, the complexity of the computational problem is reduced exponentially as a function of the number of axes that are merged. A further practical importance of the theorem becomes clear when it is appreciated that

within a well ordered set of patterns which go to represent a class it is likely, and indeed it is desirable, that a number of members of the set will be the same.

On a broader philosophical front, it may be argued that the idea of merging identical strings (axes) and using a weighted resulted axis is intuitively sensible and that a sensibly formulated approach to the costing must reflect this intuitive view. The fact that the theorem and its proof lead to results that agree with the intuitive reasoning is an indication of the validity of the approach adopted.

A simple, yet instructive example will now be given to illustrate the arguments and the concepts presented above. In particular, the example will illustrate the concept of the merging of axes.

EXAMPLE:

Let the set of three strings, from which the Markov Network is to be inferred, be $[\gamma,\gamma\alpha,\gamma\alpha\alpha]$, together with a fourth string, $\gamma\alpha$, which is identical with the second string.

Consider first the set of the three strings $[\gamma,\gamma\alpha,\gamma\alpha\alpha]$. It follows that they are all different and no merging can be contemplated. There are, in this case, two simple machines which are readily synthesized, which generate the three strings. These are shown in Figs 4.10 a and b. The minimum description length (the entropy) associated with the two machines is seen to be the same for each machine and each one is, in fact, an MDL generator.

Now, consider the inclusion of the fourth string and consider the total set of the four sequences: $[\gamma,\gamma\alpha,\gamma\alpha\alpha,\gamma\alpha]$. In this case, there are two machines, as shown in Figs. 4.10 c and d, which are capable of generating these strings. It will be observed for Fig. 4.10c that there are two different trajectories for the generation of the second and the fourth string respectively and that these are symmetric. In the case of 4.10d, however, only one path is followed through the machine for the generation of either the second or the fourth string. Examination of the entropies associated with the two machines shows that the

machine of Fig 4.10c has an entropy cost of 8, whereas the machine shown in Fig 4.10d has an entropy cost of 6.39.

An alternative approach to synthesizing a machine for the generation of the 4 strings could be the merging of the second and fourth string, since they are identical, into a weighted string of weight 2, and then, to deal with the synthesis of a machine generating the three strings $[\gamma, 2\gamma\alpha, \gamma\alpha\alpha]$. The theorem tells us at once that no extra cost is incurred when considering this reduced dimensional situation, and that the optimum, when found, will have the same cost associated with it as would be associated with a machine which operates and treats the four sequences as being essentially different.
Examination of the three strings above immediately suggests a machine identical to that of the Fig 4.10d. It is important to note that the alternative shown in Fig. 4.10c is not a possibility when considering the merged sequences and the theorem ensures that the omission of machines such as that shown in Fig 4.10c does not result in the removal from consideration of any machine that is more cost effective than the possibilities shown in Fig 4.10d. It is this reduction in the number of possible different machines that need be considered that is in the heart of the computational gain referred to earlier.

4.3 CONCLUSIONS.

In the first part of this chapter, the concept of the Handprinted character was dealt with. Attention was drawn to the lack of an ideal pattern as representative of a pattern class, and the weakness of the template matching approach to pattern classification was discussed.

It was argued that a character (pattern class) must be considered as *a global structure of locally identifiable elements* with the human being acting as the ultimate judge of the accuracy of any OCR system.

The need for a model to describe the various pattern classes was explained and the idea of testing a model by *pattern synthesis* was put forward. This is

an area that is worthy of detailed consideration.

Although no attempt was made to develop actual models for representing character classes, attention was drawn to the fact that an acceptable model should be able to express the variations within a pattern class resulting from the stochastic nature of the characteristics of the different elements constituting a character. It was also argued that the model should be able to reflect the interrelation between the various characteristics. It was then explained in a chapter that a fully representative model of the characteristics is likely to be too complicated for analysis. As a consequence of this difficulty, attention was shifted to the search for a less succinct model which was capable of being analysed.

It was argued that the Markov Model was one such model that was worthy of investigation. The question of the criterion of inference was considered at some length and a Minimum Description Length (MDL) criterion was adopted. The choice of the cost function (that is the Description Length) was justified by Information Theoretic reasoning. The idea of using a Multidimensional Dynamic Programming approach was put forward as a means of finding the minimum description length. In this approach, the strings that are being modelled are represented as the axes of multidimensional space and the minimum description length has to be derived in this space. A new theorem is stated and proved by which it is shown that it is possible to merge identical axes without incurring any loss in the optimality in respect to the MDL criterion. The theorem means that the dimension of the space can be reduced and the computational burden involved in determining the minimum description length and the associated Markov model can, thereby, be significantly reduced.

Time did not permit a thorough application of the ideas developed in this Chapter to the actual pattern recognition pproblem. However, some initial work was done which indicated the value of the techniques.

REFERENCES

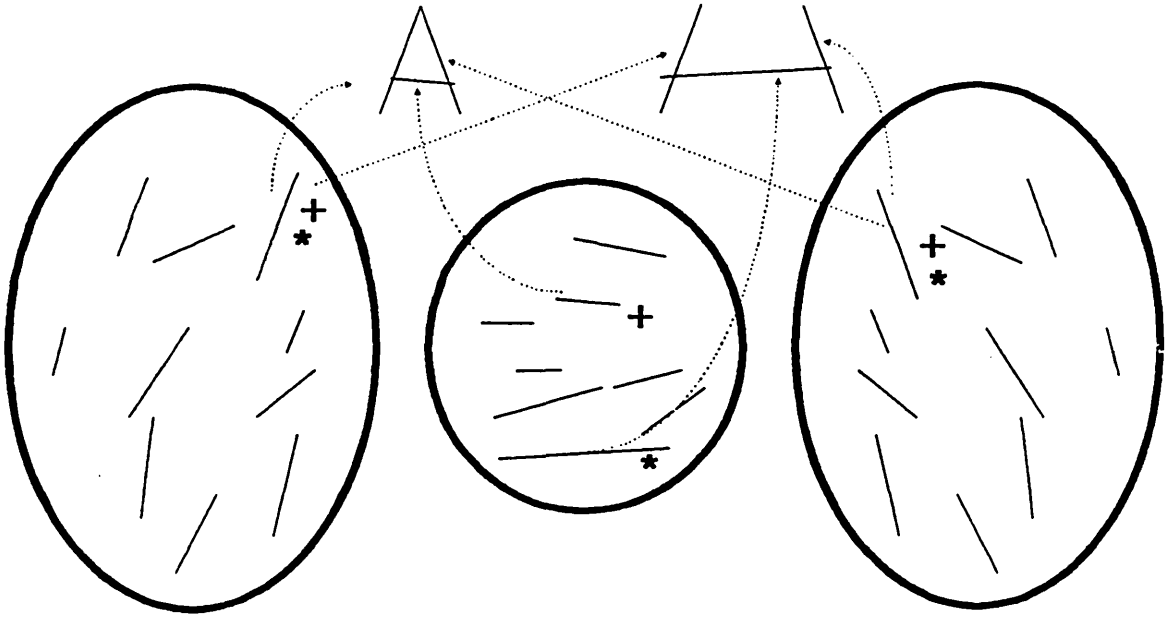
[4.1] M. BRANDY: "Toward a Computational Theory of Early Visual Processing in Reading", Visible Language, Vol XV, No 2 (1981) pp. 183-214.

[4.2] K.S.FU: "Syntactic Pattern Recognition and Applications", Englewood Cliffs, NJ: Prentice-Hall, 1982

[4.3] M. FEDER: "Maximum Entropy as a Special Case of the Minimum Description Length Criterion", IEEE Trans. on Information Theory, Vol. IT-32, No 6 (1986), pp 847-849

[4.4] J. RISSANEN: "A universal prior for integers and estimation by MDL", Ann. Statist. Vol 11, No. 2, (1983), pp. 416-432

[4.5] M.G.THOMASON, E.GRANUM: "Dynamic Programming Inference of Markov Networks from Finite Sets of Sample Strings", IEEE Trans. on Pattern Analysis ans Machine Intelligence, Vol. PAMI-8, No 4 (1986), pp 491-501



Strokes marked with a "+", when combined , form a character A, whereas strokes marked with a "*" do not, though they are elements of some characther A.

Fig 4.1

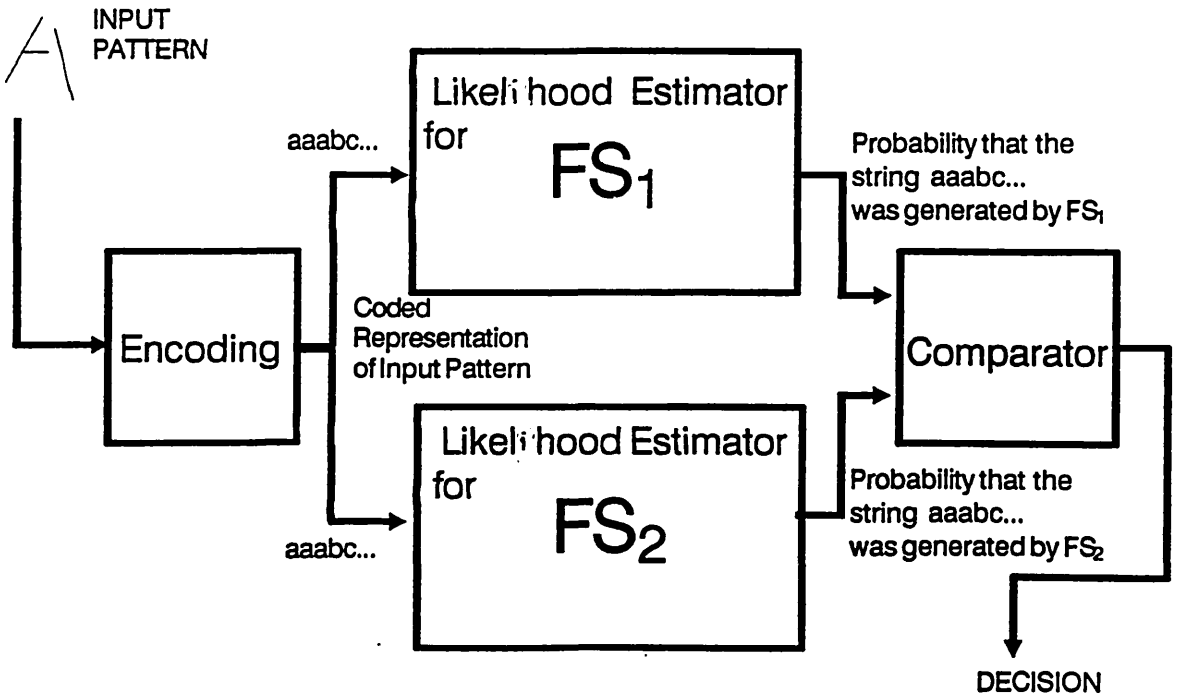


Fig 4.2

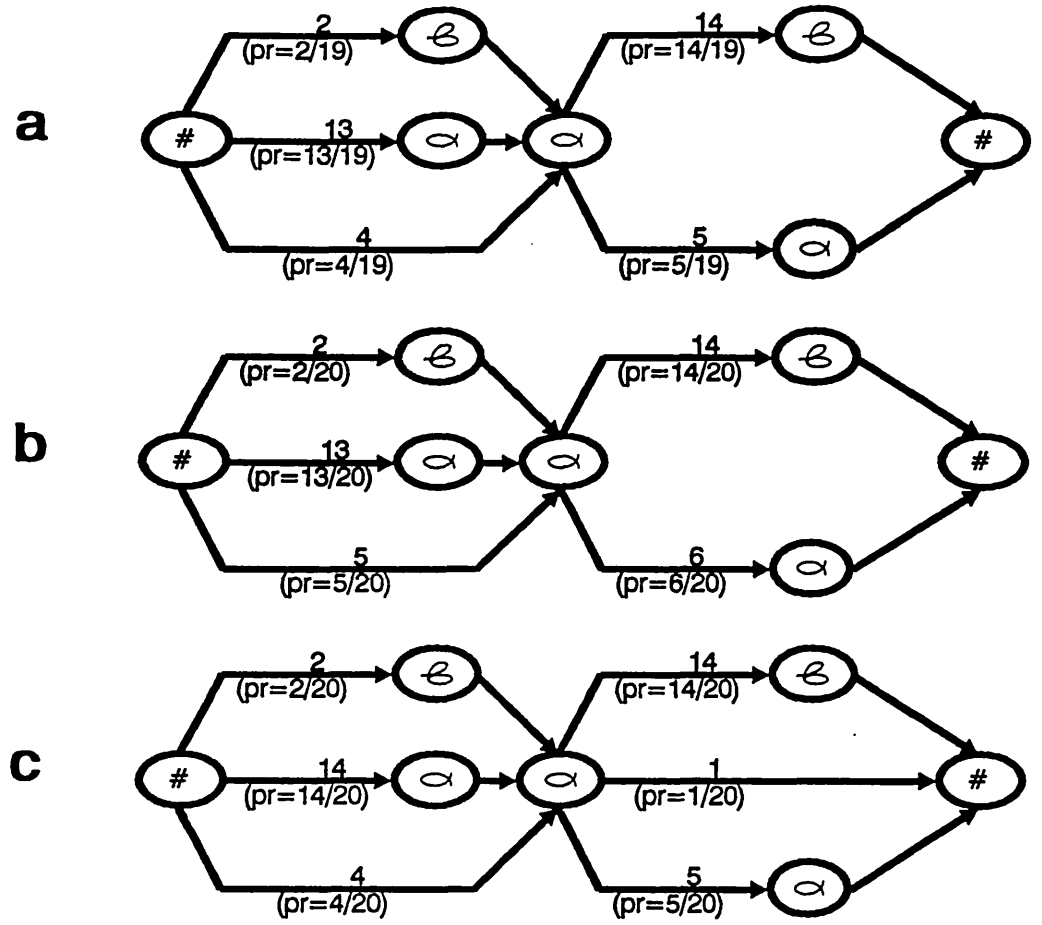


Fig 4.3 Sequential Inference of Markov Network

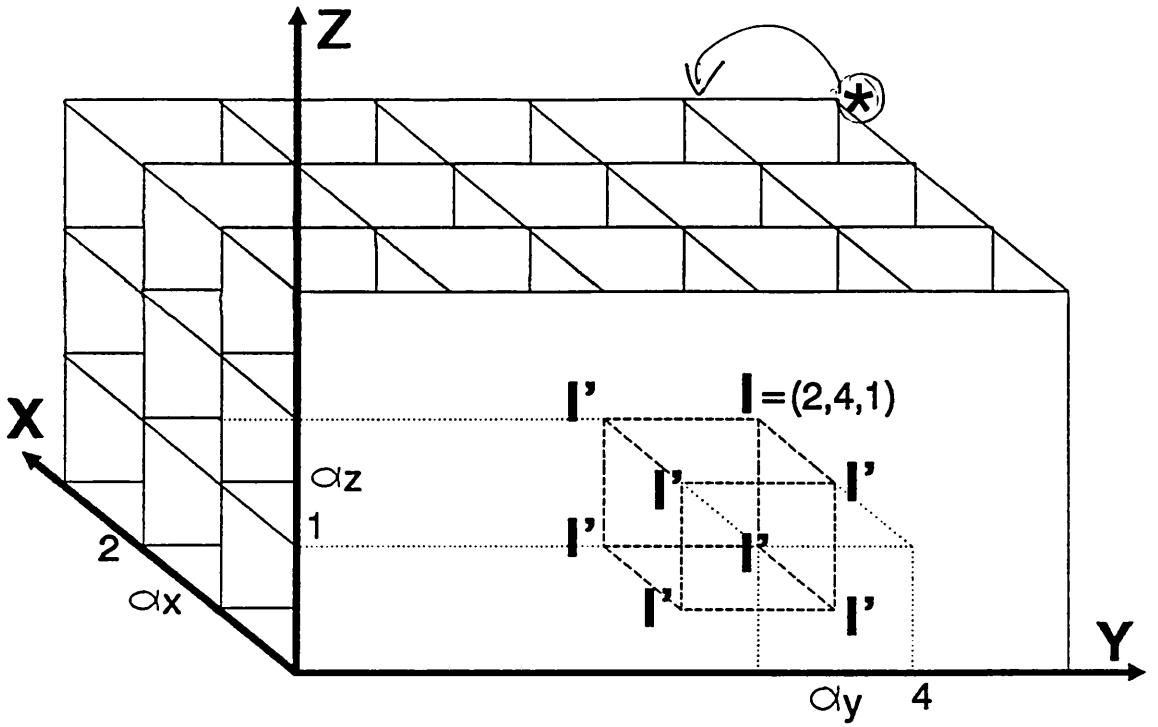
TABLE 4.1 : Calculations of the Probability of Generation of the Sample

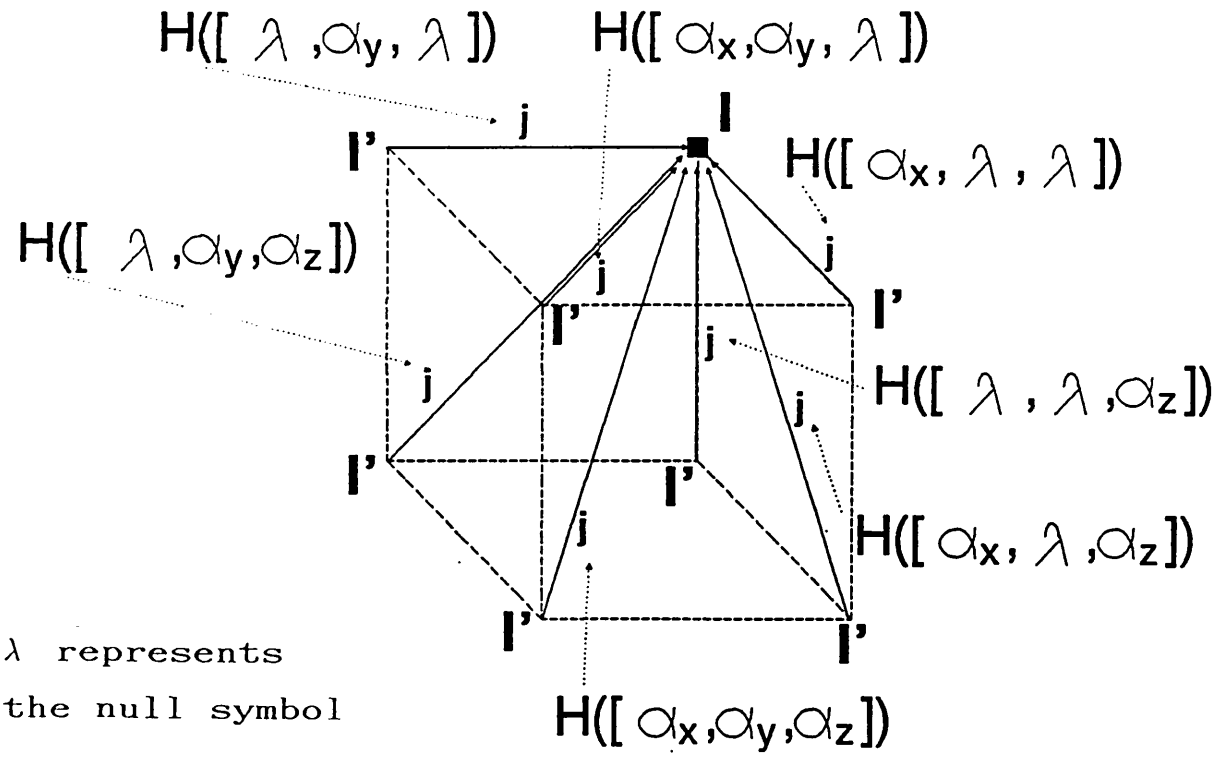
Branch frequences in network:

PREVIOUS		SEQUENTIAL		COUNTEREXAMPLE	
2	14	2	14	2	14
13	0	13	0	14	1
4	5	5	6	4	5

Number of strings in training sample:

	PREVIOUS		SEQUENTIAL		COUNTEREXAMPLE	
	$\in S_I$	$\notin S_I$	$\in S_I$	$\notin S_I$	$\in S_I$	$\notin S_I$
$\beta\alpha\beta$	28		28		28	
$\beta\alpha\hat{}$		0		0		2
$\beta\alpha\alpha$		10		12		10
$\alpha\alpha\beta$	182		182		196	
$\alpha\alpha\hat{}$	0		0		14	
$\alpha\alpha\alpha$		65		78		70
$\hat{}\alpha\beta$		56		70		56
$\hat{}\alpha\hat{}$		0		0		4
$\hat{}\alpha\alpha$	20		30		20	
sums	230	131	240	160	258	142
$\text{Prob}(S_I)$	0.637		0.600		0.645	





λ represents
the null symbol

Fig 4.4

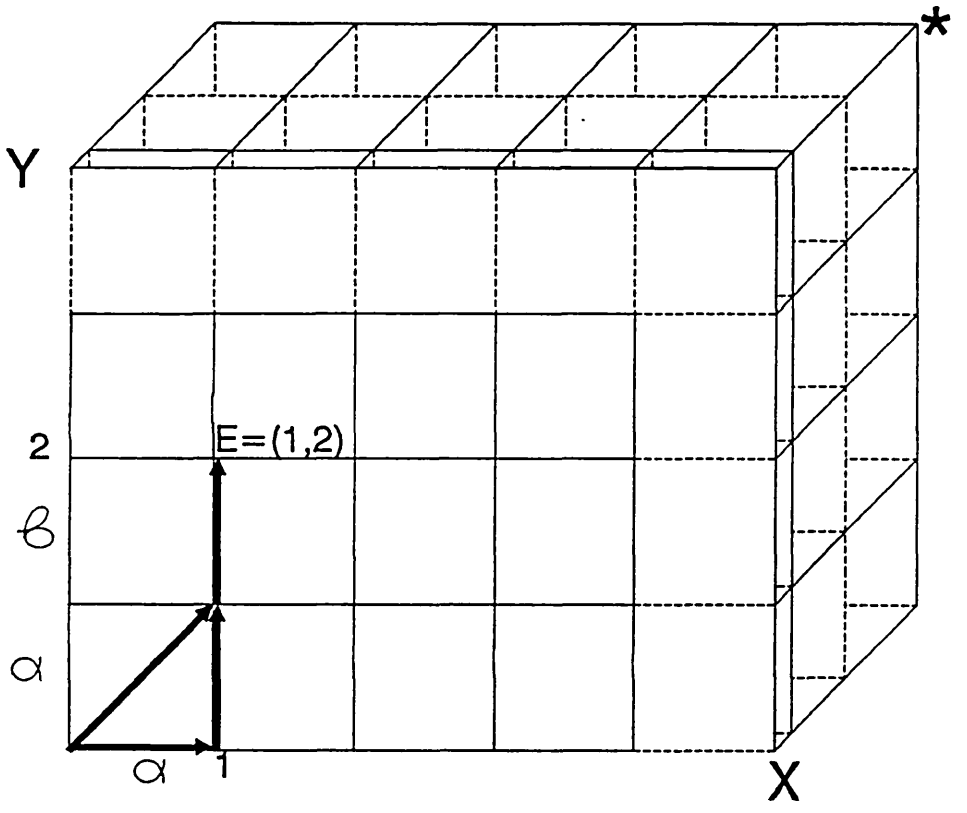


Fig 4.5

Let $\mathbf{l}$ be the point (9,5,7,3)

Let the corresponding positions of the 4 strings (axis) W,X,Y,Z contain:

string W .....a....

string X ....a.....

string Y .....b.....

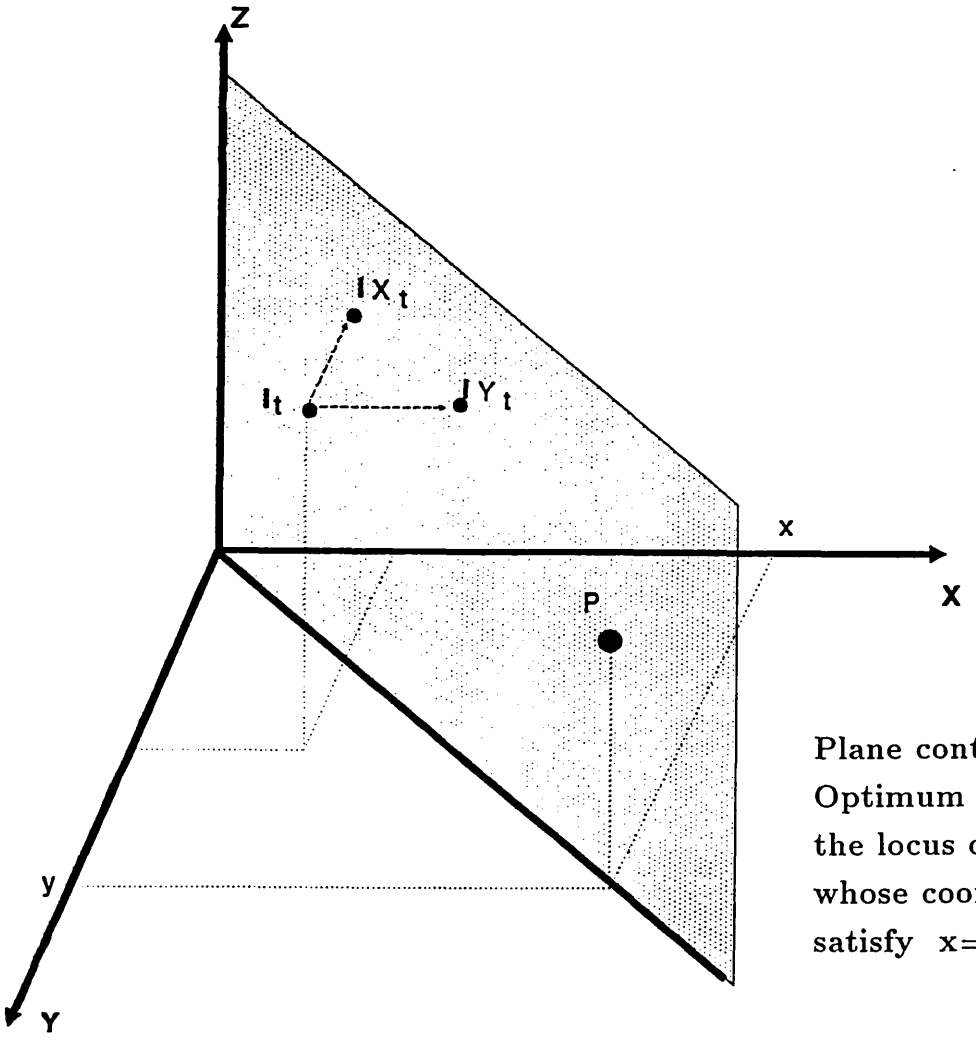
string Z ..c.....

in other words contain the 4-tuple [a,a,b,c].

j	$\mathbf{l}'$	set A	H(V)
1st	[0,0,0,1]	(9,5,7,2)	[λ,λ,λ,c]	$H(3/4,1/4)=0.8113$
2nd	[0,0,1,0]	(9,5,6,3)	[λ,λ,b,λ]	$H(3/4,1/4)=0.8113$
3rd	[0,0,1,1]	(9,5,6,2)	[λ,λ,b,c]	$H(2/4,1/4,1/4)=1.5$
4th	[0,1,0,0]	(9,4,7,3)	[λ,a,λ,λ]	$H(3/4,1/4)=0.8113$
5th	[0,1,0,1]	(9,4,7,2)	[λ,a,λ,c]	$H(2/4,1/4,1/4)=1.5$
6th	[0,1,1,0]	(9,4,6,3)	[λ,a,b,λ]	$H(2/4,1/4,1/4)=1.5$
7th	[0,1,1,1]	(9,4,6,2)	[λ,a,b,c]	$H(1/4,1/4,1/4,1/4)=2$
8th	[1,0,0,0]	(8,5,7,3)	[a,λ,λ,λ]	$H(1/4,3/4)=0.8113$
9th	[1,0,0,1]	(8,5,7,2)	[a,λ,λ,c]	$H(1/4,2/4,1/4)=1.5$
10th	[1,0,1,0]	(8,5,6,3)	[a,λ,b,λ]	$H(1/4,2/4,1/4)=1.5$
11th	[1,0,1,1]	(8,5,6,2)	[a,λ,b,c]	$H(1/4,1/4,1/4,1/4)=2$
12th	[1,1,0,0]	(8,4,7,3)	[a,a,λ,λ]	$H(2/4,2/4)=2$
13th	[1,1,0,1]	(8,4,7,2)	[a,a,λ,c]	$H(2/4,1/4,1/4)=1.5$
14th	[1,1,1,0]	(8,4,6,3)	[a,a,b,λ]	$H(2/4,1/4,1/4)=1.5$
15th	[1,1,1,1]	(8,4,6,2)	[a,a,b,c]	$H(2/4,1/4,1/4)=1.5$

Fig 4.6

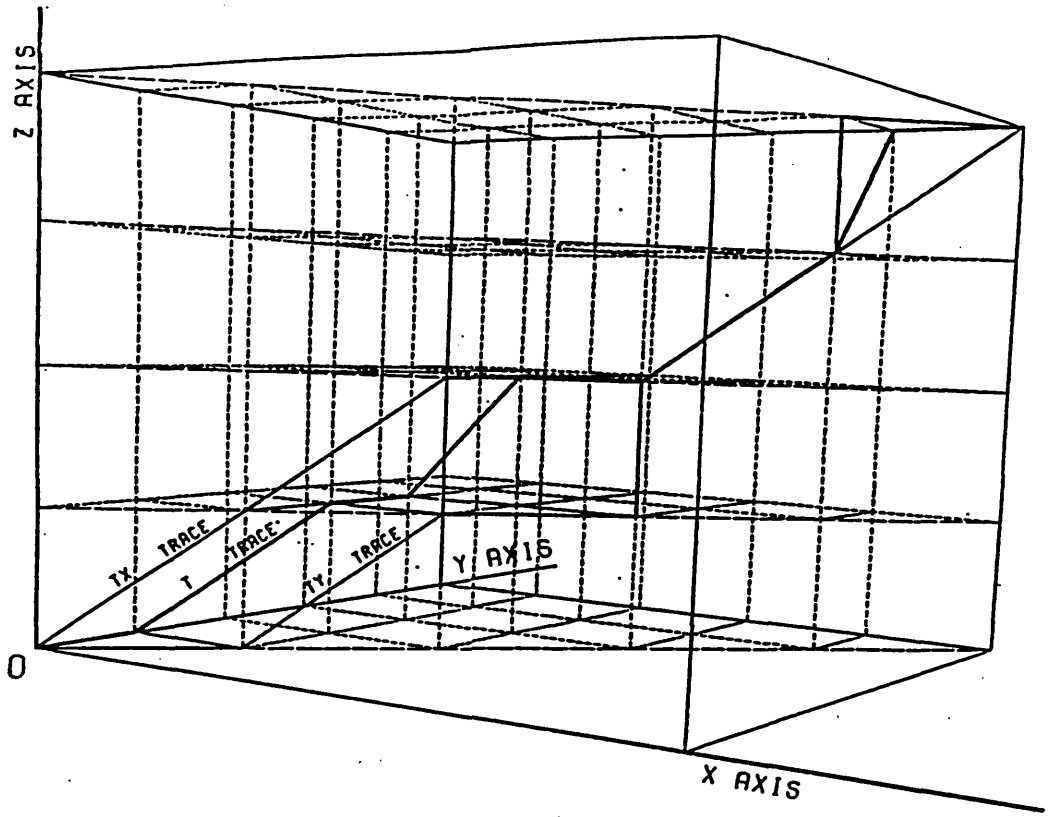
Example of the Evaluation of the Cost $\Delta(\mathbf{l}',\mathbf{l})$



Plane containing the Optimum Trace, that is, the locus of the points whose coordinates satisfy x= y

Fig 4.7

Projections of a point of the space onto the bisector Hyperplane



$\mathcal{T}$	$\mathcal{T}_X$	$\mathcal{T}_Y$
(0,0,0)	(0,0,0)	(0,0,0)
(0,1,0)	(0,0,0)	(1,1,0)
(1,2,1)	(1,1,1)	(2,2,1)
(1,3,1)	(1,1,1)	(3,3,1)
(2,3,2)	(2,2,2)	(3,3,2)
(3,3,2)	(3,3,2)	(3,3,2)
(4,4,3)	(4,4,3)	(4,4,3)
(4,5,4)	(4,4,4)	(5,5,4)
(5,5,4)	(5,5,4)	(5,5,4)

FIG 4.8

THE NON OPTIMUM TRACE $\mathcal{T}$ AND ITS PROJECTIONS $\mathcal{T}_X$ AND $\mathcal{T}_Y$ ONTO THE BISECTOR HYPERPLANE

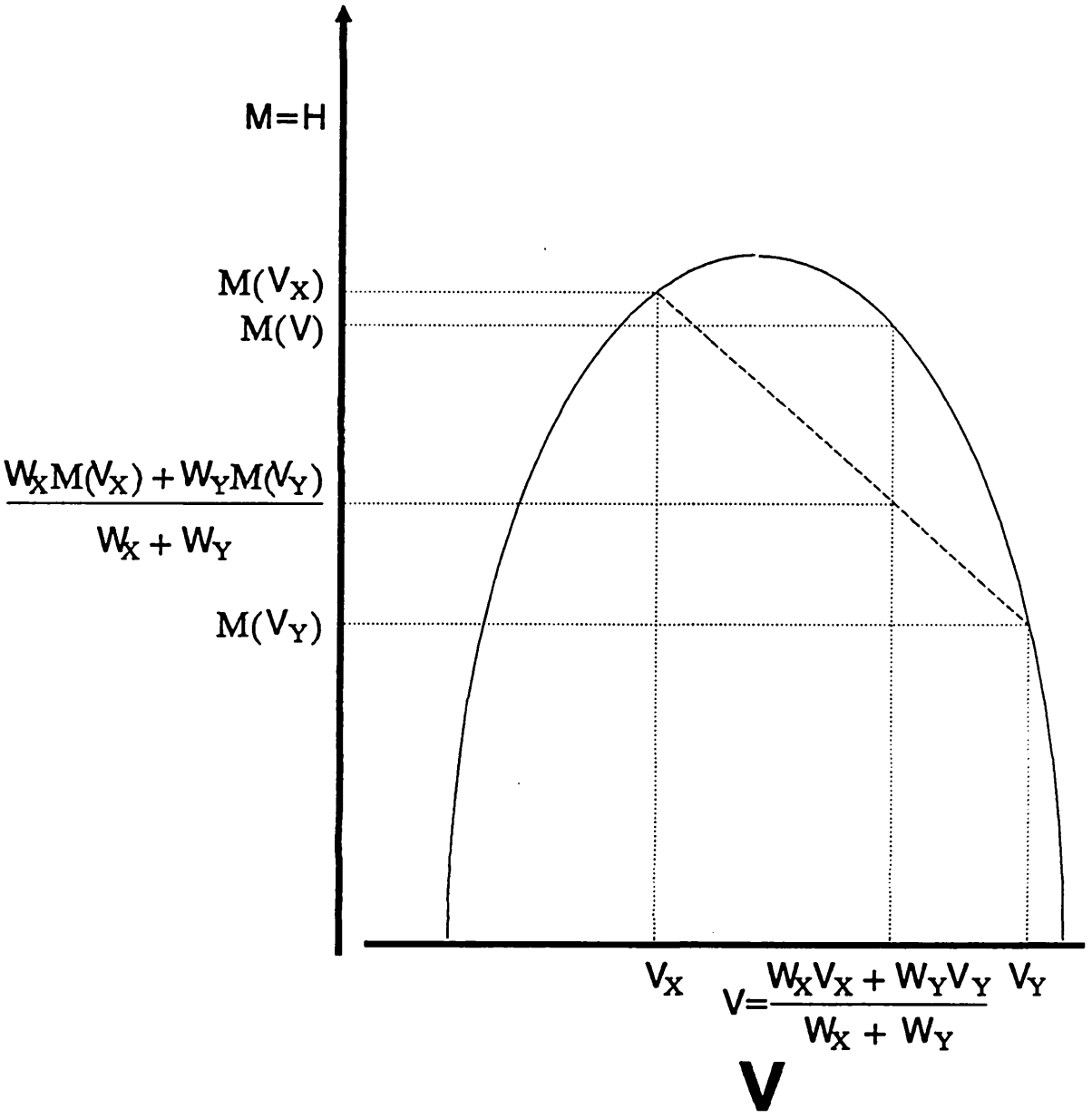


Fig 4.9

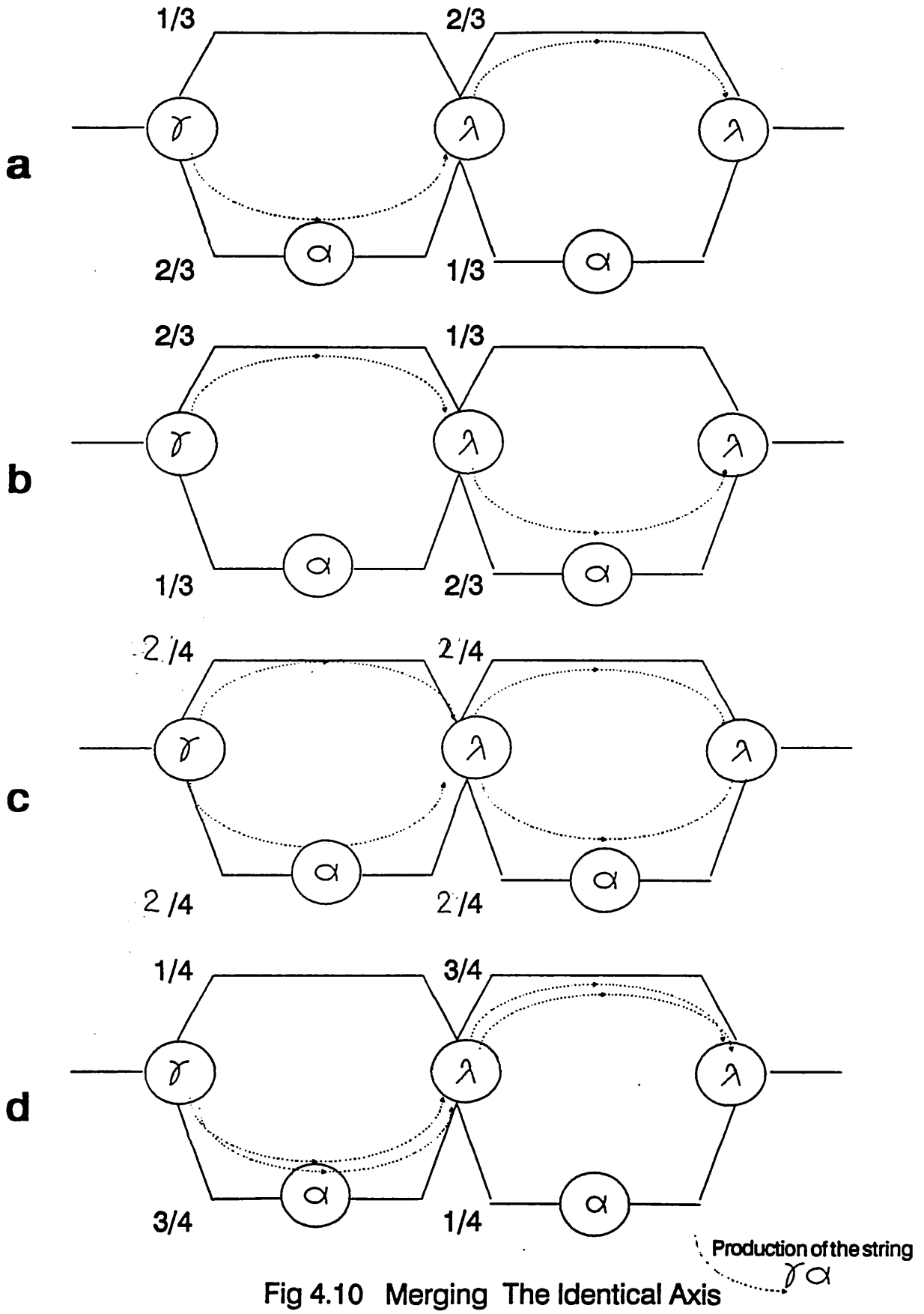


Fig 4.10 Merging The Identical Axis

Chapter 5
Conclusions And Suggestions For Further Work

CHAPTER 5
CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

5.1 CONCLUSIONS

This thesis is concerned with certain aspects relating to redundancy in communication. The two aspects that are dealt with relate to the central issues handled by Shannon in his classic paper on Information Theory [Ref 5.1]. In Chapter 2 aspects relating to channel coding are dealt with. In the chapter the approach adopted is that of adding redundancy to a signal in order to protect it against the noise and other perturbations encountered in the course of transmission. The basic idea of adding redundancy was dealt with by Shannon and found explicit formulation in his capacity theorem.

In Chapters 3 and 4, a process that can be regarded as the inverse of that dealt with in Chapter 2 is developed. The material contained in these Chapters is concerned with the process of redundancy removal. Fundamental theorems relating to this topic were also developed by Shannon in his classic paper.

The inherent interrelation between the two approaches has been widely understood since the advent of Shannon's paper. In this specific case of pattern processing it will be appreciated that reduced redundancy as dealt with in Chapters 3 and 4 might well be a prerequisite to the effective transmission and/or storage of a pattern such as a handprinted character, or a string of symbols. Once the inherent redundancy associated with a pattern has been significantly reduced, then it is more susceptible to noise and interference than when in the original redundant form. The material contained in Chapter 2 deals with one particular method by which controlled and hence more efficient use of redundancy can be re-incorporated into a signal prior to either the storage or transmission.

In Chapter 2, a new class of codes, referred to as 1.5 Reed-Muller codes (1.5 RM codes), is introduced and described. The codes are such that they add to the range of available lengths and distances as compared with those that are currently available with the existing class of Reed-Muller codes. More precisely, the new codes have the same length-to-distance ratio as certain members of the existing class of Reed-Muller codes, and the new codes are available with lengths and dimensions which lie between those values that are available with members of the existing class of RM codes that have the same order (length-to-distance ratio).

Consideration is also given in the chapter to the question of whether the new codes can be majority logic decoded. It is shown that, as formulated and synthesized, the codes cannot be decomposed in such a way as to render them majority logic decodable.

In Chapter 3, consideration is given to the pre-processing of images prior to recognition. In particular, the question of thresholding, (that is the production of a two-tone image from a grey-scale image) is considered. Important recent work on the topic of thresholding is considered and a new approach is developed in the chapter. Specifically, the work of Pal et al is reviewed and a new way of evaluating the indices of fuzziness to be used in thresholding is set out. The proposed technique has computational advantages. Also, the question of the purely automatic location of threshold positions is examined and a new Annealing-based technique is developed. The technique is shown to work well, both with binary (two-level) and with multi-level images. Following this, attention is turned to the problem of the generation of a coded representation of a line image and a new approach to the problem is put forward. The approach involves two interconnected and interrelated stages. The first is referred to as a "*Soft Skeletonization*" technique in which either a grey-scale, or a two-tone, image is processed and a pseudo grey scale image is produced using a Center of Gravity technique. It is then explained how this image is used in a stride encoding operation to produce a low redundancy representation of the image. The results presented, although limited, indicate that the technique is likely to

be useful as part of a system for the automatic recognition of images. The technique is more efficient (needs fewer bits) than existing schemes such as chain coding, and it seems to be capable of removing certain unwanted features, such as small serifs, and to be capable of bridging small unwanted gaps in and between strokes.

Following the preprocessing operations such as those dealt with in Chapter 3, the next stage in recognition is to take the strings of symbols representing patterns and, using some decision making process, to decide to which pattern class a particular pattern belongs. This is a difficult problem, since there is a significant interaction between the feature extraction process (the preprocessing and the obtaining of a string of symbols which represents a pattern) and the decision making process. Generally speaking, the more sophisticated the feature extraction the simpler the decision making process, and vice versa. At this time, there is still no general consensus as to what constitutes a "good" set of features representing the large range of patterns that are considered to be members of a class, and which make it possible to reliably distinguish between different pattern classes.

In Chapter 4 arguments are formulated relating to the modelling of handprinted characters. These arguments are of a very general nature and apply to a wide range of pattern classes.

The material contained within the chapter bears on both the question of "feature extraction" and "decision making". In the first part of the chapter, the concept of the Handprinted character is dealt with and attention is drawn to the lack of an ideal pattern as representative of a pattern class. The weakness of the template matching approach to pattern classification is discussed.

It is argued that a character (pattern class) must be considered as *a global structure of locally identifiable elements* with the human being acting as the ultimate judge of the accuracy of any OCR system.

The need for a model to describe the various pattern classes is explained and the idea of testing a model by *pattern synthesis* is put forward. Although

no attempt is made in the Chapter to develop actual models for representing character classes, attention is drawn to the fact that an acceptable model should be able to express the variations within a pattern class resulting from the stochastic nature of the characteristics of the different elements constituting a character. It is also argued that the model should be able to reflect the interrelation between the various characteristics. It is then explained in the chapter that a fully representative model of the characteristics is likely to be too complicated for analysis. As a consequence of this difficulty, attention is shifted to the search for a less succinct model which is capable of being analysed.

It is argued that the Markov Model is one such model that is worthy of investigation. The question of the criterion of inference is considered at some length and a Minimum Description Length (MDL) criterion is adopted. The choice of the cost function (that is the Description Length) is justified by Information Theoretic reasoning. The idea of using a Multidimensional Dynamic Programming approach is put forward as a means of finding the minimum description length. In this approach, the strings that are being modelled, (which in the present case are taken to represent handprinted characters, and which, in general, may be strings of symbols representing any pattern), are associated with the axes of a multidimensional space and the minimum description length has to be derived in this space. A new theorem is stated and proved by which it is shown that it is possible to merge identical axes without incurring any loss in the optimality in respect to the MDL criterion. The theorem means that the dimension of the space can be reduced and the computational burden involved in determining the minimum description length and the associated Markov model can, thereby, be significantly reduced.

Methods of further speading up the algorithm should be investigated and the ideas contained in Chapter 4 should be applied to actual pattern recognition problems.

5.2 SUGGESTIONS FOR FURTHER WORK

Although the work contained in Chapter 2 is essentially complete, there are a number of interesting and important questions that might be considered further. The first topic worthy of further investigation concerns the decoder for the new class of codes introduced in the chapter. It was explained in the chapter that, as formed, the codes cannot be Majority Logic Decoded. It is possible, however, that by using the inductive ideas set out in the chapter it might be feasible to synthesize the new codes in such a way that they can be decomposed in the manner illustrated in matrix (2.1) of Chapter 2. If such a set of codes could be found, then it might be possible to decode overall by considering the reduced length codes and then applying some relatively easy decoding technique, such for example, a simple look-up procedure, to these reduced length subcodes.

In Chapter 2, a process for generating the submatrix GM_{23} was put forward and minimum weight arguments were established, based on the construction and the associated lemma. The arguments developed at the foot of the page 26 establish the existence of a large number of acceptable mappings and a further topic worth examining is the question of whether any of these mappings has an advantage insofar as the weight distribution of the code is concerned, or whether advantages can be found with respect to decoding.[1] Furthermore, the condition expressed by the lemma is a sufficient rather than a necessary condition and the question, thus, arises as to whether a condition less restrictive than the one set out in the lemma exists, which still establishes the requirements in respect to the minimum distance of the code but which, at the same time, might result in a simpler code construction and hence, easier associated decoding. Although it is not dealt with in this thesis, it has already been established that such a construction exists for 1.5 RM codes of order 2 and 3 but, as yet, no proof has been found for higher order codes.

With regards to the work contained in Chapter 3, an interesting area worthy of further investigation concerns the question of Automatic Thresholding. A new Annealing-based technique has been put forward and

[1] see page 145b

shown to work well. It would be interesting, however, to consider other alternative approaches to the automatic thresholding and to compare them subjectively with the technique set out in this thesis.

The Center of Gravity criterion of Soft Skeletonization and the associated Stride encoder set out in Chapter 3 have been investigated for two-tone (black and white) images only. It would also be interesting to investigate how the technique works for grey- and half-tone images. Another interesting question in connection with the soft skeletonization approach is what would happen on repeated applications of the technique to a grey scale image.

A further interesting question relating to the design of the Stride encoder and the relevant Soft Skeletonization is the following: Is it possible to make use of information that can be obtained from pattern synthesis and thereby design improved Stride Encoders and similarly improved Soft Skeletonization methods? Additionally, it would be worth investigating alternatives to the Center of Gravity criterion as a means of obtaining a Soft Skeleton of a line image.

The use of Markov Network as a means of modelling a string of symbols representing a member of a pattern class was discussed in Chapter 4. The work set out in the chapter was fundamental but somewhat embryonic. It is clearly important that the effectiveness of the technique developed be investigated more fully from the point of view of the recognition of the patterns.

The question of the complexity of the Multidimensional Dynamic Programming approach presented in Chapter 4 was firstly approached through the development of a theorem allowing the merging of identical axes. The approach adopted in the chapter was based on the simultaneous (that is the parallel) consideration of a string of symbols. It would be interesting to see whether it is possible to divide the hyperspace down to subspaces, to obtain network solutions for the separate subspaces, and then to combine these solutions in order to arrive at the final optimum network. If this could be done, then the time and memory requirements involved in finding the optimum solution would be significantly reduced.

It is also worth examing if, by using several different mappings, the ideas presented in Chapter 2 can be extended, thereby resulting in the addition of other classes of codes having values of length and distance which lie between those values that are available with the original class of RM codes with the class of 1.5 RM codes.

REFERENCES

[5.1] C. E. SHANNON, "*The Mathematical Theory of Communication*", Bell System Technical Journal, 47, 1948.

PUBLICATIONS BASED ON MATERIAL CONTAINED IN THIS THESIS

- D.KABILAFKAS, L.F.TURNER, "New class of double inductive binary error correcting codes based on the class of Reed-Muller codes", (to be published) IEE Proccedings, Vol. 135, Pt. F, No 6, pp 507-512, December 1988

- D.KABILAFKAS, L.F.TURNER, "The Automatic Thresholding of Grey-Scale Images", accepted for publication in Pattern Recognition Letters, North-Holland Publishing Company.

New class of double inductive binary error correcting codes based on the class of Reed–Muller codes

D. Kabilafkas, MSc
L.F. Turner, PhD, DSc(Eng), CEng, FIEE

Indexing terms: Mathematical techniques, Matrix algebra, Coding theory

Abstract: A new class of binary codes is introduced. The codes have length $n = 1.5 \cdot 2^m$ ($m \geqslant 3$), and have a similar 'double inductive' structure to that possessed by the class of Reed–Muller (RM) codes. The new codes are structured such that their length-to-distance ratio $n/d = 2^m$, $m \geqslant 2$. For rates less than 1/2, the new codes have a higher rate than the next Reed–Muller code of the same order and length $n = 2 \cdot 2^m$.

1 Introduction

The Reed–Muller codes [1 and 2] cover a wide range of rates and minimum distances. For any m and $0 \leqslant r \leqslant m$, there exists a code in the class of Reed–Muller codes for which

$$n = 2^m \quad k = \sum_{i=0}^{m} \binom{m}{i} \quad \text{and} \quad d = 2^{m-r}$$

Such a code is usually denoted $RM(r, m)$ and the symbols n, k and d indicate, respectively, the codeword length, the number of information digits (the dimension of the code) and the minimum weight, or distance.

The generator matrix of the $RM(r, m)$ code contains the products of rows of an $m \times 2^m$ matrix A. The columns of A are the binary m-tuples representing the numbers from 0 to $2^m - 1$. The generator matrix of the $RM(r, m)$ code is formed from all ones row, the set of individual rows of A and all products of rows of A up to the product of r rows of A.

In this paper a new class of codes is introduced, having length $n = 1.5 \cdot 2^m$, $m \geqslant 3$ and integer, and minimum distance $d = 1.5 \cdot 2^{m-r}$, $1 \leqslant m - r \leqslant m - 2$. The members of the class will be referred to as $1.5RM$ codes on account of the facts:

(*a*) that they have a double inductive property that can be associated with RM codes

(*b*) the new codes have the same length-to-distance ratio as certain members of the class of the RM codes

(*c*) the members of the class of RM codes for which the length-to-distance ratio is the same as the new class of codes, are those whose generator matrix forms part of the generator matrix of the new codes.

Paper 6352F (PG, E16), first received 27th January and in revised form 29th July 1988

Mr. Kabilafkas and Prof. Turner are with the Department of Electrical Engineering, Imperial College of Science and Technology, Exhibition Road, South Kensington, London SW7 2BT, United Kingdom

The double inductive property referred to is the method which is used for the proof of the minimum distance property of the new codes, and which can also be used for the proof of the minimum distance property of the class of RM codes.

The double inductive property [1] provides an added insight into the synthesis of codes. In the case of RM codes, the double inductive property means that the generator matrix of an RM code can be expressed in terms of submatrices, which are themselves generator matrices of RM codes of the same, or higher, order and smaller length. When considering the sequence of RM codes of the same order, i.e. the same length-to-distance ratio, the $RM(r, m)$ code is based on the previous code $RM(r, m-1)$ of the sequence (i.e. the code of the same r, but of half the length (and as the length-to-distance ratio is the same, the minimum distance is also halved), and on the code $RM(r-1, m-1)$ which has half the length and the same minimum distance $d = 2^{m-r}$. This, of course, is provided both of these codes exist.

The generator matrix of the $1.5RM(r, m)$ code, which is a member of the new class of codes, is constructed in a similar manner from matrices that generate the $RM(r, m-1)$, $RM(r-1, m-1)$ and $RM(r-2, m-1)$ codes. The new codes can be shown to have length and dimension that are 1.5 times those of the $RM(r, m)$ code. Although the rate of the constructed code is less than that of the $RM(r, m)$ code, it can be shown that for rates less than 1/2 it has a higher rate than the next RM code, i.e. the $RM(r, m+1)$ code in the RM set.

2 Construction of the new codes

As mentioned above, the notation $RM(r, m)$ is used to indicate a Reed–Muller code of length $n = 2^m$, order r and minimum weight $d = 2^{m-r}$. In the case of the new codes that are to be defined, the notation $1.5RM(r, m)$ is used to indicate a code of length $n = 1.5 \times 2^m$ and minimum weight $d = 1.5 \times 2^{m-r}$. Let the generator matrices (GM) of the $RM(r, m)$ and $1.5RM(r, m)$ codes be denoted by $[RM(r, m)]$ and $[1.5RM(r, m)]$, respectively.

It can be shown (see theorem 2, Chapter 13, of Reference 1) that, if the rows of the usual form of the generator matrix of the $RM(r, m)$ code are rearranged, then the generator matrix can be expressed in terms of four submatrices as follows:

$$[RM(r, m)] = \begin{bmatrix} [RM(r, m-1)] & [RM(r, m-1)] \\ [0] & [RM(r-1, m-1)] \end{bmatrix} \quad (1)$$

For example, the usual form of the generator matrix of the $RM(2, 4)$ code is

$$\left[\begin{array}{cccccccccccccccc}
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\
0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1\\
0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1\\
0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1
\end{array}\right]$$

If the rows of this matrix are rearranged as follows:

$$\left[\begin{array}{cccccccc|cccccccc}
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
\multicolumn{8}{c|}{\text{---------------}}&\multicolumn{8}{c}{\text{---------------}}\\
0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\
0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1\\
\multicolumn{8}{c|}{\text{---------------}}&\multicolumn{8}{c}{\text{---------------}}\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1\\
0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1\\
\hline
0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
\multicolumn{8}{c|}{\text{---------------}}&\multicolumn{8}{c}{\text{---------------}}\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1\\
0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1
\end{array}\right]$$

then this matrix can be expressed in terms of the four submatrices referred to above. The appropriate partitioning of the generator matrix is also indicated in the matrix above by the continuous lines.

The new $1.5RM(r, m)$ codes are structured so that, for each member of the class, the generator matrix consists of nine submatrices, as follows:

$$[1.5RM(r, m)] = \left[\begin{array}{c|c|c}
[RM(r, m-1)] & [RM(r, m-1)] & [RM(r, m-1)] \\
\hline
[0] & [RM(r-1, m-1)] & [RM(r-1, m-1)]^* \\
\hline
[0] & [0] & [RM(r-2, m-1)]
\end{array}\right] \quad (2)$$

The matrix $[RM(r-1, m-1)]^*$ in expr. 2 generates the same code as the matrix $[RM(r-1, m-1)]$, but differs from it in a manner to be described in the following Section. All other matrices in expr. 2 are of the conventional form associated with the corresponding RM code.

The notation GM_{ij}, $1 \leqslant i \leqslant 3$, $1 \leqslant j \leqslant 3$, will be used in a conventional way to denote the vertical and horizontal position of the submatrix within the generator matrix (expr. 2); with, for ease of reference, the position of a submatrix being defined in terms of its row and column position. This must not be confused with the actual rows and columns of $[1.5RM(r, m)]$ however. From the description given in the introductory Section relating to the generator matrix of the $RM(r, m)$ code, it will be clear that the generator matrix of the $RM[r, m-1]$ code contains the products of rows of an $(m-1) \times (2^{m-1})$ matrix A. The columns of A are the binary n-tuples representing the numbers from 0 to $2^{m-1}-1$. The generator matrix of the $RM(r, m-1)$ code is formed from the all ones row, the set of individual rows of A, and all products of rows of A up to the product of r rows of A. The generator matrix $[RM(r, m-1)] = GM_{11} = GM_{12} = GM_{13}$ of the $RM(r, m-1)$ code can be divided into three parts with the parts being denoted by GM_{ijp}, $i = 1$, $p = 1, 2, 3$, respectively. The first part consists of the all ones row, the set of individual rows of A and all products up to $r-2$ rows of A. The second part consists of the set of products of $r-1$ rows of A, and the third part consists of the products of r rows of A. In general, the notation $GM_{ijp}(t)$ will be used to indicate the tth row in the matrix GM_{ijp}.

The matrix $GM_{22} = [RM(r-1, m-1)]$ in eqn. 2 is defined to consist of two parts: the first part being identical to GM_{1j1} of the matrix $[RM(r, m-1)]$ and the second part identical to GM_{1j2} of $[RM(r, m-1)]$, i.e.

$$GM_{221} = GM_{1j1}$$

and

$$GM_{222} = GM_{1j2}, \quad j = 1, 2, 3$$

The matrix GM_{33} in expr. 2 is defined to consist of only the first part, GM_{1j1}, of the matrix $[RM(r, m-1)]$. For consistency of notation, matrix GM_{33} will be denoted by GM_{331} and hence $GM_{33} \equiv GM_{331} = GM_{1j1}, j = 1, 2, 3$.

Consider now the question of the matrix $GM_{23} = [RM(r-1, m-1)]^*$ in expr. 2. It will be shown in the following Section that, if GM_{231} is identical to GM_{221}, a construction (which is not in fact unique) which results in the new class of codes is that in which GM_{232} is obtained from GM_{222} according to the following rules:

(i) row t in GM_{232} is the same as row $t+1$ in GM_{222} for $t = 1, \ldots, l-1$; where l is the dimension of GM_{222} (and of course GM_{232})

(ii) row l in GM_{232} is given by a linear combination:

$$\alpha_0 \cdot GM_{222}(1) + \alpha_1 GM_{222}(2) + \cdots + \alpha_{l-2} \times GM_{222}(l-1) + \alpha_{l-1} GM_{222}(l)$$

of the rows of GM_{222}, with $\alpha_0, \alpha_1, \ldots, \alpha_{l-1}, 1$, being the coefficients of any irreducible polynomial of degree l in $GF(2)$.

Before proving that the matrix in expr. 2 generates the new class of codes, an illustrative example is given in which the generator matrix of the $1.5RM(2, 4)$ code is constructed. Beginning with the generator matrix

$$\begin{array}{cccccccc}
1&1&1&1&1&1&1&1\\
0&0&0&0&1&1&1&1\\
0&0&1&1&0&0&1&1\\
0&1&0&1&0&1&0&1\\
0&0&0&0&0&0&1&1\\
0&0&0&0&0&1&0&1\\
0&0&0&1&0&0&0&1
\end{array}$$

which generates the $RM(2, 3)$ code, then the matrix that generates the $RM(1, 3)$ code is seen to be

$$\begin{array}{cccccccc}
1&1&1&1&1&1&1&1\\
0&0&0&0&1&1&1&1\\
0&0&1&1&0&0&1&1\\
0&1&0&1&0&1&0&1
\end{array}$$

Now, in accordance with the description given above it follows that GM_{221} is

1 1 1 1 1 1 1 1

and hence GM_{222} is

0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1

Also, GM_{231} is

1 1 1 1 1 1 1 1

If the irreducible polynomial $\alpha^3 + \alpha + 1$ is used and the rules for combination set out above are applied, then GM_{232} is seen to be

0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1
0 0 1 1 1 1 0 0

The final resulting matrix for the 1.5RM(2, 4) code of length 24, dimension 12 and minimum distance 6 is shown in Fig. 1.

1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1
.....	.....	.....
0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1	0 0 1 1 0 0 1 1	0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1	0 1 0 1 0 1 0 1	0 1 0 1 0 1 0 1
.....	.....	.....
0 0 0 0 0 0 1 1	0 0 0 0 0 0 1 1	0 0 0 0 0 0 1 1
0 0 0 0 0 1 0 1	0 0 0 0 0 1 0 1	0 0 0 0 0 1 0 1
0 0 0 1 0 0 0 1	0 0 0 1 0 0 0 1	0 0 0 1 0 0 0 1
0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1
.....	.....	.....
0 0 0 0 0 0 0 0	0 0 0 0 1 1 1 1	0 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0	0 0 1 1 0 0 1 1	0 1 0 1 0 1 0 1
0 0 0 0 0 0 0 0	0 1 0 1 0 1 0 1	0 0 1 1 1 1 0 0
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1

Fig. 1 *Generator matrix of the 1.5RM(2, 4) code*

3 Proof that matrix 2 generates the new class of codes

In this Section it will be proved that the matrix given in expr. 2 generates the new class of codes. It is clear from the generator matrix (expr. 2) that the length of the code generated is $3 \cdot 2^{m-1}$ (i.e. the length is 1.5 times that of the $RM(r, m)$ code. It is now to be proved that the minimum weight, and hence the minimum distance, of the new class of codes, is also 1.5 times the minimum weight of the $RM(r, m)$ code.

Since any codeword of the code is made up of linear combinations of rows of the generator matrix, it follows that a codeword, C, can be considered to be a juxtaposition

$$\begin{aligned} C &= (C_{11} : C_{12} : C_{13}) \oplus (C_{21} : C_{22} : C_{23}) \\ &\quad \oplus (C_{31} : C_{32} : C_{33}) \\ &= C_1 \oplus C_2 \oplus C_3 \end{aligned}$$

where C_{ij} is a linear combination of rows of the submatrix GM_{ij} and C_i, $i = 1, 2, 3$, is defined as the juxtaposition $C_{i1} : C_{i2} : C_{i3}$. The codeword can be expressed alternatively as

$$\begin{aligned} C &= (C_{11} \oplus C_{21} \oplus C_{31}) : (C_{12} \oplus C_{22} \oplus C_{32}) : \\ &\quad (C_{13} \oplus C_{23} \oplus C_{33}) \\ &= \check{C}_1 : \check{C}_2 : \check{C}_3 \end{aligned}$$

in which $\check{C}_j$, $j = 1, 2, 3$, is the modulo 2 sum $C_{1j} \oplus C_{2j} \oplus C_{3j}$.

Clearly, the weight, wt, of C, is

$$wt(\check{C}_1 : \check{C}_2 : \check{C}_3) = wt(\check{C}_1) + wt(\check{C}_2) + wt(\check{C}_3)$$

Now, as has been explained previously, each of the submatrices GM_{ij} consists of three parts, of two parts or of a single part. Hence it follows that:

$$C_{ij} = C_{ij1} \oplus C_{ij2} \oplus C_{ij3} \tag{3}$$

where C_{ijp} is a linear combination of rows taken from the corresponding submatrix GM_{ijp}. For those submatrices consisting of three parts, for example $[RM(r, m-1)]$, eqn. 3 applies directly. For those submatrices consisting of two parts, for example $[RM(r-1, m-1)]$ and $[RM(r-1, m-1)]^*$, C_{ij3} does not exist. For submatrices, such as $[RM(r-2, m-1)]$, which consist of one part, only C_{ij1} exists. For clarity of exposition the situation is illustrated in Fig. 2. It is important to note that the codeword construction in terms of the nine parts $C_{11}, \ldots, C_{33}$ is unique, as is the decomposition into the parts C_{ijp}.

$$\left[\begin{bmatrix} C_{111} \\ C_{112} \\ C_{113} \end{bmatrix} \begin{bmatrix} C_{121} \\ C_{122} \\ C_{123} \end{bmatrix} \begin{bmatrix} C_{131} \\ C_{132} \\ C_{133} \end{bmatrix}\right]$$

$$\left[\begin{bmatrix} 0 \\ 0 \\ \text{nul} \end{bmatrix} \begin{bmatrix} C_{221} \\ C_{222} \\ \text{nul} \end{bmatrix} \begin{bmatrix} C_{231} \\ C_{232} \\ \text{nul} \end{bmatrix}\right]$$

$$\left[\begin{bmatrix} 0 \\ \text{nul} \\ \text{nul} \end{bmatrix} \begin{bmatrix} 0 \\ \text{nul} \\ \text{nul} \end{bmatrix} \begin{bmatrix} C_{331} \\ \text{nul} \\ \text{nul} \end{bmatrix}\right]$$

Fig. 2 *Decomposition of the generator matrix of a 1.5RM code indicated by the corresponding parts C_{ijp}, nul = nonexistent*

From the way in which the codewords of the new code are constructed it follows that, in terms of the notation of above

$$C_{11p} = C_{12p} = C_{13p} \quad p = 1, 2, 3 \tag{4}$$

and also

$$C_{221} = C_{231} \tag{5}$$

Lemma: $C_{222} = C_{232}$ if and only if $C_{222} = C_{232} = 0$

Proof of lemma: It is known from the theory of finite fields [3] that if a set of 2^k elements forms a group under addition, a multiplication operation can be defined, which has a unique inverse, and hence the set has the structure of the field $GF(2^k)$. Now, since any codeword is obtained by multiplying an input information vector by the generator matrix of the code, it follows that the invertible operations of multiplication and addition between any two codewords can be affected by, and considered to be equivalent to, applying multiplication and addition between the corresponding input information vectors, prior to the multiplication by the generator matrix. This follows since the operation of encoding is invertible. Thus, operations with respect to codewords can be considered with respect to the set of 2^k input information vectors.

Using the normal polynomial representation, an input information vector $i_1, i_2, \ldots, i_k$ can be represented by a polynomial of degree $k-1$, or less; i.e.

$$i_1, i_2, \ldots, i_k \rightarrow i_1 + i_2 X + \cdots + i_k X^{k-1}$$

The addition of input vectors can be carried out by simply adding polynomials element by element, and the multiplication of input vectors can be carried out by multiplying the corresponding polynomials modulo an irreducible polynomial $A(x)$, of degree k. Such irreducible polynomials exist for degrees ≥ 2 [3].

Consider any information polynomial $I(X)$ of degree equal to or less than $k-1$ and consider an irreducible polynomial $A(X)$ of degree k. Consider also another polynomial $B(X)$ of degree equal to or less than $k-1$. This can be considered to be another information vector. It can be proved that

$$(I(X)B(X)) \bmod A(X) \neq I(X) \tag{6}$$

Now, $(I(X)B(X)) \bmod A(X)$ forms the same space of polynomials as the set $I(x)$ itself, and hence the set $XI(X)$ when used in conjunction with a generator matrix GM generates the same code as the input information vector set $I(X)$. However, the codeword generated by $XI(X)$ is different from that generated by $I(X)$ itself. This follows directly from expr. 6.

Consider now the creation of a matrix GM^* which has the following property: when GM^* is used as generator matrix the codeword generated by an information vector $I(X)$ is the same as that generated by $XI(X)$ when the generator matrix GM is used.

On account of the linearity of the codes involved, GM^* has the desired property if it possesses it for the basis set of input information vectors $1, X, \ldots, X^{k-1}$. Consider first the basis vectors $1, X, \ldots, X^{k-2}$. Now these information vectors, when multiplied by X, become $X, X^2, \ldots, X^{k-1}$ respectively, and when using the generator matrix GM they result in codewords which correspond to the 2nd, 3rd, ..., kth rows of GM respectively. Therefore, it is clear that a GM^* having the required property for the particular set of input polynomials being considered is a matrix which has as its ith row the $(i+1)$th row of GM.

Consider now the input information vector X^{k-1}. When this information vector is multiplied by X and reduced modulo

$$A(X) = \alpha_0 + \alpha_1 X + \cdots + \alpha_{k-1} X^{k-1} + X^k$$

the result is a polynomial $\alpha_0 + \alpha_1 X + \cdots + \alpha_{k-1} X^{k-1}$ and hence if the matrix GM^* is to have the desired property, it must have as its kth row the n-tuple

$$\alpha_0 \text{ (row 1)} \oplus \alpha_1 \text{ (row 2)} \oplus \cdots \oplus \alpha_{k-1} \text{ (row } k\text{)} \tag{7}$$

where (row i) in expr. 7 is the ith row of GM. This completes the proof of the lemma.

It now remains to be shown that the generator (expr. 2) generates a code having minimum weight $1.5 \cdot 2^{(m-r)}$. The minimum weight of the new class of codes can be determined by considering the minimum weight associated with certain subsets of codewords. These various subsets can be considered as various cases with the subsets being chosen so as to cover the whole code. Let $wt(C_{ij})$ and $wt(C_{ijp})$, respectively, denote the minimum weight of the vector associated with the previously defined linear combinations C_{ij} and C_{ijp} of rows.

Case 1: $\check{C}_1 = 0, \check{C}_2 = 0, \check{C}_3 = 0$. Clearly, this case corresponds to the zero vector, therefore $wt(C) = 0$.

Case 2: $\check{C}_1 = 0, \check{C}_2 = 0, \check{C}_3 \neq 0$. Thus, $C_{11} = C_{12} = C_{13} = 0$ and $C_{22} = 0$. In this case $C_{221} = C_{231} = 0$ and by the lemma $C_{222} = 0 = C_{232}$. Therefore the weight of $C = wt(C_{33}) \geq 2^{(m-r)+1}$, since GM_{33} is defined as that generator matrix which generates an RM code of minimum weight $2^{(m-r)+1} (> 1.5 \cdot 2^{(m-r)})$.

Case 3: $\check{C}_1 = 0, \check{C}_2 \neq 0, \check{C}_3 = 0$. Thus, $C_{11} = C_{12} = C_{13} = 0$ and $C_{22} \neq 0, C_{23} \oplus C_{33} = 0$. $\check{C}_2 \neq 0$ means that the all zero combination is excluded and hence the minimum weight of C_{22} is equal to $2^{(m-r)}$, since GM_{22} is the generator maxtrix of an RM code having minimum weight $2^{(m-r)}$. Since $\check{C}_3 = 0$ it is clear that $C_{23} = C_{33}$ and hence that $C_{331} = C_{231} = C_{221}$ and from the lemma that $C_{232} = C_{222} = 0$. Therefore the weight, $wt(C)$, of $C = wt(C_{221}) \geq 2^{(m-r)+1}$, since GM_{221} generates an RM code of minimum weight $2^{(m-r)+1}$.

Case 4: $\check{C}_1 = 0, \check{C}_2 \neq 0, \check{C}_3 \neq 0$. Assuming $C_{11} = C_{12} = C_{13} = 0$ and $C_{22} \neq 0$. By the second assumption: the minimum weight of $C_{22} = 2^{(m-r)}$, as mentioned before. In this case $wt(C_{23} \oplus C_{33}) \geq 2^{(m-r)}$ since the code generated by GM_{33} is contained within the code generated by GM_{23}; and GM_{23} is a matrix which generates an RM code of minimum weight $= 2^{(m-r)}$. It is thus clear that the minimum weight is

$$\begin{aligned} wt(C) &= wt(C_{22}) + wt(C_{23} \oplus C_{33}) \\ &\geq 2^{(m-r)} + 2^{(m-r)} \\ &= 2^{(m-r)+1} \end{aligned}$$

Case 5: $\check{C}_1 \neq 0, \check{C}_2 = 0, \check{C}_3 = 0$. Thus $C_{11} = C_{22} = C_{33} \neq 0$ and $wt(C_{1j}) \geq 2^{(m-r)-1}$. From $\check{C}_2 = C_{21} \oplus C_{22} \oplus C_{23} = 0$ and $\check{C}_3 = C_{31} \oplus C_{32} \oplus C_{33} = 0$ and since neither C_{223} or C_{233} exist, it follows that $C_{123} = 0$ and in consequence $C_{113} = C_{133} = 0$. Since $\check{C}_2 = 0$ and $C_{23} = 0$ it follows that $C_{12} = C_{22}$. Therefore, $C_{121} = C_{221}$, $C_{122} = C_{222}$, and at least one of $C_{121} (= C_{221})$ and $C_{122} (= C_{222})$ is nonzero. Now, using the fact that $\check{C}_3 = 0$ it follows that $C_{131} \oplus C_{231} \oplus C_{331} = 0$. That is, $C_{131} \oplus C_{231} = C_{331}$. But, by virtue of the way the generator matrix (expr. 2) is structured, it is clear that $C_{131} = C_{231}$ and hence $C_{131} \oplus C_{231} = 0 = C_{331}$. Again, by virtue of the structure of matrix 2, $C_{132} \oplus C_{232} = 0$ and $C_{132} = C_{232}$. From above we have that $C_{122} = C_{222}$ and from eqn. 4 $C_{112} = C_{122} = C_{123}$. Thus $C_{222} = C_{122} = C_{132} = C_{232}$. But by the lemma $C_{222} = C_{232}$ implies that $C_{222} = C_{232} = 0$. Hence, $C_{112} = C_{122} = C_{132} = C_{222} = C_{232} = 0$ which also means that (see above) $C_{121} \neq 0$ or $C_{111} = C_{111} = C_{131} \neq 0$ and it thus follows that the weight, $wt(C) = wt(C_{111})$ and this is seen to be $\geq 2^{(m-r)} + 1$.

Case 6: $\check{C}_1 \neq 0, \check{C}_2 = 0, \check{C}_3 \neq 0$. It is proved above that when $\check{C}_2 = 0$ then $C_{113} = C_{123} = C_{133} = 0$ and so $wt(C) = wt(\check{C}_1 : \check{C}_3) = wt(\check{C}_1) \oplus wt(\check{C}_3) \geq 2^{(m-r)} + 2^{(m-r)} = 2^{(m-r)+1}$.

Case 7: $\check{C}_1 \neq 0, \check{C}_2 \neq 0, \check{C}_3 = 0$. By the same arguments as before it is clear that since $\check{C}_3 = 0$ and $C_{113} = C_{123} = C_{133} = 0$ and therefore $wt(C) = wt(\check{C}_1 : \check{C}_3 : 0) = wt(\check{C}_1) + wt(\check{C}_2) \geq 2^{(m-r)} + 2^{(m-r)} = 2^{(m-r)+1}$.

Case 8: $\check{C}_1 \neq 0, \check{C}_2 \neq 0, \check{C}_3 \neq 0$. Since $\check{C}j \neq 0$, $j = 1, 2, 3$, $wt(\check{C}j) \geq 2^{(m-r)-1}$. Therefore, $wt(C) = wt(\check{C}_1 : \check{C}_2 : \check{C}_3) = wt(\check{C}_1) + wt(\check{C}_2) + wt(\check{C}_3) \geq 2^{(m-r)-1} + 2^{(m-r)-1} + 2^{(m-r)-1} = 1.5 \cdot 2^{(m-r)}$.

This last case, which completes the proof, also shows that the minimum weight is not greater than $1.5 \cdot 2^{(m-r)}$. It is interesting to note that the last case is the only case in which the minimum weight is equal to $1.5 \cdot 2^{(m-r)}$, and all the other cases have a minimum weight of $2^{(m-r)+1}$ which has implications relating to the weight distribution of the code.

4 Dimensionality considerations and results

The new 1.5*RM* codes can be considered within the sequences of *RM* codes of constant length-to-distance ratio n/d. Fig. 3 shows the sequence for three different length-to-distance ratios, n/d, with, for each length-to-distance ratio, the code dimension, k, being plotted as a function of codeword length, n. The axes of the figure are plotted to a logarithmic scale and the half-rate locus is shown, as a heavy dotted line. The loci of other constant rates are lines parallel to the 1/2 rate locus, with rates $< 1/2$ being to the right of the 1/2 rate locus and values in excess of 1/2 being to the left of the 1/2 rate locus.

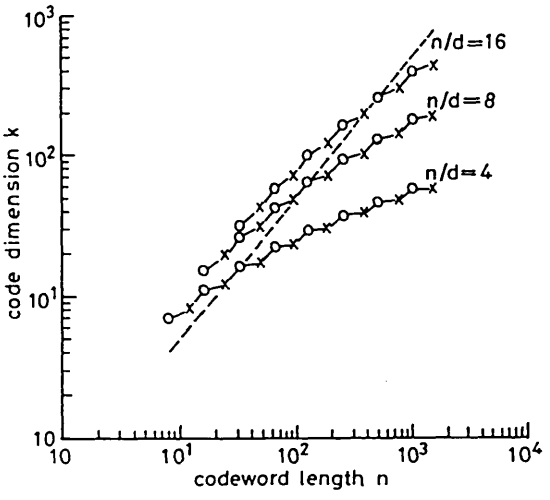


Fig. 3 *Dimension against length for the RM and 1.5RM codes*
OOO *RM* codes
+++ 1.5*RM* codes
---- 1/2 rate locus

It is important to consider the dimensionality of the new class of codes and its relation to the dimension of members of the *RM* class of codes. The dimension, k_m, of the $RM(r, m)$ code is $k_m = \sum_{i=0}^{r} \binom{m}{i}$ and the dimension, k_{m+1}, of the $RM(r, m+1)$ code which has the same length-to-distance ratio as the $RM(r, m)$ is $k_{m+1} = \sum_{i=0}^{r} \binom{m+1}{i}$. Using the relation $\sum_{i=0}^{r} \binom{m+1}{i} = 2\sum_{i=0}^{r} \binom{m}{i} - \binom{m}{r}$ it follows that $k_{m+1} = k_m - \binom{m}{r}$. The corresponding rates for the codes are

$$\frac{\sum_{i=0}^{r} \binom{m}{i}}{2^m} \quad \text{and} \quad \frac{2\sum_{i=0}^{r} \binom{m}{i} - \binom{m}{r}}{2^{m+1}}$$

respectively. It is thus clear that, for a given length-to-distance ratio, the rate of an *RM* code decreases by a value of $\binom{m}{r}(1/2^{m+1})$ as the length is doubled.

In the case of the dimension, $k_{1.5m}$, of the new codes it is clear from the way they are constructed that: $k_{1.5m} = k_m + \sum_{i=0}^{r-2} \binom{m-1}{i} = 1.5k_m - 1/2\{\binom{m}{r} + \binom{m-1}{r-1}\}$. Thus, as the length of the $RM(r, m)$ code increases by a factor of 1.5, the rate decreases by $\{1/2\binom{m}{r} + \binom{m-1}{r-1}\}/1.5 \cdot 2^m$ and therefore the same general pattern of behaviour is observed as with the *RM* class of codes.

Comparison of the rates of the $RM(r, m)$ and the $1.5RM(r, m)$ codes shows that, for a given length-to-distance ratio, the rate of the $1.5RM(r, m)$ code is less than the rate of the $RM(r, m)$ code. A similar comparison of the values of the rate of the $RM(r, m+1)$ code with that of the $1.5RM(r, m)$ code shows that the difference, ΔR, in rates is

$$\Delta R = \text{rate of } RM(r, m+1) - \text{rate of } 1.5RM(r, m)$$
$$= \frac{1}{6 \cdot 2^m}\binom{m}{r}\left(2\frac{r}{m} - 1\right)$$

The various points raised above are illustrated in Fig. 3. From the figure it is clear that for both the original class of Reed–Muller codes, and the new class, as the codeword length is increased, for a given length-to-distance ratio, the rate of the code decreases. This is shown by the fact that the locus for both the original *RM* codes and the new class of codes starts to the left of the 1/2 rate locus and crosses it as n increases.

For a given length-to-distance ratio, the new class of codes has the property that the rate of the next longer *RM* code is less than that of the (immediately previous) new code, if both rates are less than 1/2. The converse is true if the rates are greater than 1/2. The relative rates of the codes can be obtained by measuring the perpendicular distance of the code (the point) under consideration from the 1/2 rate locus.

5 Conclusions

In this paper a new class of codes, referred to as 1.5*RM* codes, has been developed and described. The codes are such that they add to the range of available lengths and distances as compared with those that are currently available with the existing class of *RM* codes. More precisely, the new codes have the same length-to-distance ratio as certain members of the existing class of Reed–Muller codes and for those cases in which the length-to-distance ratio is the same as for those members of the existing class of RM codes; the new codes are available with lengths and dimensions which lie between those values that are available with the existing class of *RM* codes. This feature is illustrated in Fig. 3.

Although no consideration has been given in this paper to the question of the decoding of the new codes, the double inductive property can be used to show that, in the form presented, the new codes cannot be majority logic decoded. However, the question remains open as to whether it is possible using the double inductive concept, to synthesise codes having the same dimension and distance property as the new codes, but to do so in such a way that they can be decomposed into lower order 1.5*RM* codes in the manner illustrated in generator matrix (expr. 1) for the existing class of *RM* codes. If such a set of codes could be found, then it might well be possible to decode overall by considering the reduced length codes and applying some relatively easy decoding technique, for example, a simple look-up procedure, to these reduced length subcodes.

6 Acknowledgments

Mr. Kabilafkas wishes to thank Dr. M.V.D. Burmester who acted as his supervisor at Royal Holloway and

Bedford New College (formerly Royal Holloway College) for introducing him to the coding theory and advanced algebra. He also wishes to thank the State Scholarship Foundation of Greece for the financial support both for his MSc studies at Royal Holloway College and PhD studies at Imperial College.

7 References

1 McWILLIAMS, F.J., and SLOANE, N.J.A.: 'The theory of error-correcting codes', **1 and 2** (North-Holland, Amsterdam, 1977)
2 PETERSON, W.W., and WELDON, E.J.: 'Error-correcting codes' (MIT Press, Cambridge, MA, 1972)
3 FRALEIGH, J.B.: 'A first course in abstract algebra' (Addison-Wesley, Reading, MA, 1967)

THE AUTOMATIC THRESHOLDING OF GREY-SCALE IMAGES

by D. Kabilafkas, Dip. E.E., M.Sc. and L.F.Turner Ph.D., D.Sc., C.Eng., F.I.E.E.

Abstract:

Techniques for the automatic thresholding of grey-scale images are investigated. A computationally efficient method for evaluating a class of functions used in thresholding is described and a new annealing-based technique for automatically locating threshold positions is presented.

1. Introduction

A common and important step involved in the processing of images is that involved in going from a grey scale representation of a pixel to a corresponding binary coded representation of the pixel. The process of establishing quantization levels, which is an essential step in forming a binary coded representation of a pixel, has come to be known as "thresholding".

The process of thresholding is important since the information so derived is used in further subsequent analysis of the image. As the process of thresholding is irreversible i.e. is information lossy, care must be taken to ensure that the process preserves adequate information for use in further processing. The thresholding technique may be designed to take advantage of the nature of the equipment involved in producing the image, or may make use of information available about the equipment (the camera etc) used when taking signals from the "paper" and converting them into electrical form for further manipulation. Thresholding is by no means a simple matter and it is not easy to treat mathematically. Study has been given to efficient methods for carrying out the thresholding process and an interesting and important relatively recent development is that due to Pal et al (Ref [1]).

In their paper Pal et al use arguments drawn from fuzzy set analysis to arrive at a function that can be used to determine the position at which a threshold(s) should be placed and they argue that this function can be evaluated without the need to refer directly to the histogram of intensities. This proposal is, however, inefficient with regard to the number of computations involved. In this paper an alternative method is proposed for evaluating the function which is computationally more efficient. In Ref [1] the thresholds were placed at the middle of broad minima associated with the function and this was done by human visual inspection. Clearly, it would be advantageous if the location of the position of the thresholds could be done purely automatically. In this paper a new technique based on the concept of annealing as used in the refining of hypothesis testing is put forward as a means of achieving purely automatic threshold location.

The computationally effective technique for evaluating the function to be used in determining the threshold positions is presented in Section 2, and in Section 3, the technique for automatically locating thresholds is given. The results of the application of this technique are contained in Section 4.

2. A Computationally Efficient Technique for Evaluating a Class of Functions Used in Thresholding.

With a view to locating the positions of thresholds to be used in the quantization of an image Pal et al (Ref [1]) use arguments drawn from fuzzy set analysis in order to arrive at a function which can subsequently be used to determine the position at which thresholds should be placed.

An image is considered to be made up of M by N pixels and the following indices (v_l, v_q and H(X)) of fuzziness are suggested:

$$v_l = \frac{2}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left|\mu_X(x_{mn}) - \mu_X(x_{mn})\right| \tag{1}$$

$$v_q = \frac{2}{\sqrt{MN}}\left(\sum_{m=1}^{M}\sum_{n=1}^{N}\left(\mu_X(x_{mn}) - \mu_X(x_{mn})\right)^2\right)^{1/2} \tag{2}$$

$$H(X) = \frac{1}{MN\ln 2}\sum_{m=1}^{M}\sum_{n=1}^{N} Sn\left(\mu_X(x_{mn})\right) \tag{3}$$

in which x_{mn} denotes the intensity of the $(m,n)^{th}$ pixel, $\mu_X(x_{mn})$ denotes the grade of possessing some property μ_{mn} by $(m,n)^{th}$ pixel which has intensity x_{mn} (the Zadeh S-function is the property used by Pal et al), $\mu_X(x_{mn})$ denotes the nearest ordinary intensity plane such that $\mu_X(x_{mn})=0$ if $\mu_X(x_{mn})\leq 0.5$ and $\mu_X(x_{mn})=1$ if $\mu_X(x_{mn})>0.5$. In more physical terms $\mu_X(x_{mn})$ corresponds to the black and white extreme levels into which the image would be quantized, when using a two level quantizer, and Sn is the Shannon's entropy function defined by

$$Sn\left(\mu_X(x_{mn})\right) = -\mu_X(x_{mn})\ln\left(\mu_X(x_{mn})\right) - \left(1-\mu_X(x_{mn})\right)\ln\left(1-\mu_X(x_{mn})\right)$$

The results presented in Ref [1] relating to the thresholding of binary images indicate that for the images considered there appears to be little difference in the threshold arrived at using the three indices referred to above.

Further insight into the evaluation of the indices proposed by Pal et al can be gained by considering the general form of the equations 1 to 3. It is important to note that the arguments and conclusions reached in considering these equations extend to other indices.

Equations 1 to 3 above can be expressed in the general form

$$I(X) = E\left(F(\mu(X))\right)$$

where X is the image, $F(\mu(X))$ is the functional (the index of fuzziness in Ref [1]) and E indicates the averaging of the function $F(\mu(X))$ over the image. As mentioned earlier, in Ref [1] the function μ_X is the standard S function of fuzzy set theory which can be defined to be $\mu(l) = S(l, l_0, \delta l)$, where l is the intensity level of a given pixel, l_0 is the threshold (that is b in the notation of Ref [1]), and δl is the bandwidth (bandwidth is defined in Ref [1] to be $\Delta b = c - a$)

Hence:

$$\begin{aligned}
\mu(l) &= 0 && \text{for} \quad 0 \le l \le l_0 - \tfrac{\delta l}{2} \\
&= 2\left(\tfrac{l-l_0}{\delta l}\right)^2 && \text{for} \quad l_0 - \tfrac{\delta l}{2} \le l \le l_0 \\
&= 1-2\left(\tfrac{l-l_0}{\delta l}\right)^2 && \text{for} \quad l_0 \le l \le l_0 + \tfrac{\delta l}{2} \\
&= 1 && \text{for} \quad l_0 + \tfrac{\delta l}{2} \le l \le l_{\max}
\end{aligned}$$

In Ref [1] it is suggested that I(X) can be evaluated without the need to refer directly to the histogram of intensities, and the form given in Ref [1] for I(X) suggests that it can be evaluated by considering each of the N·M pixels constituting the image and forming the appropriate summation. If, however, use is made of the histogram then the number of computations can be significantly reduced. The way in which this can be achieved is to realize that

$$I(X) = \underset{X}{E}\Big(F(\mu(X))\Big) = \frac{1}{MN}\sum_m\sum_n F\Big(\mu(X_{mn})\Big) = \frac{1}{MN}\sum_{l=0}^{l_{\max}} \sum_{\substack{m,n \\ x_{mn}=l}} F\Big(\mu(l)\Big) \tag{4}$$

where $\sum_{\substack{m,n \\ x_{mn}=l}}$ indicates the summation over all pixels within the N by M pixel image that have intensity l and hence it is clear that Equation 4 can be written as

$$I(X) = \frac{1}{MN}\sum_{l=0}^{l_{\max}} \text{HIST}(l)\cdot F(\mu(l))$$

where HIST(l) is the number of pixels in the histogram of the image which have intensity l.

But, from above, $\mu(l)$ is equal to the S function, that is,

$$\mu(l) = S(l, l_0, \delta l)$$

which can be written as

$$\mu(l) = S(l, l_0, \delta l) = S_{\delta l}(l - l_0)\ ;$$

since for a given δl, the function S, and hence $\mu(l)$, is a function of the difference $l - l_0$ between the intensity and the threshold l_0 being considered. For a given δl, the combined operation of weighting in accordance with the S function and then using this within the F function can be considered as an overall "windowing" denoted by $W_{F,\delta l}(l - l_0)$ which is the convolution $W_{F,\delta l} = F * S_{\delta l}$.

Hence, finally, we have

$$I(X) = \frac{1}{MN}\sum_{l=0}^{l_{\max}} \text{HIST}(l)\cdot W_{F,\delta l}\Big(l - l_0\Big) \tag{5}$$

In those cases in which the window is not symmetrical

$$I(X)=\frac{1}{MN}\sum_{l=0}^{l_{max}} \text{HIST}(l)\cdot\hat{W}_{F,\delta l}\left(l-l_0\right) \text{ where } \hat{W}(l)=W(-l)$$

The overall windowing functions corresponding to the fuzzy indices put forward in Ref [1] and defined in Equations 1,2 and 3 above are shown plotted in Fig 1. The windows are drawn for a normalized bandwidth $\delta l=1$ and the effect of changing bandwidth is simply to horizontally scale a window. The significance of the Equation 5 is: (i) by using the histogram the number of computations is reduced since all that is necessary is for the number of pixels having a given particular intensity (between 0 and l_{max}) to be multiplied by the windowing function and for this to be done for each of the levels $0,1,\ldots,l_{max}$; (ii) standard computational packages for evaluating convolutions are available and hence simply by obtaining the windowing function and convolving this (using existing programs) with the histogram, the function I(X) can be obtained; (iii) the approach developed shows that the technique of thresholding developed by Pal et al can be considered within the wider context of windowing and this opens up the possibility of studying thresholding (and its relation to the subjective acceptability of the images after thresholding) in a new light. Some results relating to the computational advantages referred to directly above in (i) are given in Section 4.

3. An Annealing-Based Technique for the Automatic Location of Thresholds

In Ref [1] a fuzziness index was obtained as a function of threshold point l_0 and a threshold was associated with each broad deep minimum of the index function and located at the middle of the broad minimum.

As an aid to locating the positions of thresholds, it would be of value to use the fuzziness index technique and to combine it with some purely automatic method for finding the position of the thresholds (the number of which may be a-priori unknown). A technique for doing this, which is based on similar ideas to those contained in the annealing schemes that have been put forward (Ref [2]) for the refining of hypothesis testing, is as follows:

(1) Compute the I(X) function (using some given window function) for an initially high number of threshold positions.

(2) Convert the distribution I(X) to a function $I(X)'=k-I(X)$ with k being a constant which is chosen so that $k-I(X)=0$ for the largest value of I(X) that is possible. The reason for doing this is that the function I(X) in which minima are associated with the desired positions of the thresholds is converted to a function in which maxima are associated with the most probable positions of the thresholds. There are many ways in which the function

I(X) can be "translated" and a number of alternatives have been examined. It has been found, however, that the translation $I(X)' = k - I(X)$ performs best.

(3) The set of distinct $I(X)'$'s are then split and divided, with neighbouring points being combined so as to obtain a number of discrete $I(X)'$'s that are of equal magnitude, but which are positioned at the centre of gravity of the merged points. The rationale behind this operation is that points having large values of $I(X)'$ tend to attract their neighbours.

The operation involved in Step 3 can be illustrated with the simple example shown in Fig 2. In the figure it is assumed that 5 initial threshold points have been selected and that the values of $I(X)'$ associated with these threshold points are shown in part **a** of the figure. The average value of $I(X)'$ is seen to be 3. The next operation involves splitting the values of $I(X)'$ as shown in part **b** of the figure and dividing into groups each having a total weight equal to the average value of $I(X)'$. The groups satisfying this condition are shown contained within the dotted lines. The members of each group are then converted into a single element of weight equal to the average weight, with this being positioned at the centre of gravity of the members of the group. This is illustrated in part **C** of the figure. The result of this operation is a new set of threshold points, indicated in part **C** of the figure. The process of splitting, dividing and combining is then repeated using the new points of threshold location. Steps 2 and 3 are repeated until a sufficiently close packing of the thresholds is achieved in respect of a particular broad minimum in the original I(X) function.

A practical illustration of the overall process is illustrated in Fig 3. This figure is based upon data given in Ref [1]. It will be noted from Fig 3 that there is a tendency towards congregation of points close to the maxima of I(X), ie close to the minima of the original discrete I(X) function.

Clearly the extent of the convergence depends on the number of iterations involved and the bandwidth of the windowing function. The shape of the window function is also important in that by appropriate choice it is possible to obtain enhanced detection of minima above and beyond that which can be obtained simply by using a reduced bandwidth.

Since the single window uses only local histogram information, it can only detect areas of low value, irrespective of the value of I(X) of the window in the neighbourhood of the window. To remedy this defect, a three-unit window was used in order to discriminate between local minima and broad flat regions of the histogram. The window structure is shown in Fig 3 with the associated intensities $I_0(X)$, $I_+(X)$, $I_-(X)$ being indicated. It is clear that the characteristic of a local minimum is its low value in comparison with the values of its side windows. The combined intensity of the side windows, $I_+(X)$ and $I_-(X)$, taken either as the arithmetic or geometric mean value of the window intensities, is subtracted from the intensity of the central window $I_0(X)$. The annealing intensity function, I(X), is thus, $I(X) = I_0(X) - \frac{I_+(X) + I_-(X)}{2}$ and $I(X) = I_0(X) - \left(I_+(X) \cdot I_-(X)\right)^{1/2}$, for arithmetic and geometric mean value subtraction respectively.

4. Results

The computational advantages to be gained from using the technique presented in Section 2 have been evaluated relative to the technique suggested in Ref [1]. The results are illustrated in Fig 4. From the figure it can be seen that greater than an order of magnitude saving results from application of the technique presented in Section 2

Application of the annealing technique is shown in Figs 5, 6 and 7. The histogram data for Fig 5 was obtained by digitizing the newspaper image which is shown in Fig 6a. The image was digitized to a resolution of 8 bits per pixel. The figure deals with the situation in which the histogram of intensities contains a single local minimum. In applying the annealing technique to the data contained in Fig 5, a window was used which consisted of three identical entropy-based windows similar to the entropy-based window in Fig 1.

The annealing intensity function, I(X), used was $I(X) = I_0(X) - \frac{I_+(X) + I_-(X)}{2}$. The window function was such that the window width of each unit was initially set at 48 and then reduced to 32,16,8 and 4 after each successive 10 iterative steps. From Fig 5 it is clear that there is a convergence towards a local minimum. Fig 6b shows the newspaper image after it has been quantized to two levels when using the threshold arrived at and shown in Fig 5. In Fig 6c the two-level quantized image obtained when using the middle(128) of the range of intensities (0 to 255) is shown. Comparison of Figs 6b and 6c indicates that subjectively more acceptable results are obtained when using the threshold derived by annealing.

The technique described above has also been applied to an image having a multi-modal intensity histogram and has been found not to work well since certain local minima are not detected. In order to overcome this problem the geometric mean of the side window intensities was used in conjunction with the previously described window function.

Application of the modified intensity function has been investigated using the histogram material contained in Ref 1. A bandwidth of 11 was used for each unit of the window. The results of the application are shown in Fig 7. From this figure it is clear that the technique works well. Of the initial 39 seed points, 9 converged to the obvious lowest local minimum, 9 converged to the middle local minimum and 19 converged on to the highest obvious local minimum. One seed point converged to the low end of the histogram and one to the high end. These latter two convergences can be considered as very weak and can be discarded. It is also likely that if the number of iterations had been increased, then these points would also have converged to the nearest obvious local minima. A total of only 20 iterations was involved in arriving at the results shown in Fig 7.

5. Conclusions

The question of the automatic thresholding of grey scale images has been addressed in this paper and a new computationally efficient method for evaluating a class of functions used in thresholding is described. It is shown that greater than an order of magnitude saving in the amount of computation involved results from the use of the technique presented. In addition, a new purely automatic technique for arriving at threshold positions has been presented. The technique is based on an annealing approach and it has been shown to work well.

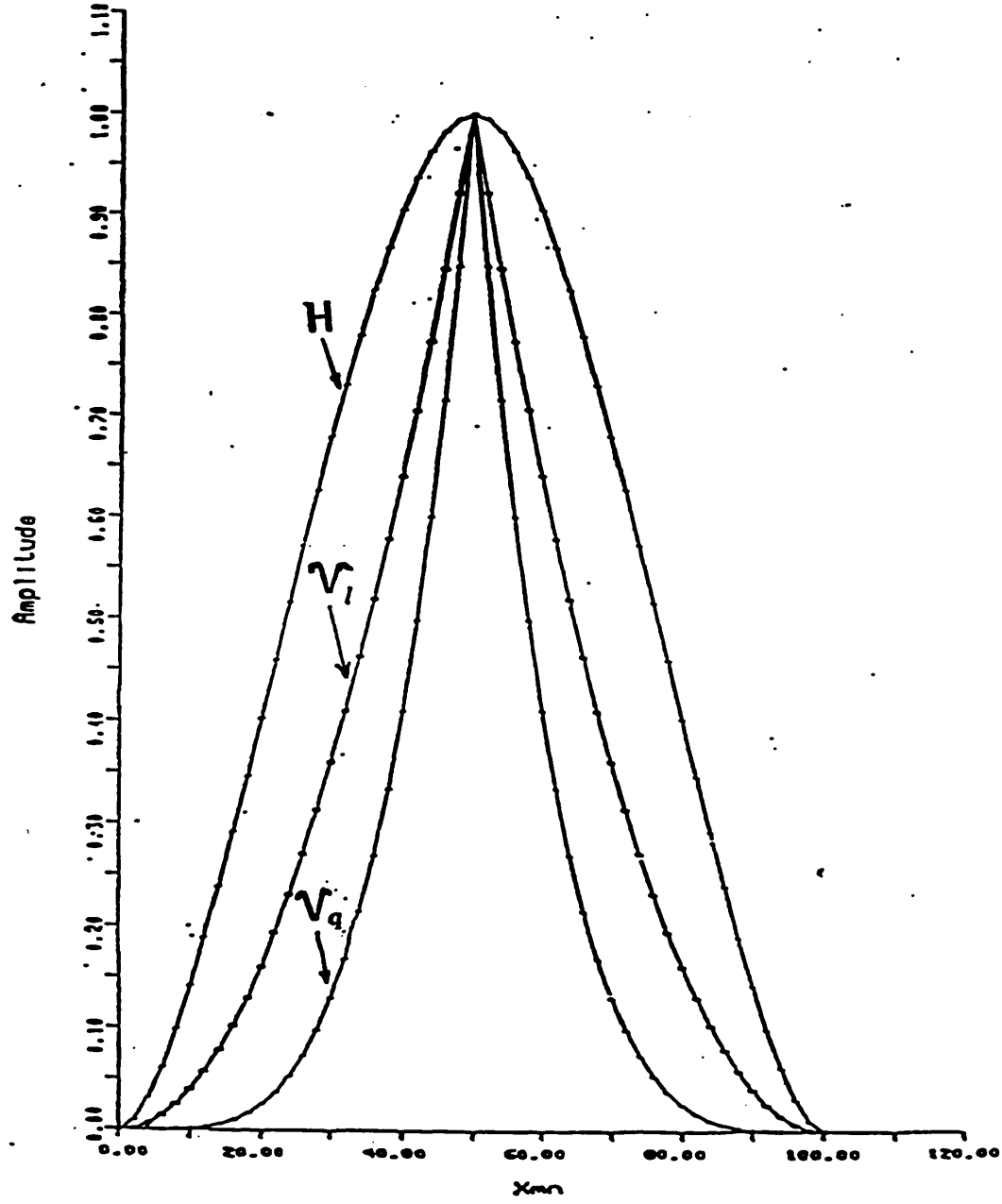


Fig 1
WINDOWING FUNCTIONS

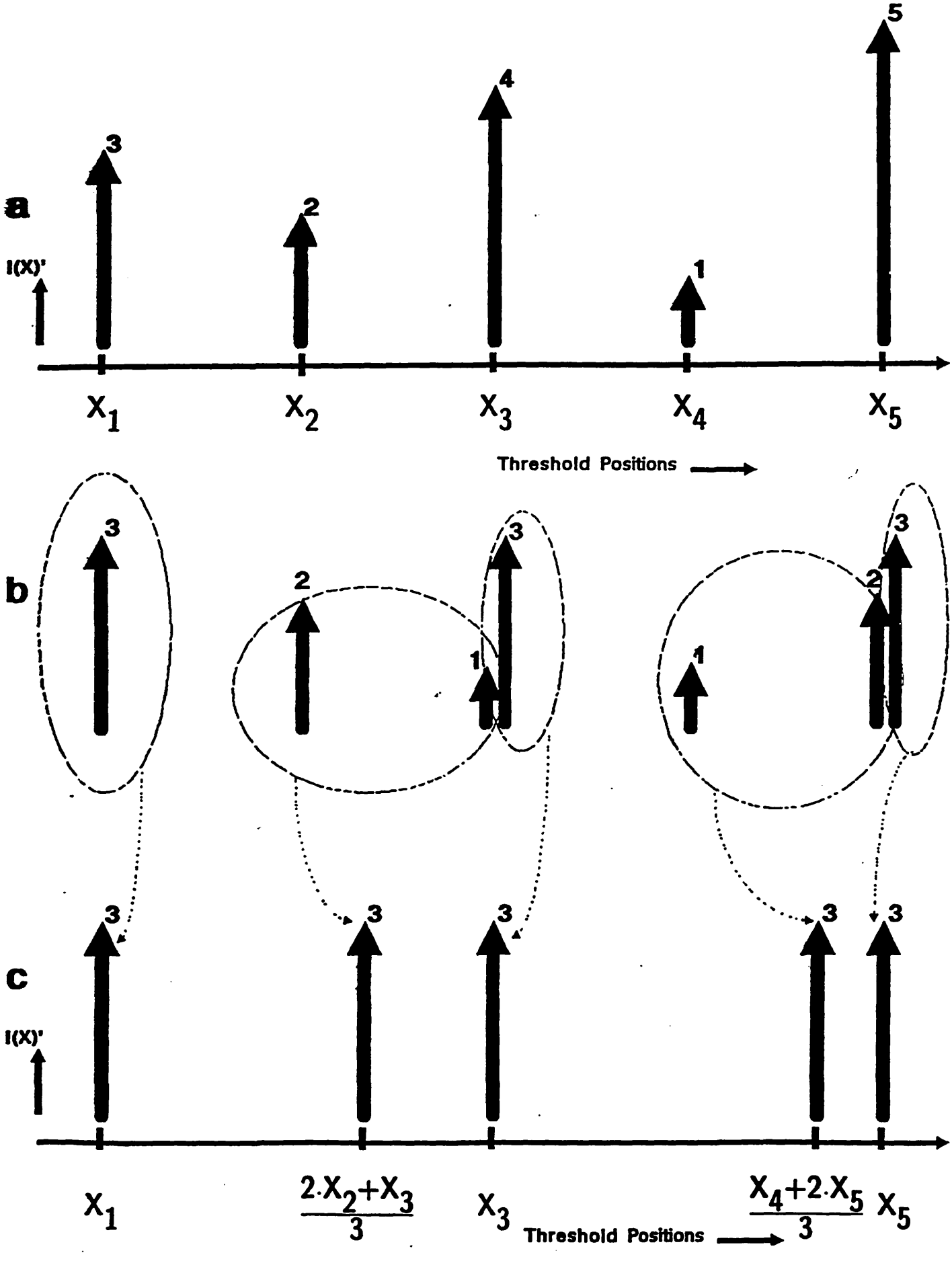


Fig 2
ILLUSTRATION OF STEP 3 OF THE ANNEALING TECHNIQUE

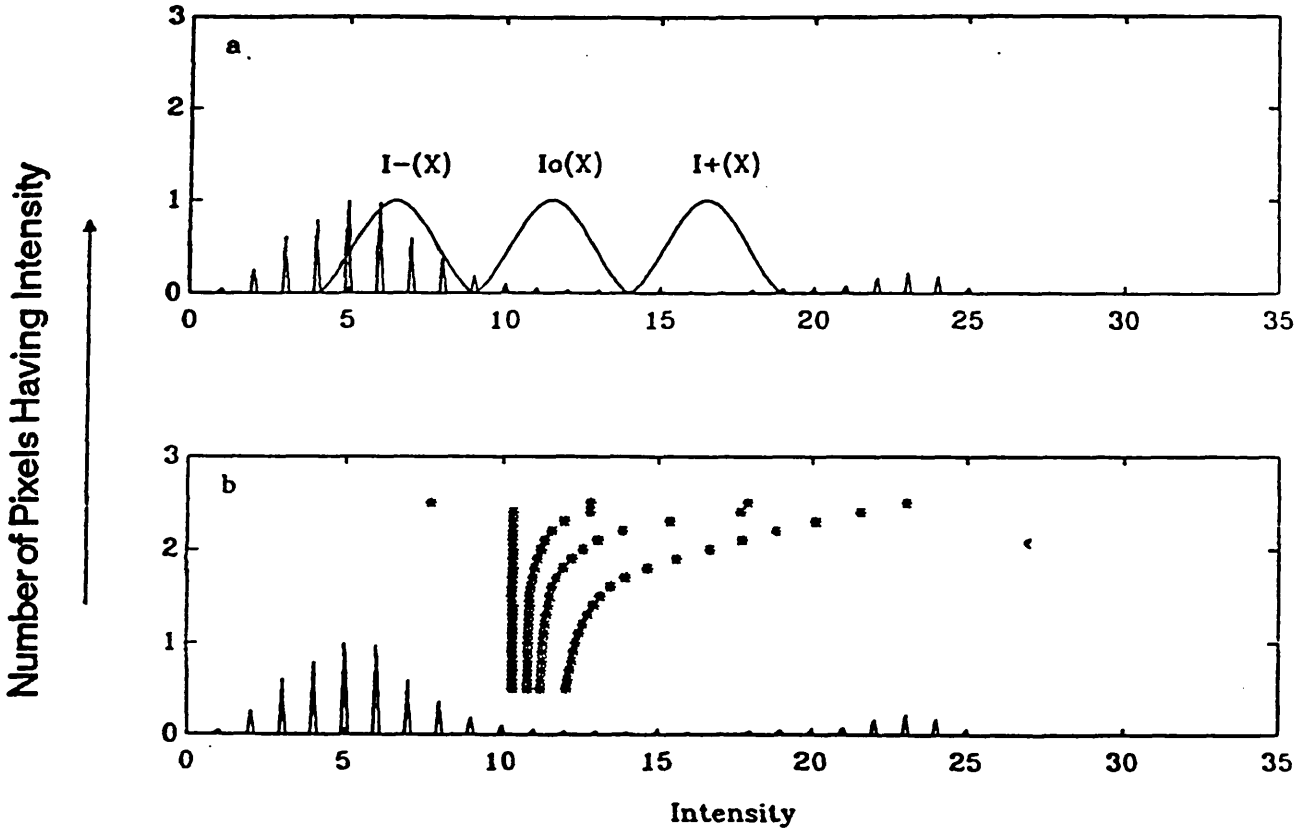
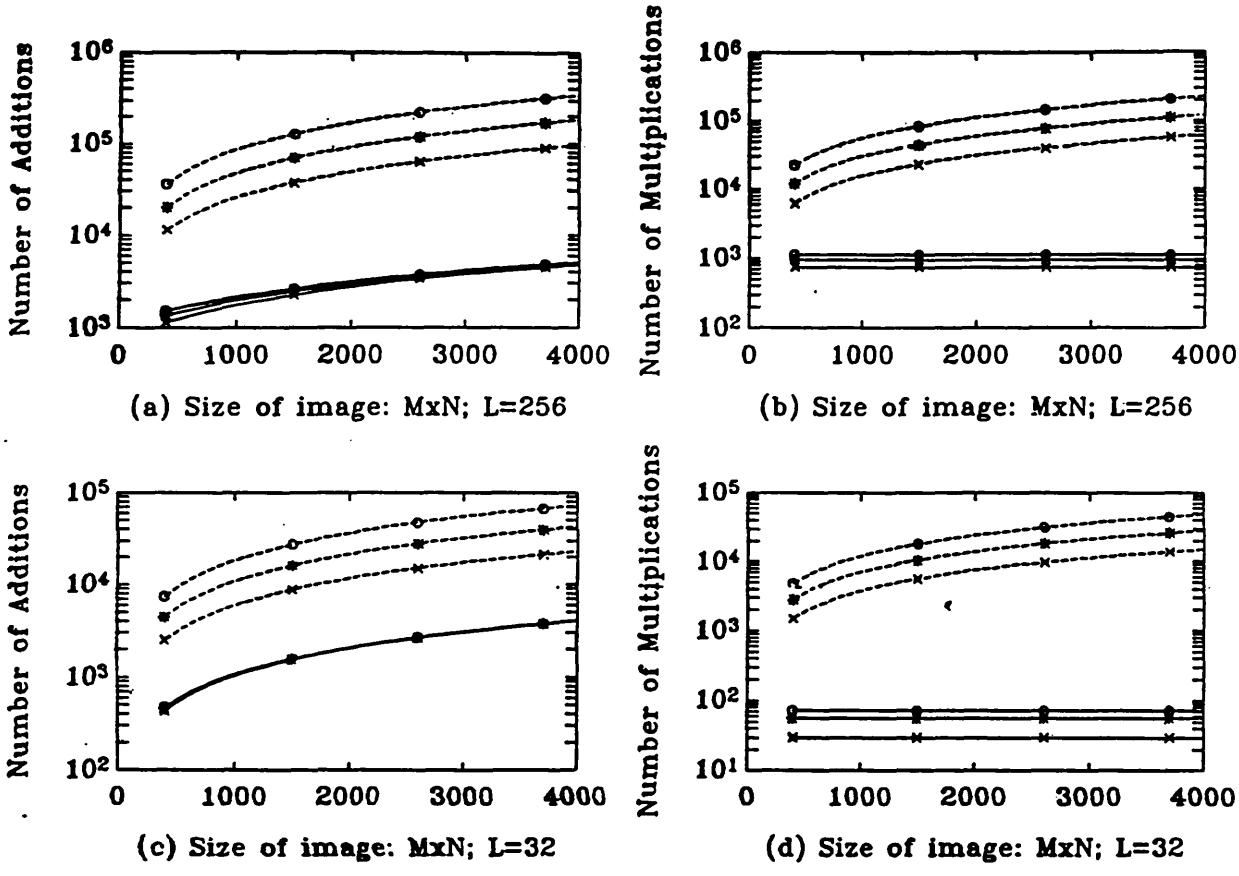


Fig 3
ANNEALING THRESHOLDING ON A BIMODAL HISTOGRAM



(a) Size of image: MxN; L=256

(b) Size of image: MxN; L=256

(c) Size of image: MxN; L=32

(d) Size of image: MxN; L=32

Fig 4

COMPARISON OF THE NUMBER OF COMPUTATIONS REQUIRED BY THE TECHNIQUE OF SECTION 2 (CONTINUOUS LINES) VS THAT OF REF [1] (DOTTED LINES)

MxN: SIZE OF IMAGE IN PIXELS

L : NUMBER OF LEVELS

δl: BANDWIDTH

IN (a) AND (b): o DENOTES δl=32, * DENOTES δl=16, x DENOTES δl=8

IN (c) AND (d): o DENOTES δl= 8, * DENOTES δl= 4, x DENOTES δl=2

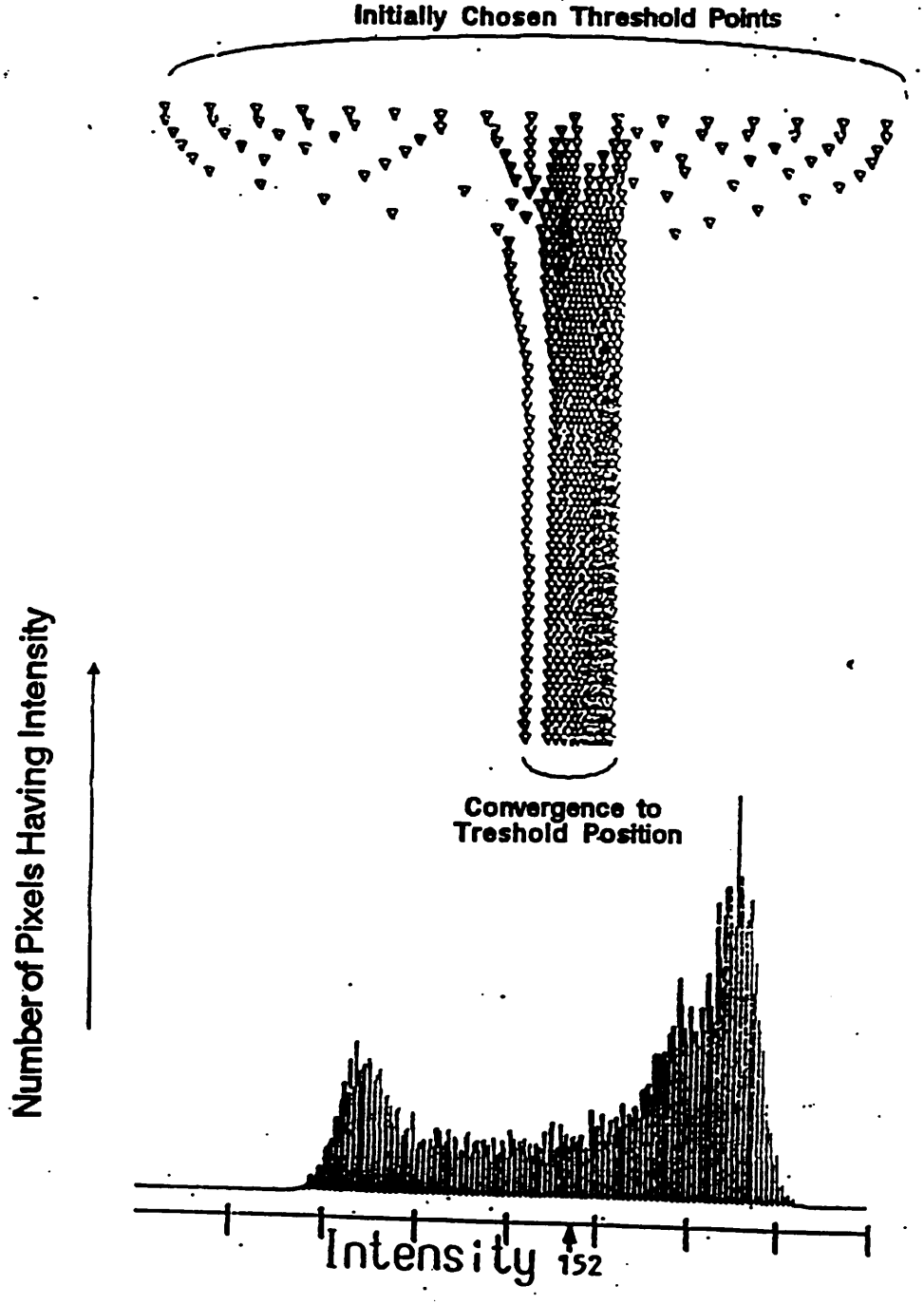


Fig 5
ANNEALING THRESHOLDING ON A NEWSPAPER IMAGE



a



c



b

Fig 6: ORIGINAL NEWSPAPER IMAGE (a) AND IMAGES AFTER THRESHOLDING AT THE MIDDLE OF THE RANGE(b) AND BY ANNEALING(c)

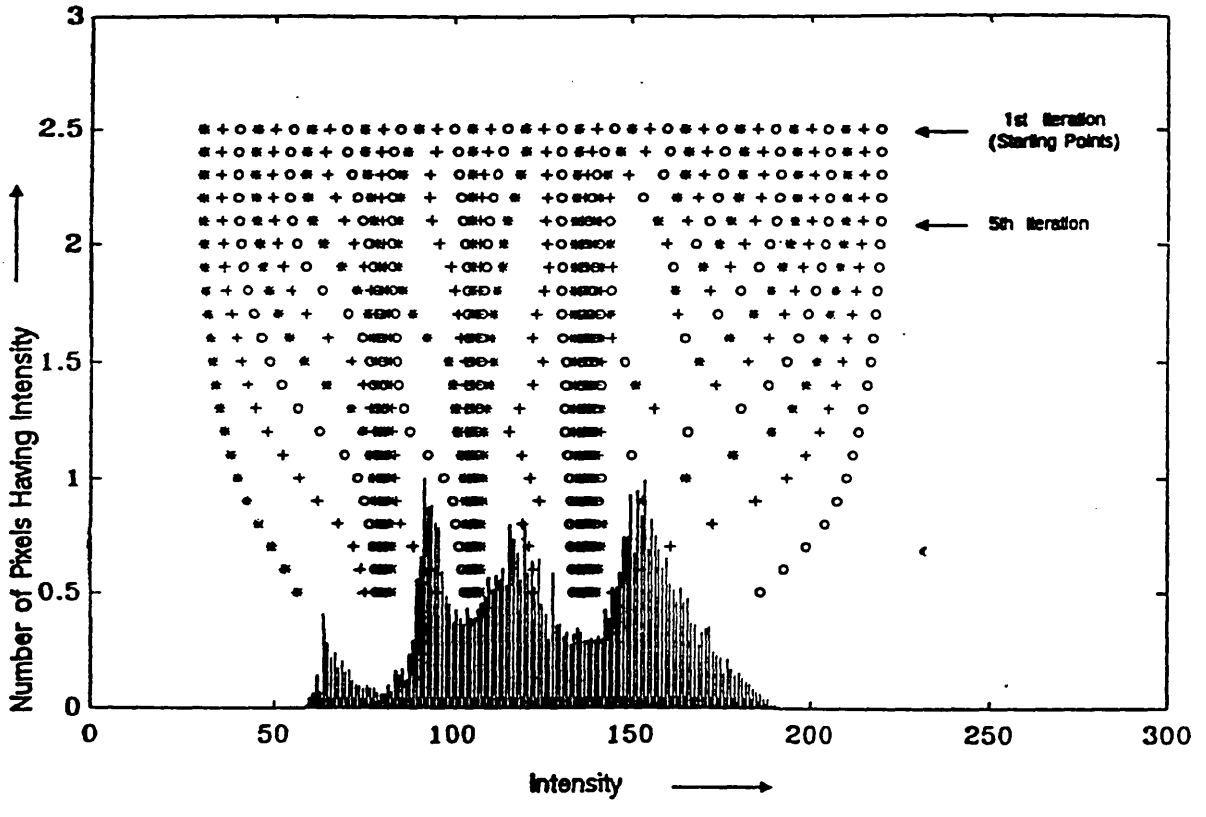


Fig 5
ANNEALING THRESHOLDING ON A MULTIMODE HISTOGRAM