



National Library of Canada

Cataloguing Branch  
Canadian Theses Division

Ottawa, Canada  
K1A 0N4

Bibliothèque nationale du Canada

Direction du catalogage  
Division des thèses canadiennes

## NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

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

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

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

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

THIS DISSERTATION  
HAS BEEN MICROFILMED  
EXACTLY AS RECEIVED

## AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de mauvaise qualité.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

LA THÈSE A ÉTÉ  
MICROFILMÉE TELLE QUE  
NOUS L'AVONS REÇUE

AUTOMATIC RECOGNITION OF NUMERALS VIA  
FOURIER SHAPE DESCRIPTORS  
AND  
BOUNDARY LINE ENCODINGS

Michael Tai-Yuen Lai

A Thesis  
in  
The Department  
of  
Computer Science

Presented in Partial Fulfillment of the Requirements  
for the degree of Master of Computer Science at  
Concordia University  
Montreal, Quebec, Canada

September 1978

© Michael Tai-Yuen Lai, 1978

## ABSTRACT

AUTOMATIC RECOGNITION OF NUMERALS VIA  
FOURIER SHAPE DESCRIPTORS  
AND  
BOUNDARY LINE ENCODINGS

Michael Tai-Yuen Lai

A character recognition system designed to recognize handprinted and machine-printed numerals is proposed.

Preprocessing techniques which perform filling, deleting, and linking operations have been developed. By making use of Fourier shape descriptors, the information generated from boundary line encodings, and the inside boundary information of each input character, a set of features has been extracted for analysis. Decision is achieved by a classification scheme based on discriminant analysis and nearest neighbor classification techniques.

A high recognition performance is confirmed by computer simulation and implementation of this proposed system on a CDC 7200 digital computer. Experimental results obtained from 9500 samples of 3 different data bases are briefly described.

## ACKNOWLEDGEMENTS

This research thesis was supported by the Department of Education of Quebec and was formatted by the DRAFT word-processing system at the university. I wish to express my gratitude to my supervisor, Dr. C. Y. Suen, for his valuable advice, suggestions and encouragement in my work and in preparation of this thesis. I wish to thank C. C. Kwan, Jeff Mulherin and Roger W. Y. Chan for preparing the data used in the experiments. I would like to acknowledge the excellent co-operation of the staffs in the computer center, and would like to thank the members in the Pattern Recognition and Man-Computer Communication Research Groups for their suggestions and comments.

## TABLE OF CONTENTS

	PAGE
TITLE PAGE .....	i
SIGNATURE PAGE .....	ii
ABSTRACT .....	iii
ACKNOWLEDGEMENTS .....	iv
TABLE OF CONTENTS .....	v
LIST OF FIGURES .....	vii
LIST OF TABLES .....	ix

## CHAPTER

1. INTRODUCTION	
1.1 Optical Character Recognition .....	1
1.2 Research in Optical Character Recognition ..	3
1.3 Basic Structure of the Proposed System .....	5
1.4 Scope of the Thesis .....	6
2. PREPROCESSING OF INPUT DATA	
2.1 Introduction .....	8
2.2 Input Data for Preprocessing .....	9
2.3 Preprocessing Algorithm .....	9
2.3.1 Deletion and Linking Operations .....	11
2.3.2 Filling Operation .....	13
2.4 Results of the Smoothing Operations .....	14

3.	FEATURE EXTRACTION	
3.1	Introduction	18
3.2	Contour (Curve) Tracing Algorithm	19
3.3	Fourier Shape Descriptors for a Polygonal Curve	25
3.4	Boundary Line Encodings	27
3.4.1	Directions and Direction Length Features	27
3.4.2	Curvature Features	29
3.5	Inside Boundary Features	33
3.5.1	Hole Detection Algorithm	35
3.6	Summary of Generated Features	39
4.	CLASSIFICATION PROCESS	
4.1	Introduction	40
4.2	Discriminant Analysis Technique	42
4.3	Nearest Neighbor Classification Method	46
4.4	Classification Scheme	48
5.	EXPERIMENTAL RESULTS	
5.1	Data Sets	53
5.2	Results from Recognition Experiments	57
6.	CONCLUSIONS AND DISCUSSIONS	
6.1	Conclusions	73
6.2	Discussions and Comments	73
	REFERENCES	76

## LIST OF FIGURES

FIGURE	PAGE
1.1 Major components of an Optical Character Recognition system .....	2
1.2 Basic structure of the proposed character recognition system .....	5
2.1 Samples of input digitized numerals .....	10
2.2 Points involved in smoothing operations .....	11
2.3 A 5x5 window used in the linking process .....	13
2.4a Sample handprinted numerals from IEEE data base #1.2.1 .....	15
2.4b Sample machine-printed numerals from OCR-A data base .....	16
3.1 Operation of the contour tracing algorithm .....	20
3.2 Some modifications in the contour tracing algorithm	22
3.3 Samples of input characters with their outer boundaries .....	23
3.4 Examples of rejected character from contour tracing algorithm .....	24
3.5 A polygonal boundary .....	26
3.6 Normalized starting point of characters .....	27
3.7 The Freeman code .....	28
3.8 Generated direction codes of sample characters ...	30
3.9 Examples of curvature information from boundary line encodings .....	32

3.10 The four quadrants of a character and its centre point .....	34
3.11 The position measurement grid .....	34
3.12 An illustration of the confused pairs of characters on the basis of their outer boundaries .....	36
3.13 Four divided regions in a character grid .....	38
4.1 A pattern classifier .....	41
4.2a Classification tree for decision-making .....	49
4.2b Classification tree for decision-making .....	50
4.2c Classification tree for decision-making .....	51
5.1 Detailed structure of the proposed character recognition system .....	55
5.2 Standard numeral models from different data bases ..	56
5.3 Misclassified and rejected samples of data set A ..	61
5.4a Misclassified and rejected samples of data set B ..	64
5.4b Misclassified and rejected samples of data set B ..	65
5.5 Rejected and misclassified samples of data set C ..	72



## LIST OF TABLES

TABLE	PAGE
5.1a Classification results from data set A (500 training samples -- classification was based on the highest score generated from the discriminant functions) .....	58
5.1b Results from data set A (500 training samples -- classification scheme was used) .....	59
5.1c Results from data set A (250 samples for training)	60
5.2a Classification results from data set B (3000 training samples) .....	63
5.2b Classification results of 3000 training samples from data set B (classification was based on the highest score generated from the discriminant functions) .....	66
5.2c Classification results of 3000 testing samples from data set B (classification was based on the highest score generated from the discriminant functions) .....	67
5.3a Classification results from data set C (1500 training samples) .....	70,
5.3b Classification results from data set C (1500 training samples, classification was based on the highest score generated from the discriminant functions) .....	71

# CHAPTER ONE

## INTRODUCTION

### 1.1 Optical Character Recognition

The concepts in Pattern Recognition have been increasingly recognized as an important factor in the design of modern automatic information systems. During the past two decades, extensive research and development related to this field has taken place, and the principles have been applied to different areas such as computer science, engineering, information science, biomedical and natural science, etc. Among the many subject areas in the field of pattern recognition, Optical Character Recognition has perhaps been the most popular and challenging subject which researchers are interested in.

Optical Character Recognition (OCR) system was first introduced in the 1950's as a replacement for keypunching, and it was promised to be the most practical means of direct source-data entry in modern information and data processing system. Optical readers, which process large volumes of data and provide the fastest means of data conversion at low errors rate, are making OCR a more widely acceptable means of data entry than other devices such as keypunches, key-to-tape and key-to-disk systems, etc.

An OCR system, such as the one shown in Figure 1.1, is

basically made up of three major subsystems : (1) the character-transformation device, (2) the information selector, and (3) the recognition logic. The transformation device scans the characters and converts them into a form suitable for further processing. The information selector performs the feature extraction operation which measures certain characteristic features of the characters. The recognition logic examines the derived features and determines the identity of the input character.

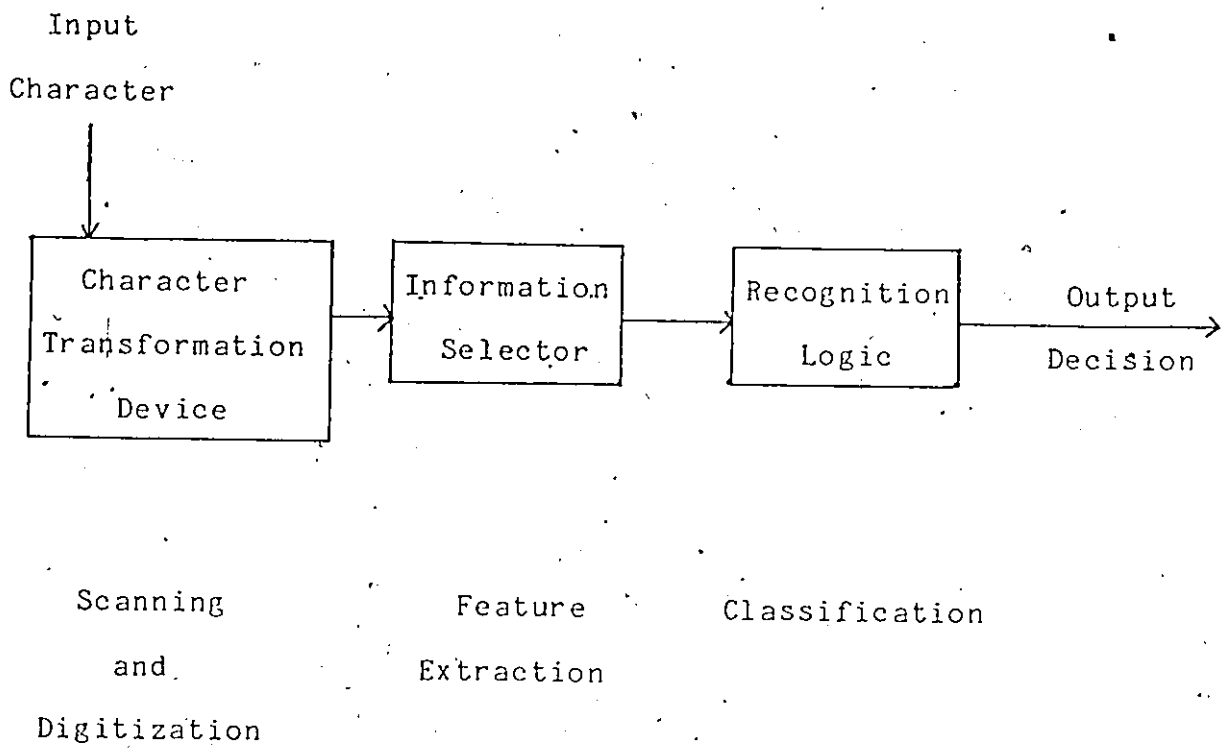


Figure 1.1 Major components of an Optical Character Recognition system.

Several books as well as some comprehensive

state-of-the-art reports on Optical Character Recognition are cited in [3], [6], [16], [18], [37], [47], [48], [49].

## 1.2 Research in Optical Character Recognition

Recent research efforts in Optical Character Recognition can be divided into two categories: (1) the recognition of machine-printed characters and (2) the recognition of handwritten characters.

For the first category, since the data to be recognized is essentially invariant, OCR machines which can read at high speed several common machine-printed character fonts with high degree of reliability have been developed for different applications [5], [7], [24]. The trend of these advances is toward the development of OCR machines which can handle a wide variety of applications on all types and sizes of input, and which can provide virtually unlimited font recognition at low cost.

The recognition of handwritten character, on the other hand, is still a difficult problem to solve. The data to be recognized can vary infinitely which cause many extraordinary problems. Great distortions in handwritten characters, such as the size and structural details which vary greatly from people to people, still pose a major problem to be solved. Several difficulties which affect the recognition of handwritten characters are briefly mentioned in [35].

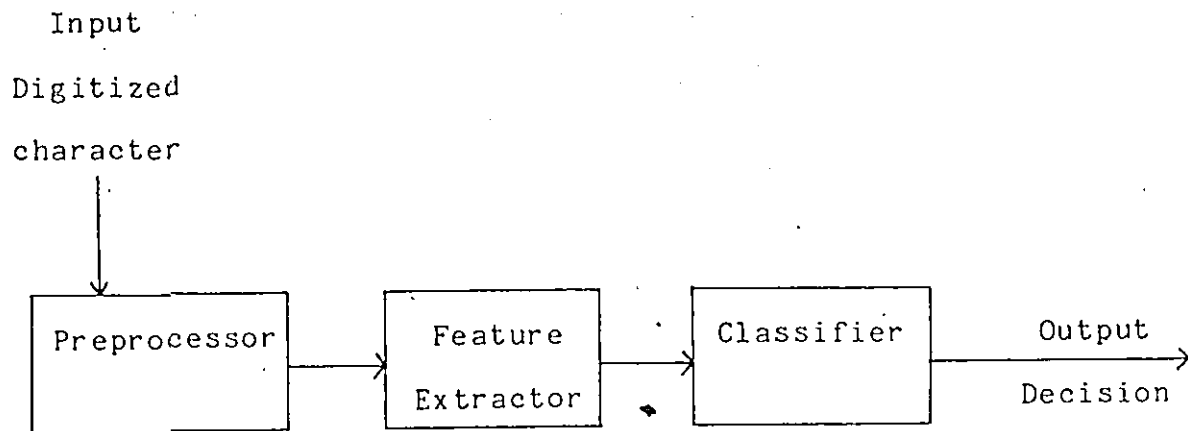
The problem of handwritten character recognition can also

be subdivided into (1) cursive script and (2) print script. Due to the difficulties and high costs in the segmentation of words into individual letter; it is not desirable to develop a system to automate the recognition of Roman cursive script. Several studies related to this work are discussed in [14], [15], [20]. Because of its wide applications [22], [24], [44] and less complexity than cursive script, automatic recognition of handprints (mostly numeric and alphanumeric characters) pertains a larger part of research in OCR. Extensive literatures concerning these advances are well reported in [27], [50].

Research in OCR has produced many different recognition techniques. The most commonly used recognition methods are image (template) matching, curve tracing, and stroke or feature analysis. In template matching, a prototype of each character is stored, and a measurement of the correlation or match between a specimen character and each stored prototype is performed. Curve tracing involves the tracing of the outline structure of a character and recognition is based on the features generated from it. Feature analysis technique is based upon the detection of a group of characteristic properties such as horizontal lines, vertical lines, loops, curves, junctions, bays, line ends, corners, and crossover which, when appropriately combined, will describe each character class. Each input character is then matched against a "table" representing each reference character class. Miscellaneous techniques which were employed in OCR have also been reviewed by Harmon [27].

### 1.3 Basic Structure of the Proposed System

The proposed character recognition system is shown diagrammatically in Figure 1.2. The function of this system is to detect and extract common features from the input character, and to recognize and classify this character as a member of one of the character classes.



Preprocessing:	Feature	Classification
Smoothing	Extraction	
Operations		

Figure 1.2 Basic structure of the proposed character recognition system.

After the input character was scanned by an input device, namely, a high-speed optical scanner, it was transformed into a

digital representation and was put into the system. The digitized character is then fed into the preprocessing stage where the preprocessor performs the filling, deleting and linking operations. The preprocessed image enters the feature extraction stage where the measurements of its characteristics, called features, are processed by the feature extractor. The classifier, which operates on the feature vectors, will determine to which of the several classes the character belongs.

#### 1.4 Scope of the Thesis

The work in this thesis is confined to the scope of recognition of handprinted and machine-printed numerals '0' to '9'.

Chapter 1 gives a general description in Optical Character Recognition, and the research involved in it. Basic structure of a proposed character recognition system has been described.

Preprocessing techniques which smooth the input data, and the extraction of features such as Fourier shape descriptors and information generated from inner and outer boundaries of the character are briefly described in chapter 2 and chapter 3 respectively.

The use of discriminant-analysis-based technique and the nearest neighbor methods for classification purposes is presented in chapter 4. A classification scheme for decision making is mentioned in detail.

In chapter 5, experimental results from simulation of the proposed system on a digital computer to measure its performance is briefly described.

In the final chapter, conclusions and suggestions for further study will be discussed.



## CHAPTER TWO

## PREPROCESSING OF INPUT DATA

## 2.1 Introduction

In most recognition systems, preprocessing is often required to smooth irregularities, to remove noises, and to remove redundancies in the input patterns, so that effective patterns can be provided before the measurements of their characteristics.

Different approaches in the preprocessing of input data have been developed and tried by researchers. Unger [56], Dinneen [12], Rao [42], and Freyer and Richmond [19] have suggested different algorithms for the smoothing of data. Stefanelli and Rosenfeld [46], Deutsch [10] and Triendl [52] have suggested algorithms for thinning and skeletonization of noisy input data. Hussian, Toussaint and Donaldson have also described the size-normalization preprocessing algorithm in [28].

In order to reduce the classification errors in a recognition system, Gudesen [26] also performed a quantitative analysis of different preprocessing techniques for the recognition of handprinted characters.

In this chapter, we are mainly concerned with a

preprocessing algorithm used in the proposed recognition system which perform the filling, deleting and linking operations in each input character.

## 2.2 Input Data for Preprocessing

Digitized character representations in the form of binary ( black and white ) matrices, called 'character grids', are fed into the computer for preprocessing. Figure 2.1 shows some sample input numerals in which the black points ( -1's ) represent the elements of the character and the white points ( blanks ) represent the background.

## 2.3 Preprocessing Algorithm

This algorithm is basically a filling and deleting process which includes the following functions :

- 1) to eliminate isolated points,
- 2) to eliminate small bumps along straight line segments,
- 3) to link small breaks or broken lines,
- 4) to fill in small notches along straight line segments,
- 5) to fill in isolated holes,
- 6) to replace missing corner points if certain conditions are satisfied.

In performing the algorithm, all the points in both the original input character matrix and the resulting ( new ) output character matrix are involved in the process. Basically, a 3x3



submatrix, called 'window', is considered for operation, and the changing of the value at the centre point of the window is dependent upon the value of its neighbors. Figure 2.2(a) shows a point  $x(ij)$  and its eight neighbors in the original image of the input character, where  $i$  and  $j$  stand for the  $i$ th row and  $j$ th column of the matrix respectively. Figure 2.2(b) shows a point  $X(ij)$  and its eight neighbors in the resulting image.

a	b	c
d	$x(ij)$	e
f	g	h

A	B	C
D	$X(ij)$	E
F	G	H

(a) 3x3 window points in original input image

(b) 3x3 window points in resulting output image

Figure 2.2 Points involved in smoothing operations.

Before the start of the preprocessing process, all the points in the resulting image are assigned equal to the corresponding position points in the original image. We have to note that all the filling and deleting of points will be done in the resulting image, but no deletion or filling of points in the original image.

### 2.3.1 Deletion and Linking Operations

The first step in the preprocessing operation is the deletion and linking processes which will accomplish functions 1) to 3) of the smoothing algorithm.

Every point in the input matrix has been scanned. If the point  $x(ij)$  is black, then the point  $X(ij)$  will become a white point when either one of the following conditions is satisfied :

- 1) If all the eight neighbors (points a, b, c, d, e, f, g, h) of  $x(ij)$  are white,  $X(ij)$  will be white.
- 2) We count the number of black points in the eight neighbors of  $x(ij)$  and then compare the total with a threshold (a threshold of 2 is set for this condition). If the count is equal to the threshold, then  $X(ij)$  will be white when either
  - a) Points b, c, e, g, h are white and either point a or point f is white, or
  - b) Points a, b, d, f, g are white and either point c or point h is white, or
  - c) Points a, b, c, d, e are white and either point f or point h is white, or
  - d) Point d, e, f, g, h are white and either point a or point c is white.
- 3) If the count for the number of black points in the eight neighbors of  $x(ij)$  is equal to 1 or 3, and either points b, c, e, g, h, or points a, b, d, f, g, or points a, b, c, d, e, or points d, e, g, h are white.

On the other hand, if the point  $x(ij)$  is white and either

- a) points d, e are black and points b, g are white, or b) points b, g are black and points d, e are white, or c) points a, h are black and points c, f are white, or d) points c, f are black and points a, h are white, then we have to count the number of black points in all the neighbors of  $X(ij)$  in the 5x5 window (as shown

in Figure 2.3). If the count is less than or equal to the threshold (a threshold equal to 10 is set), then the element  $X(ij)$  will be assigned black, otherwise,  $X(ij)$  will be white. Since this procedure and the deleting process are performed together, some elements may be eliminated before the assignment of black point to  $X(ij)$  is done, we have to examine the eight neighbors of  $x(ij)$ , and to assign the values back to the elements in the corresponding position of the resulting image if necessary. This process will accomplish function 3).

	A	B	C	
	D	$X(ij)$	E	
	F	G	H	

Figure 2.3 A 5x5 window used in the linking process.

### 2.3.2 Filling Operation

After the deleting and linking operations, all the elements in the original image are assigned equal to the corresponding position elements in the resulting image, and both images are then used in the filling process.

The same 3x3 windows are used in the operation. If the element  $x(ij)$  is white, the filling operation can then be performed. Function 4) can be accomplished when one of the following conditions is valid :

- 1) Points A, B, C, D, E are black and points F, G, H are white.
- 2) Points D, E, F, G, H are black and points A, B, C are white.
- 3) Points A, B, D, F, G are black and points C, E, H are white.
- 4) Points B, C, E, G, H are black and points A, D, F are white.

Function 5) can be accomplished if points B, D, G, E are all black.

If the number of black points in the eight neighbors of  $x(ij)$  is less than or equal to 5 (a threshold), function 6) can be accomplished when one of the following conditions holds :

- 1) Points a, b, d are black, and points G, H are white.
- 2) Points b, c, e are black, and points F, G are white.
- 3) Points d, f, g are black, and points B, C are white.
- 4) Points e, g, h are black, and points A, B are white.

#### 2.4 Results of the Smoothing Operations

Several sample characters in their original and preprocessed forms are shown in Figure 2.4 (a) and (b).

111  
   1111111  
  111   11  
          11  
        11  
       11  
      11  
     11111  
       111  
          11  
           11  
          11  
          11  
        111  
       111  
      111111  
      111

Original input numeral '3'

   111111,  
  11111111  
  111   111  
          111  
        1111  
       1111  
      11  
     11111  
       11111  
          1111  
           111  
          111  
          1111  
        1111  
       11111  
      111111  
      111

Preprocessed numeral '3'

      1111  
   1111 111  
  11   11  
  1      1  
  1  
  11          11  
  1          11  
  11          1  
  11          1  
  11          11  
  111          1  
  111111  
   111111  
    1      111  
    1      11  
   11      11  
   11      11  
   1      11  
   1          11  
  11      1  
  1      1  
  1      1  
  11      11  
  1111111111  
  11111

Original input numeral '8'

      111111  
   1111111111  
  111   111  
  11   11  
  11          1  
  11          11  
  111          11  
  11          11  
  11          11  
  11          11  
  11111      11  
  111111111  1  
   111111  
    11   1111  
    1      111  
   111      11  
   111      11  
   11      11  
   11          11  
   11          1  
   11          11  
  111      111  
  11111111111  
  1111111

Preprocessed numeral '8'

Figure 2.4 (a) Sample handprinted numerals from IEEE data base #1.2.1.



```

1111111111111111
1111111111111111
11111111 11111111
111      111
111      111
111      111
111      111
111      111
1111     111
11      111
111      111
1111     111
111      111
111      111
11      111
111      111
1111     111
1111     111
1111111111111111
111111111111

```

```

1111111111111111
1111111111111111
1111111111111111
1111      1111
111      111
111      111
111      111
111      111
111      111
111      111
111      111
111      111
111      111
111      111
111      111
1111     111
1111     111
1111     1111
1111111111111111
11111111111111

```

Original input numeral '0'

Preprocessed numeral '0'

```

1111111111111111
1111111111111111
111111111 11111111
111      111
11      111
111      111
1111111111111111
1111111111111111
1111111111
111
111
111
111
111
111
111
111

```

```

1111111111111111
1111111111111111
1111111111111111
1111      1111
111      111
1111     1111
1111111111111111
1111111111111111
111111111111
111
111
111
111
111
111
111
111

```

Original input numeral '9'

Preprocessed numeral '9'

Figure 2.4 (b) Sample machine-printed numerals from OCR-A data base.

We have to note that the linking process which closes breaks in lines is applicable only when the gaps are no wider than one unit point in the digitized image. The thresholds, which were set equal to certain value in each particular condition, are chosen after several trials on some sample data which give favourable appearance. And, we also noted that all the thresholds are held constant over the entire smoothing operation.

## CHAPTER THREE

## FEATURE EXTRACTION

## 3.1 Introduction

In order to simplify the task of a recognition system, feature extraction is often required to reduce the amount of data which is presented to it. The purpose of the feature extraction process is to generate from the input pattern an n-dimensional vector or feature vector which captures the essential characteristics or properties of the pattern to be recognized.

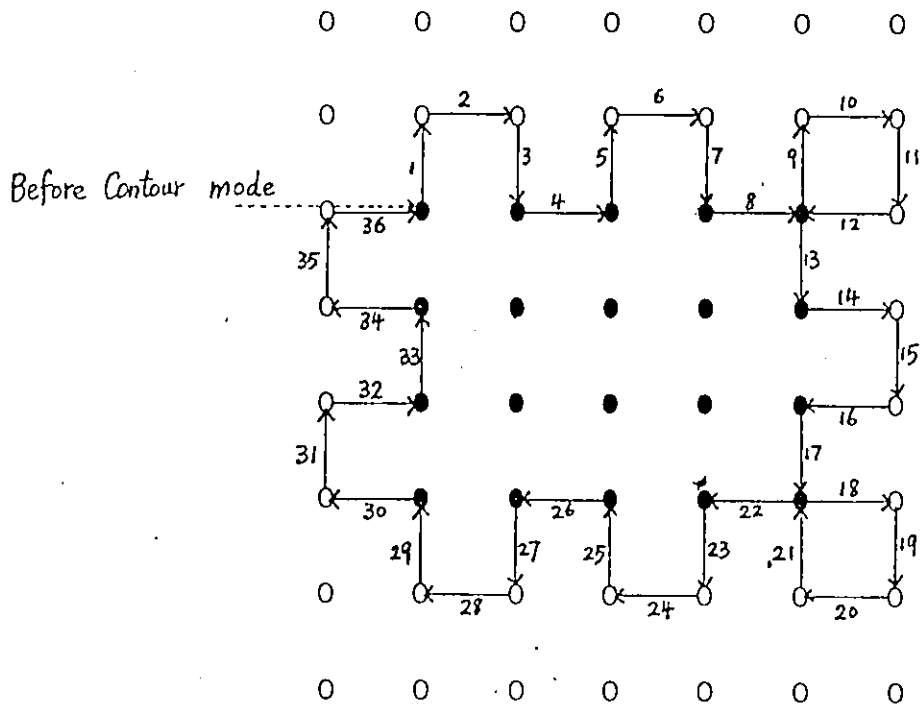
Since the boundary of a pattern can be represented by a set of closed contour curves or line segments which vary according to the shape of the pattern, a sequential trace of a character's boundary can yield useful information for distinguishing one character from another. Besides the various techniques mentioned in section 1.2, shape analysis and transformation techniques such as Fourier analysis of boundary information [ 4], [23], [41], [43], [51], [57], boundary line encoding [17], [36], [55], and polygonal boundary approximations [39], [40] have also been actively explored and applied to the recognition of characters. These techniques which focus our attention on methods which produce compact descriptions of curves have demonstrated good recognition performance.

In this chapter, we are primarily concerned with the use of the above shape analysis techniques in the proposed system. Investigations in contour tracing algorithm, Fourier shape descriptors, and the inner and outer boundary information of each character are described in detail.

### 3.2 Contour (Curve) Tracing Algorithm

The first step in any shape analysis method usually deals with the tracing of the outer boundary of each input pattern. Several contour tracing algorithms have been developed and suggested by different researchers [ 8], [25], and the one which we have developed was similar to Toussaint's [54]. The contour tracing algorithm is outlined as follows :

A scanning spot is initiated which moves point by point, from the leftmost column of the first row to the rightmost column of the grid (matrix), and this procedure is repeated on the row which is immediately following the previous scanned row until the first black point is encountered. Upon locating this point, the scanner enters the CONTOUR mode, in which it moves right after encountering a white point and left after encountering a black point. The movement of left and right is relative to the spot direction. The CONTOUR mode terminates when the scanning spot completes its trace around the outside boundary of a character and returns to its starting point. The operation of this algorithm is illustrated in Figure 3.1.



- -- represents the elements of the character
- 0 -- represents the background elements

Figure 3.1 Operation of the contour tracing algorithm.

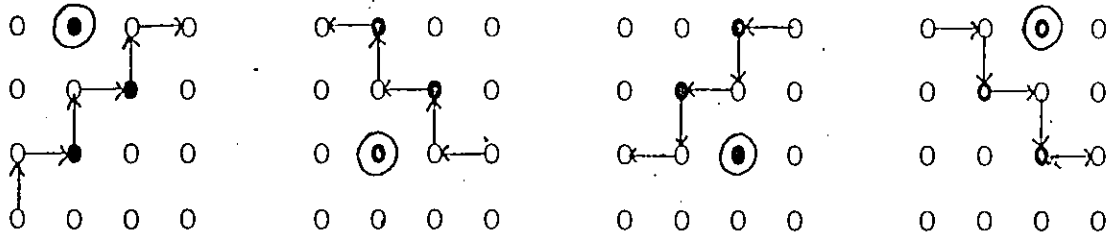
One drawback of some contour tracing algorithms is that they do not perform perfectly well especially when the outline structure of a character is thin and irregular. Our contour tracing algorithm overcomes the above drawbacks by considering the following conditions :

a) After entering the CONTOUR mode, when the first black point is scanned twice and either the number of black points scanned is equal to one, or other scanned black points are

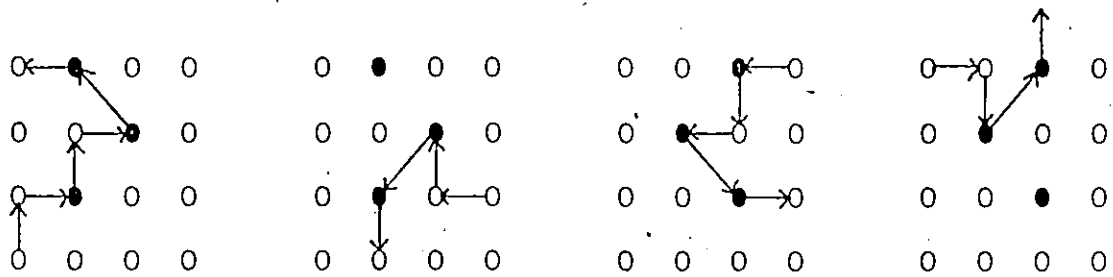
located in the same row as the first point, we have to continue the mode rather than to terminate it. The point which is located in the successive row but the same column as the first point will be processed. If this point is white, the scanning spot will move to the left when there is a black point, and move to the right otherwise. Since the linking operation and the process for elimination of isolated point have been performed during the preprocessing stage, there might be at least a point connected with the first black point in the successive row, otherwise, the input character will be rejected due to severe distortion.

b) Figure 3.2a is shown the circled points which were skipped by the routine movements of the contour tracing algorithm. In our contour tracing scheme, these difficulties can be overcome by making a forty-five degree move of the scanning spot (as shown in Figure 3.2b) after either one of the conditions has been examined. Before the movement of forty-five degree, we have to note down to which direction the scanning spot is moving. After the move, the scanning spot will move in the way which is opposite to this recorded direction, and the tracing mode can continue as normal.

During the contour mode, all the x-y coordinates of the black edge points are recorded, and by examining the changes in the x-y coordinates of each successive pair of edge points, the vertices in the outer boundary of each character can be located. Figure 3.3 is shown some input characters and their outer boundaries after contour tracing, asterisks '\*' represent the boundary vertices.



a) Points (circled) not traced by the routine process of the contour tracing algorithm.



b) Solutions for the conditions depicted in a).

Figure 3.2 Some modifications in the contour tracing algorithm.

```

11111
11111111
111 111
  111
  1111
  1111
  11
  11111
  11111
    1111
    111
    111
    1111
    1111
  111111
1111111
111

```

```

*111*
** ** *
*1* * 1
  * 1
  1 *
  * **
  1*
  * *1*
  ** 1
    1 *
    * 1
    * 1
    1 *
    1 *
  ** *1*
  *1*

```

Preprocessed numeral '3' and it's outer boundary

```

11111
11111
11111
11111
1111
1111
1111
1111
1111
1111
1111
1111
1111
1111
11111111111111
1111111111111111
1111111111111111
1111111111111111
111 111
111 111
111 111
1111 1111
1111111111111111
1111111111111111

```

```

*111*
1 1
1 1
1 *
1 *
1 1
1 1
1 1
* *
* 1
1 *1111111111*
1 1
1 1
1 1
1 1
1 1
1 1
1 1
* 1
*111111111111*

```

Preprocessed numeral '6' and it's outer boundary

Figure 3.3 Samples of input characters with their outer boundaries.



We also noted that the contour tracing technique is especially sensitive to patterns which are distorted or broken. After the contour mode, the scanning spot has to process the remaining row and column points in the grid which have not been scanned. If a black point is encountered, the input character will be rejected as shown in Figure 3.4. In other words, an input numeral which contains broken lines is not permitted to enter our system for recognition.

	1	
	11	
	11	
	11	1111
1111111111111111		1111
1111111111111111		1111
1111111111111111		1111
1111111111111111		1111
	1111	111
	111	111
	11	111
	11	111
	11	111
	111	111
	111	111
	111	111
	1111	111
	11111	111
111111111111		111
1111111111		111
1111111111		111
1111111111		111
1111111111		111
	111	111
		111
	111	111
	1111	111
	111	111
	111	111
	111	111
	111	111
	111	111
	1111	111
11111111	11111	111
1111111111111111		111
1111111111111111		111
1111111111111111		111
	1111	

Figure 3.4 Examples of rejected character from contour tracing algorithm.

### 3.3 Fourier Shape Descriptors for a Polygonal Curve

Most boundary transform techniques involved the Fourier transform of the boundary which can be expressed in terms of 1) tangent angle versus arc length, or 2) as complex function to denote a parametric representation of the boundary coordinates [38]. The Fourier shape descriptors which were employed in the system belong to the second category, and they were first proposed by Granlund [23] but later developed by Persoon and Fu [41]. The shape descriptors are expressed as follows :

$$a_n = \frac{1}{L} \int_0^L u(t) e^{-j(2\pi/L)nt} dt, \quad (3.3.1)$$

where  $u(t) =$  the complex function  $x(t) + jy(t)$ , which denote a parametric representation of the boundary coordinates,

$t =$  the arc length of a simple clockwise oriented closed curve, and

$L =$  the perimeter of the closed curve.

Figure 3.5 is shown a polygonal boundary in which  $V_0$  is the starting point. A discrete formula for computing the FD's of equation (3.3.1) in case the curve is polygonal is expressed as follows :

$$a_n = \frac{1}{L(n2\pi/L)^2} \sum_{k=1}^m (b_{k-1} - b_k) e^{-jn(2\pi/L)t_k} \quad (3.3.2)$$

where  $t_k = \sum_{i=1}^k |V_i - V_{i-1}|$  for  $k > 0$  and  $t_0 = 0$ ,

and

$$b_K = \frac{V_{K+1} - V_K}{|V_{K+1} - V_K|}$$

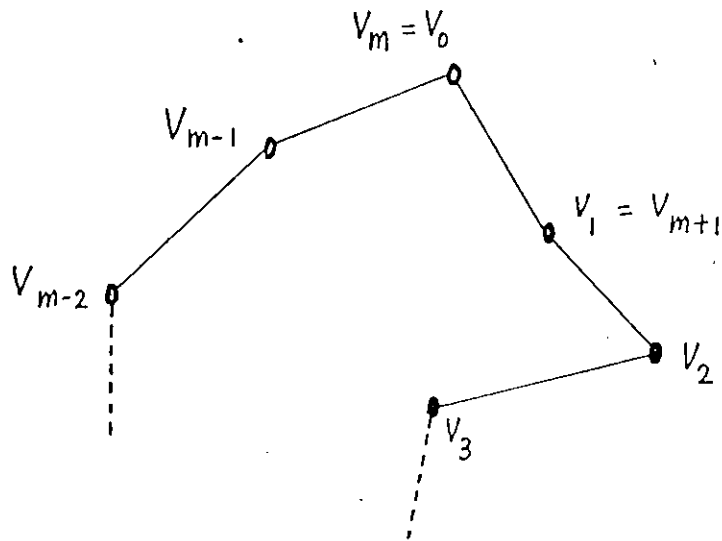


Figure 3.5 A polygonal boundary.

The moduli of the computed FD's are used as the characteristic measurements or features of each input character, and ten harmonics have been generated.

As mentioned by Zahn and Roskies [57], when two closed curves which differ only in position, orientation, and size with analogous starting points are transformed, they have identical Fourier descriptors. Therefore it is useful to normalize the starting point of each character. This normalization can be done when the contour tracing of the outer boundary is performed. It takes the first scanned black point as the starting point of the boundary as shown in Figure 3.6.

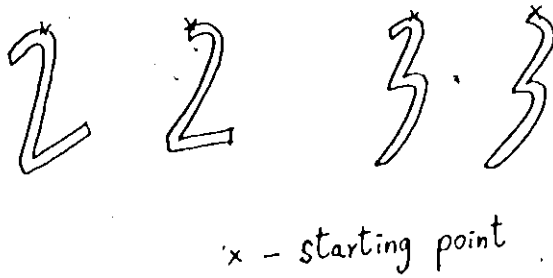


Figure 3.6 Normalized starting point of characters.

### 3.4 Boundary Line Encodings

Besides the Fourier analysis, boundary line encodings has also been a very challenging technique used for shape descriptions [38]. In the proposed system, a boundary line encoding approach has been developed for feature extraction. This method will mostly emphasize on the use of boundary vertices to generate directions, direction lengths, and curvature information of each input character.

#### 3.4.1 Directions and Direction Length Features

Freeman [17] has suggested eight basic direction codes, the Freeman code, for encoding a boundary. In his approach, each segment of the boundary curve which falls within one of the squares of a rectangular grid is approximated by one of the eight directions as shown in Figure 3.7. By making use of the boundary vertices, a digitized boundary can be expressed as a sequence  $[d_i]$  where  $d_i$  is a code for the direction from vertices

$(x_i, y_i)$  to  $(x_{i+1}, y_{i+1})$ . Eight possible direction codes which are generated from two successive vertices can be defined as follows

- 1) 'T' -- direction pointing upward.
- 2) 'B' -- direction pointing downward.
- 3) 'L' -- direction pointing left.
- 4) 'R' -- direction pointing right.
- 5) '1' -- direction pointing between 'T' and 'R'.
- 6) '2' -- direction pointing between 'R' and 'B'.
- 7) '3' -- direction pointing between 'B' and 'L'.
- 8) '4' -- direction pointing between 'L' and 'T'.

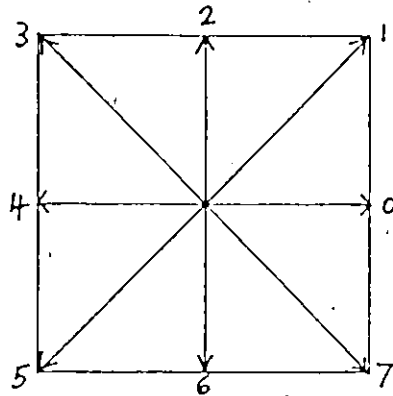


Figure 3.7 The Freeman code.

The direction length feature,  $f_i$ , which measures the distance between two successive vertices  $(x_i, y_i)$  and  $(x_{i+1}, y_{i+1})$  for each generated direction can be computed from the following equation :

$$f_i = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \quad (3.4.1)$$

We have to note that in our scheme the direction code and

direction length features for each character are represented by the percentage occurrences with respect to the total number of generated direction codes and the perimeter of the boundary respectively. There are totally sixteen generated features (eight direction features, eight direction length features) for each character. Examples of the generated direction codes of a character is shown in Figure 3.8.

### 3.4.2 Curvature Features

In our contour tracing process, the polygonal boundary of a character is scanned in a clockwise fashion. By considering each element and its successor in the direction chain (the sequence of the boundary), two main types of curvature information can be generated. They are 1) concave features, which are generated from the outside angle between two successive direction codes in the direction chain. This approximated angle should be greater than  $0^\circ$  but less than  $180^\circ$ , and 2) convex features, which are also generated from the outside angle between two successive direction codes, but the approximated angle should be greater than  $180^\circ$ .

The two types of curvature information, with each subdivided into two groups, can be represented by two successive direction codes as follows:

1) Concave feature type --

a) Direction codes starting with 'T', 'B', 'L', 'R' --

'T' : T3, TL, T4.

'B' : B1, BR, B2.

```

*1111*
** ** *
*1* * 1
  * 1
    1 *
  * **
    1*
  * *1*
    ** 1
      1 *
        * 1
          * 1
            1 *
              1 *
                *1*
                  *1*

```

Direction codes : R2B3L3R2B3B3L3LTR1T4L4T1T4L3LTR1

```

*111111111111*
1 *1111111*
1 1
1 *
1 1
1 1
1 *
1 1
1 *1111111*
*1111111* 1
  1 1
    * 1
      1 1
        1 1
          * 1
            1 1
              *1111111* 1
                *111111111111*

```

Direction codes : RBL3B2RBLTR1T4LT

Figure 3.8. Generated direction codes of sample characters.

'L' : L2, LB, L3.

'R' : R4, RT, R1.

b) Direction codes starting with '1', '2', '3', '4' --

'1' : 1L, 1T, 14.

'2' : 2T, 21, 2R.

'3' : 3R, 32, 3B.

'4' : 4L, 43, 4B.

2) Convex feature type --

a) Direction codes starting with 'T', 'B', 'L', 'R' --

'T' : T1, TR, T2.

'B' : B4, BL, B3.

'L' : L4, LT, L1.

'R' : RB, R3, R2.

b) Direction codes starting with '1', '2', '3', '4' --

'1' : 1R, 12, 1B.

'2' : 2B, 2L, 23.

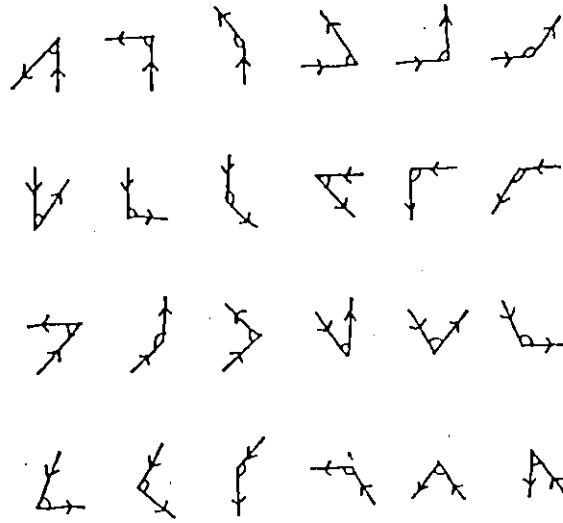
'3' : 3L, 34, 3T.

'4' : 4T, 41, 4R.

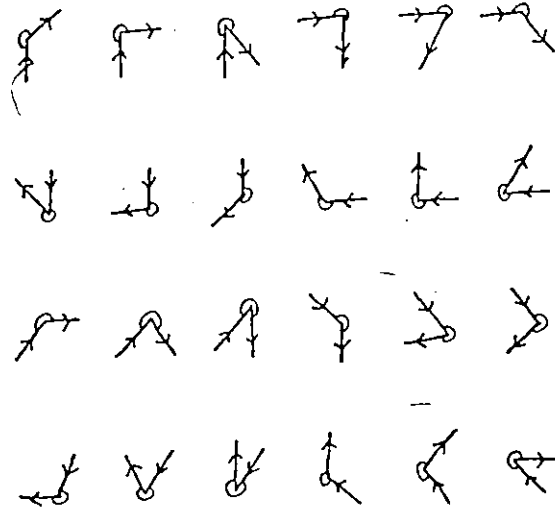
Figure 3.9 is shown some examples of curvature information.

By comparing the x-y coordinates of all the boundary vertices, we can obtain the topmost, leftmost, rightmost, and bottommost coordinates of a character, and hence its true size (width and height) can be defined. And, by dividing the height and width into equal halves, a centre point ( $C_x$ ,  $C_y$ ) and the four quadrants of a character can be located as shown in Figure 3.10. Since two successive direction codes must consist of three boundary vertices, the location of each subtype feature in the





a) Concave type



b) Convex type

Figure 3.9 Examples of curvature information from boundary line encoding.

four quadrants of a character can be obtained when we compare the middle vertex  $(x_i, y_i)$  of the successive direction codes with the centre point. Figure 3.11 is shown the position measurement grid in which is defined the conditions for the location of a subtype feature.

Each subtype feature is represented by the percentage occurrences with respect to the total number of successive pairs of direction code in the direction chain. By summing up each individual subtype feature in position 1 and 2 (right), position 2 and 3 (bottom), position 3 and 4 (left), and position 4 and 1 (top), we can obtain sixteen features. And, by summing up all the individual subtype features occurring in the four positions, four more features (overall occurrence of subtype features in the character) can also be obtained. Totally, twenty features are generated.

### 3.5 Inside Boundary Features

Since most of the shape analysis techniques have only processed the outer boundary of a character, some ambiguous results such as a thin '0' and a thick '1' may often cause confusions and errors in a recognition system. In Persoon and Fu's work [41], a large proportion of errors was caused by the confusion of '8's as '1's, and '0's as '8's. This is due to the fact that the outer boundaries of these pairs of characters are quite similar. The difficulty in distinguishing between

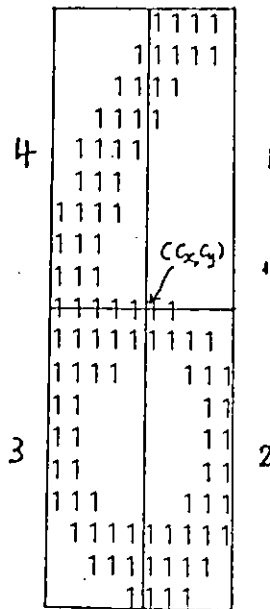


Figure 3.10 The four quadrants of a character and its centre point.

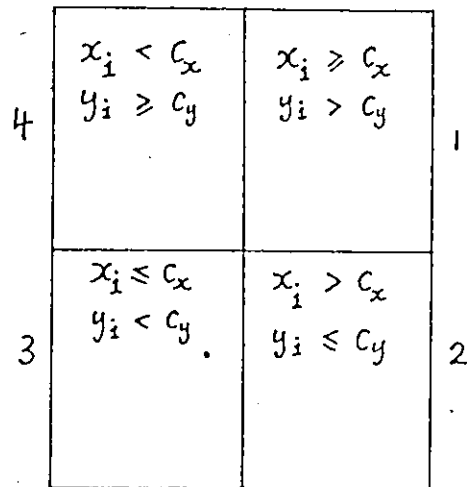


Figure 3.11 The position measurement grid.

characters on the basis of their outer boundaries is illustrated in Figure 3.12. This difficulty can be tackled if we have examined the inner details such as the presence, number and location of hole(s) in a character. This section is mainly concerned with the extraction of such features.

### 3.5.1 Hole Detection Algorithm.

Based on the idea that a hole is generally formed by the downward opening (↵) and the upward opening (↶) features, a technique for the detection of hole(s) has been developed and is described below :

The true size of a character is scanned point by point in a rowwise fashion from the leftmost column of the first row to the rightmost column of the last row. For each row, if the sequence 'black-white-black' (the sequential scanning of black points, then white points, then black points again) is encountered, the following conditions will be examined :

- a) If the number of black points in the preceding row is equal to the number of white points in the corresponding columns of the sequence, a downward opening (↵) is detected.
- b) If the number of black points in the succeeding row is equal to the number of white points in the corresponding columns of the sequence, an upward opening (↶) is detected.
- c) If the number of black points in the preceding and succeeding rows are both equal to the number of white points in the corresponding columns of the sequence, and the number

of white points is greater than two ( a threshold assuming that a hole must contain at least two white points), then a hole is detected.

111	11	1111	11111
111111	1111	111111	1111111
11 11	1111	111111	11 11
11 11	1111	11 11	11 11
11 11	1111	11 11	11 11
11 11	1111	11 11	11 11
11 11	1111	11 11	11 11
11 11	1111	11 11	11 11
11 11	111	11 11	11 11
1111	111	11 1	11111
111	111	11 1	11111
11111	111	1 1	11111
11 11	1111	1 11	11 11
11 11	1111	11 11	11 11
11 11	1111	11 11	11 11
11 11	1111	1111111	1111111
111111	11	11111	11111
1111			

Numeral '8' and '1'

Numeral '0' and '8'

Figure 3.12 An illustration of the confused pairs of characters on the basis of their outer boundaries.

For the above conditions, after the detection of each feature , the midpoint in the white area of the sequence must be computed so as to record the location of each existing feature. If condition a) is encountered, successive points in the the same column as the generated midpoint will be processed. After one or more white points in the successive rows were scanned, if a black point is encountered, we have to check whether this point belongs to one of the outer boundary (edge) points of the character stored in the memory. If the point is an edge point, the absence of hole is ensured, otherwise, the process for the detection of the upward opening feature will be performed

starting from the first column but the preceding row of this point. After both features have been detected, a hole is ensured, and by linking the midpoints generated from the upward and downward opening features, a midpoint in this linked line can be located which represents the approximate location of the detected hole. We have to note that if the downward opening feature has been extracted but no hole feature is detected (e.g. no upward opening is detected), the unscanned point which is located in the succeeding column but the same row as the extracted feature will be processed.

Let us define D and U as the midpoint of the generated downward and upward opening features respectively, and M the generated midpoint which represents the location of the detected hole. As shown in Figure 3.13, the true length of the character is divided into four equal regions and the type of holes with respect to its location can be defined as

- a) Upper hole -- where M, D, U are all located in region 1 and 2.
- b) Lower hole -- where M, D, U are all located in region 3 and 4.
- c) Big hole -- where D is located in region 1, and U is located in region 4.
- d) Upper middle hole -- where M and D are located in either region 1 or region 2, and U is located in either region 2 or region 3.
- e) Lower middle hole -- where M and U are located in either region 3 or region 4, and D is located in either region 2 or

region 3.

After a hole is processed, we have to record its presence, and the detection process will continue to operate in the second row which succeeds the midpoint generated from the upward opening feature.

Totally, there are six features used to describe the inner details of a character, they are 1) the number of holes detected, 2) the number of upper holes, 3) the number of upper middle holes, 4) the number of lower middle holes, 5) the number of big holes, and 6) the number of lower holes.

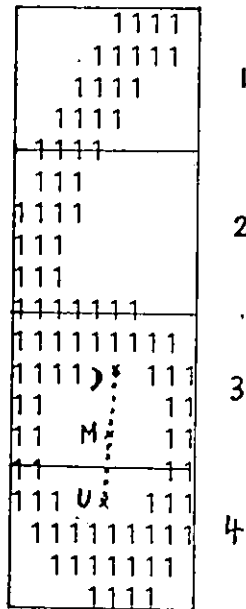


Figure 3.13 Four divided regions in a character grid.

### 3.6 Summary of Generated Features

The following list will summarize all the features generated from this feature extraction scheme :

A. Fourier shape descriptors	10
B. Boundary line encoding features	
1) direction features	8
2) direction length features	8
3) curvature features in	
i) the top, left, bottom and right part	
a) concave type	8
b) convex type	8
ii) the overall character	
a) concave type	2
b) convex type	2
C. Inside boundary features	6



## CHAPTER FOUR

## CLASSIFICATION PROCESS

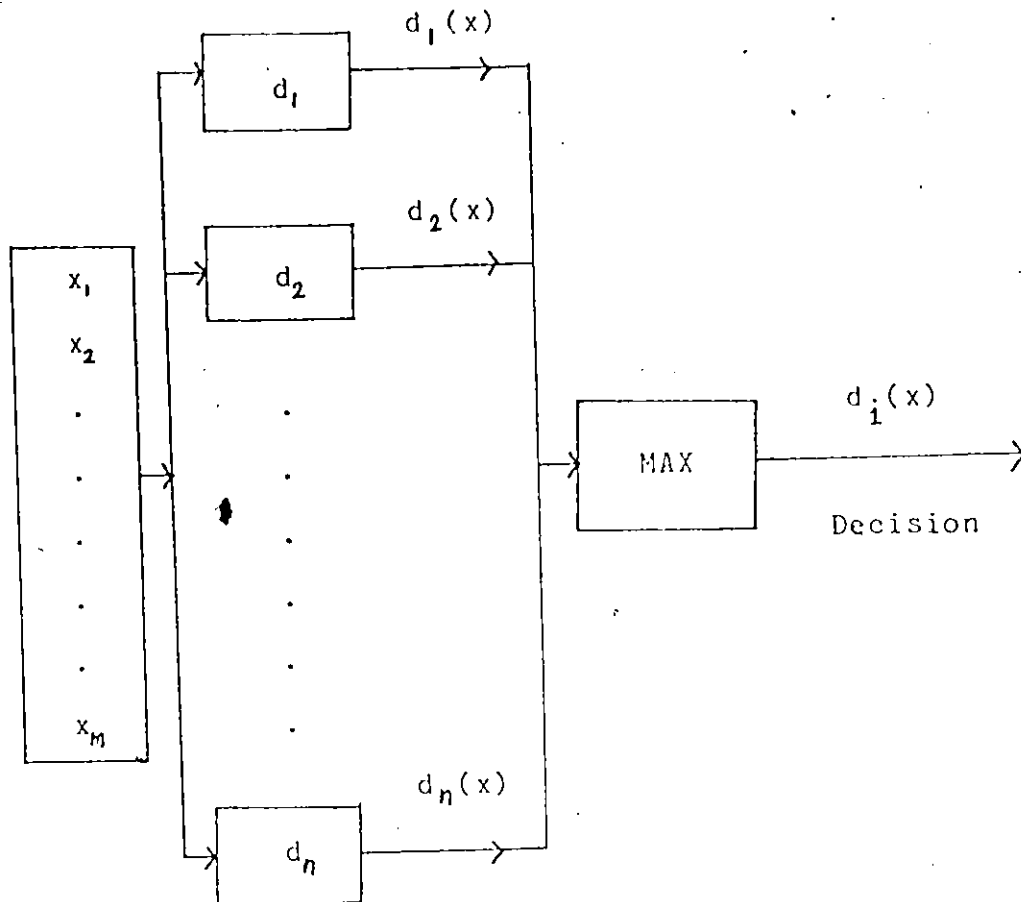
## 4.1 Introduction

Based on the measurements taken from the selected features, a classifier in a pattern recognition system will assign the pattern to a finite number of classes for identification. A usual way of representing a pattern classifier is a set of discriminant functions  $d_i(x)$ ,  $i = 1, \dots, n$ . The classifier assigns a pattern  $x$  to class  $w_i$  if

$$d_i(x) > d_j(x) \text{ for all } j \neq i.$$

In other words, if the  $i$ th discriminant function,  $d_i(x)$ , has the largest value for a pattern  $x$ , then  $x$  is assigned to class  $w_i$ . The representation of a classifier is illustrated by a block-diagram shown in Figure 4.1.

There are two general approaches for classifying patterns, namely, parametric and nonparametric classifications. Parametric classifications are used when each of the possible input classes is known a priori to be representable by a set of parameters, such as the conditional probability density functions of the feature vectors, etc. When complete a priori knowledge about the patterns to be recognized is available, the classifier may be able to make decisions with no further operation. Nonparametric classification schemes are used when no a priori information



Feature  
vector  
of  
pattern  
x

Discriminant  
calculator

Maximum  
Selector

Figure 4.1 A pattern classifier.

about the patterns is known. Under this circumstance, a direct training or learning of classification rule from input patterns must be performed.

The process of learning can be divided into two types : supervised and nonsupervised learning. In supervised learning, or learning with a teacher, the correct classification of each sample of the training patterns is known, and by evaluating the performance on the patterns, an appropriate learning procedure can then be implemented. In nonsupervised learning, or learning without a teacher, the classification of the training samples is not known, and actual learning of pattern classes from the selected feature measurements is needed. It is obvious that the nonsupervised learning situation is very difficult to solve, continuous investigations by researchers have developed an approach, called clustering analysis [11], [13], to solve this type of problems. The many different classification techniques which have been used to solve pattern recognition problems are described briefly in [ 2], [21], [53].

In our system, since a priori information about each character's feature set is not known, a variety of nonparametric procedures which incorporates discriminant-analysis-based techniques and nearest neighbor classification method have been utilized to develop a classification scheme for decision-making.

#### 4.2 Discriminant Analysis Technique

The theories underlying discriminant analysis and its practical applications are briefly described in [ 1], [ 9], [34]. When applied to pattern recognition, the goal of discriminant analysis is to assign an input pattern,  $x$ , to one or more distinct pattern classes on the basis of extracted features or discriminating variables. Classification can be achieved through the use of a series of discriminant functions in the form

$$D_i = d_{i1} x_1 + d_{i2} x_2 + \dots + d_{in} x_n + c_{i0}$$

where  $D_i$  is the discriminant score for class  $i$ , the  $d$ 's are weighting coefficients and  $c_{i0}$  is a constant. The  $x$ 's are the discriminating variables used in the analysis. There is always a separate equation for each class; thus if there are ten classes, ten discriminant scores will be generated for each input pattern case, and the case would be classified into the class with the highest score.

For initial computation, a set of patterns with known class memberships are used for learning. Once the weighting coefficients from the learning process is found, a set of discriminant functions can be derived from multiplying the coefficients by the discriminating values, summed together, and added onto a constant. The classification of new cases with unknown memberships can then be assigned to a class with the

highest score. The equations for computing the discriminant functions in our system are based on [1], [29], and the procedures are described below :

For each pattern class  $k = 1, 2, \dots, g$ , where  $g$  is the number of classes, the means of  $m$  discriminating variables in each class, the sums of cross-product of derivations from means, and the pooled dispersion matrix for the set of classes are computed as follows :

(1) Means :

$$\bar{x}_{jK} = \frac{\sum_{i=1}^{n_K} x_{ijk}}{n_K} \quad (4.2.1)$$

where  $n_K$  = sample size in the  $k$ th class,  
 $j = 1, 2, \dots, m$ .

(2) Sum of cross-products of derivations from the means :

$$s_K = \sum_{j=1}^m \sum_{p=1}^m (x_{ijk} - \bar{x}_{jK}) (x_{ipk} - \bar{x}_{pK}) \quad (4.2.2)$$

(3) Pooled dispersion matrix :

$$D = \frac{\sum_{k=1}^g s_K}{\sum_{k=1}^g n_K - g} \quad (4.2.3)$$

where  $g$  = number of classes.

A set of linear functions which serve as indices for classifying an individual pattern into one of several classes can be computed from the following :

For each discriminant function  $k^* = 1, 2, \dots, g$ , we have the following statistics :

(4) Weighting coefficients :

$$c_{ik^*} = \sum_{j=1}^m d_{ij} \bar{x}_{jk} \quad (4.2.4)$$

where  $i = 1, 2, \dots, m$

$k = k^*$

$d_{ij}$  = inverse element of the pooled dispersion matrix D

(5) Constant :

$$c_{0k^*} = -\frac{1}{2} \sum_{j=1}^m \sum_{p=1}^m d_{jp} \bar{x}_{jk} x_{pk} \quad (4.2.5)$$

For each  $i$ th case in each  $k$ th class, the following calculations are performed :

(6) Discriminant functions :

$$f_{k^*} = \sum_{j=1}^m c_{jk^*} x_{jk} + c_{0k^*} \quad (4.2.6)$$

where  $k^* = 1, 2, \dots, g$ .

As mentioned in [30], the rule of assigning a case to the class with the highest score is equivalent to assigning a case to the class which has the greatest probability. The probability associated with the largest discriminant function can also be computed as follows :

$$P = \frac{1}{\sum_{k^*=1}^g e^{(f_{k^*} - f_L)}} \quad (4.2.7)$$

where  $f_L$  = the value of the largest discriminant function  
 $L$  = the subscript of the largest discriminant function

In our system, all features except those which describe the inner boundary are used as the discriminating variables, and the discriminant analysis technique is applied for fine-tuning rather than classification purposes. In order to obtain a more precise identification of each input pattern, the first two highest score classes and the value of the greatest probability are recorded. The information gathered will be used in the classification scheme described in section 4.4.

#### 4.3 Nearest Neighbor Classification Method

Nearest neighbor classification technique is the simplest and the most intuitive approach which employs distance functions for pattern classification. It is an extension of minimum-distance pattern classification concept [21], [53], which uses the distances between the input pattern and a set of reference vectors or prototype patterns in the feature space as the classification criterion. Let us consider  $M$  pattern classes which are representable by reference vectors  $r_1, r_2, \dots, r_M$ , and given  $r_i$  associated with the pattern class  $w_i$ . The distance,  $d_i$ , between an input feature vector  $x$  and  $r_i$  can be defined as

$$d_i = \text{Min. } |x - r_i| = \text{Min. } \sqrt{(x - r_i)^T (x - r_i)} \quad (4.3.1)$$

where  $i = 1, 2, \dots, M$

$T$  = the transpose operation to a vector.

This implies that a classifier computes the distance from an

unknown class pattern  $x$  to the reference vector of each class, and assigns the input to a pattern class which is associated with the closest vector set. In other words, pattern  $x$ , is assigned to class  $w_i$  if  $d_i < d_j$ , for all  $j \neq i$ . Ties are resolved arbitrarily.

By squaring all the terms in equation (4.3.1), we have

$$d_i^2 = \text{Min} ( x^T x - x^T r_i - r_i^T x + r_i^T r_i ) \quad (4.3.2)$$

Since  $x^T x$  is independent of  $i$  for all  $d_i$ , equation (4.3.2) will become

$$d_i^2 = \text{Min} ( -x^T r_i - r_i^T x + r_i^T r_i ) \quad (4.3.3)$$

Because all the computed distances are positive, choosing the minimum  $d_i^2$  is equivalent to choosing minimum  $d_i$ , and the minimum  $d_i$  is equivalent to the maximum  $( x^T r_i + r_i^T x - r_i^T r_i )$ , the decision function used is

$$d_i(x) = \text{Max} ( x^T r_i + r_i^T x - r_i^T r_i ) \quad (4.3.4)$$

Hence, a pattern  $x$  is assigned to class  $w_i$  if  $d_i(x) > d_j(x)$  for all  $j \neq i$ .

In our system, the means of all the features (except the means of the inner boundary features) in each class of the training patterns are the elements in the reference vectors, and nearest neighbor classification based on the entire features set and the Fourier descriptors set are used separately as the reference vector. The nearest neighbor classification technique

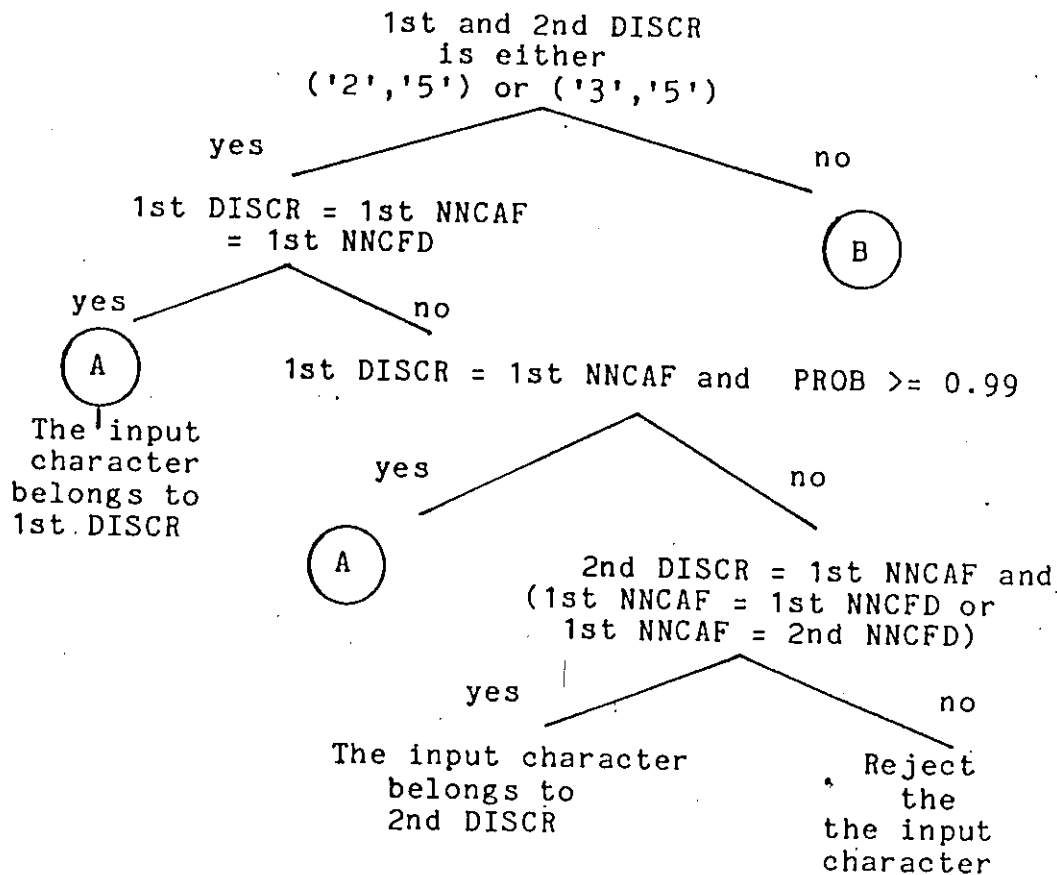


is also used for fine-tuning purpose. The first two assigned classes which have the maximum  $d$ 's computed from the entire feature set and the Fourier descriptors feature set are recorded and will be used for further processing.

#### 4.4 Classification Scheme

A simple implementation of a classification scheme based on the results from discriminant analysis and nearest neighbor classification techniques can be achieved through a classification tree. A priori information based on the basic structure of the numerals such as the numeral '0' contains a big hole, the numeral '6' contains a hole in its lower part, and the numeral '9' contains a hole in its upper part has been set up in the tree, so as to make the classification process much simpler. The classification scheme which is represented in the form of a tree structure for decision-making is shown in Figure 4.2. More details about the justification of the use of this classification scheme can be found in section 5.2.

For the purpose of generating classification statistics, an identification tag which establishes the pattern class is attached to each input pattern and each pattern class in the classification scheme. After a pattern is processed through the classification scheme, three results can be obtained, they are : 1) correct classification, 2) misclassification, and 3) unclassified. Correct classification occurs when the identification tag matches with that of the assigned pattern



Codes and symbols :

-----

DISCR -- Assigned character class based on Discriminant functions

NNCAF -- Assigned character class based on Nearest neighbor classifications of the entire feature set, excluding the hole features.

NNCFD -- Assigned character class based on Nearest neighbor classifications of the Fourier descriptor features

PROB -- Probability associated with the largest discriminant function

'x' -- Identification of numeral x

Figure 4.2a Classification tree for decision-making.

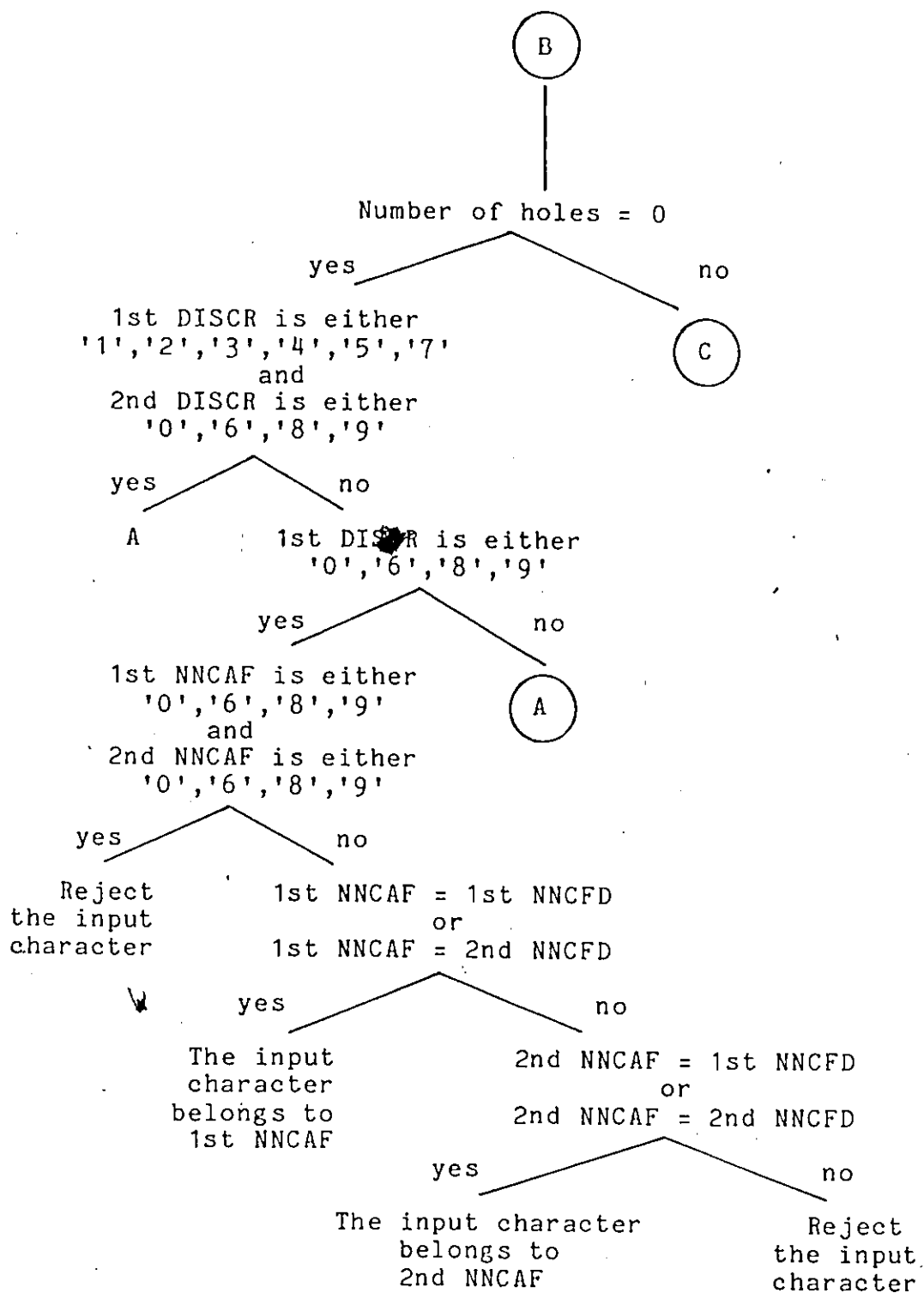


Figure 4.2b Classification tree for decision-making.

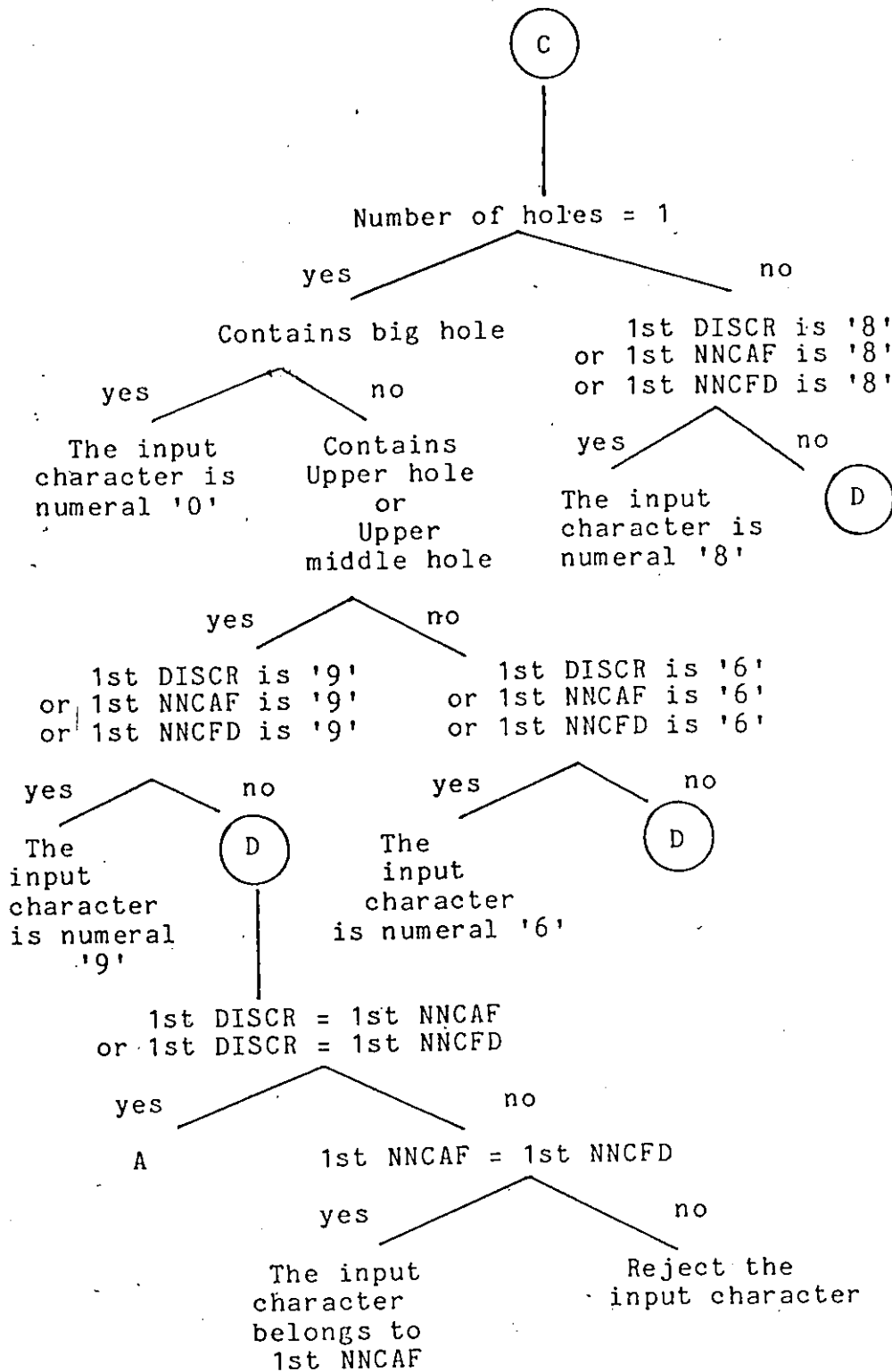


Figure.4.2c Classification tree for decision-making.

class. Misclassification occurs when the identification tag is different from that of the assigned pattern class. The unclassified (rejected) situation occurs when no pattern class is assigned to the input pattern. A confusion matrix which indicates the classification results has also been generated and will be described in the next chapter.

## CHAPTER FIVE

## EXPERIMENTAL RESULTS

## 5.1 Data Sets

The detailed structure of the proposed system shown in Figure 5.1 has been implemented on a CDC 7200 digital computer using FORTRAN computer language. In order to establish its performance, thousands of numeric samples from different data bases have been tested. The data used for the recognition experiments consist of the following :

Set A : IEEE Data Base #1.2.1

Source --

Honeywell Information System  
Data Systems Division  
Dr. A. L. Knoll

Description --

The data base consists of 50 samples of each numeric character handprinted by nine different authors. Simple printing rules were specified but not always followed. The images are binary with a resolution of 25x21.

Typical Handprints --

As shown in Figure 5.2a.

Set B : Suen's Data Base

## Source --

Concordia University  
Sir George Williams Campus  
Department of Computer Science  
Dr. C. Y. Suen

## Description --

The data base consists of more than 100,000 alphanumeric characters. There are 600 samples of each character written by 30 different authors, 15 of them are left-handed writers, and the other 15 are right-handed. Before the samples were written, brief instructions of writing were given and subjects were asked to write as close as possible to the character models in the shortest time. The images are digitized with a resolution of 64x32.

## Chosen character set --

As shown in Figure 5.2b.

Set C : OCR-A Data Base

## Source --

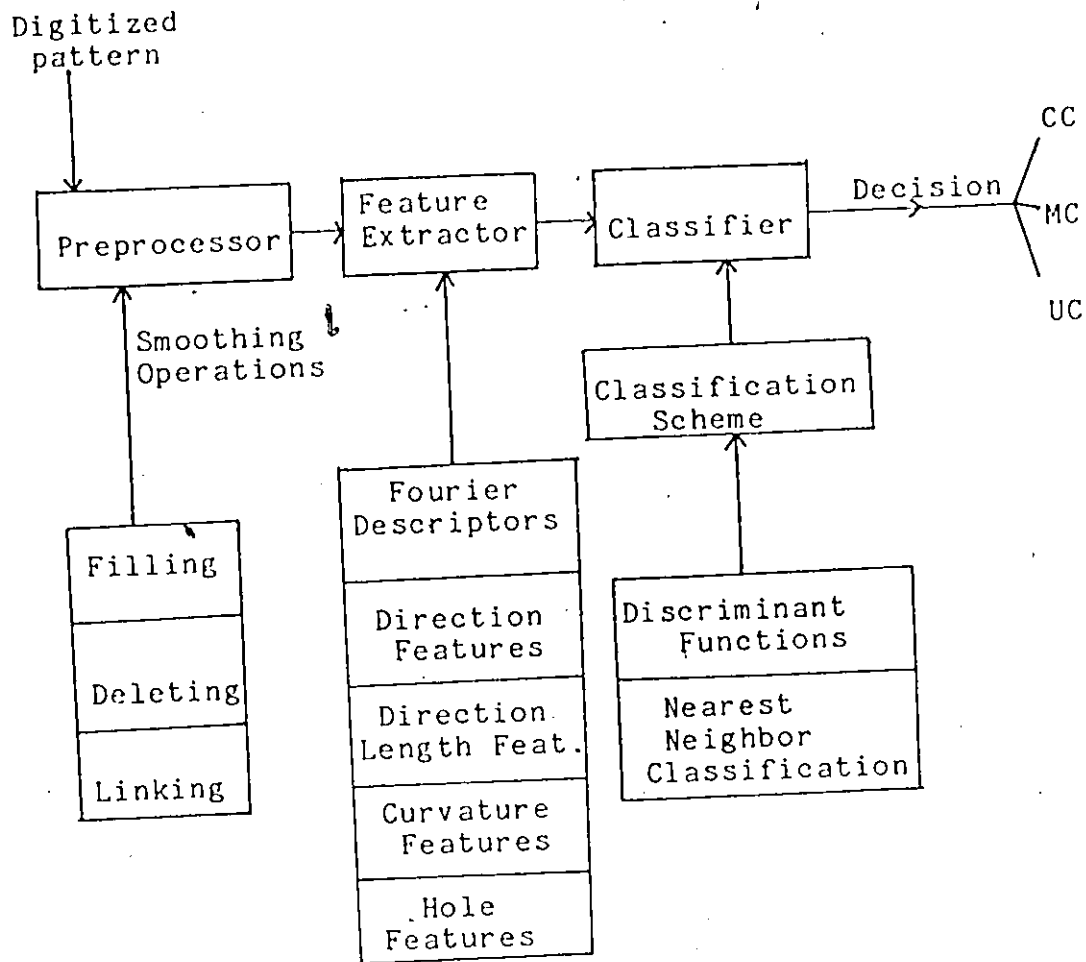
Concordia University  
Sir George Williams Campus  
Optical Character Recognition and Data Processing  
Laboratory

## Description --

The data base consists of 3000 OCR-A typewritten numerals. There are 300 samples of each character, the images are stored in a hexadecimal form and digitized with a resolution of 24x16.

Standard characters --

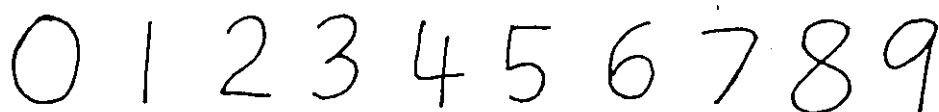
As shown in Figure 5.2c.



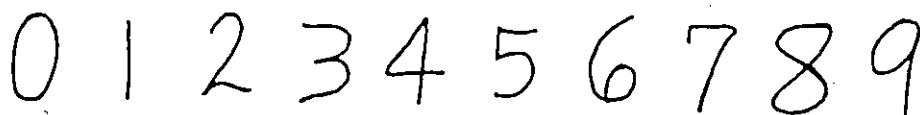
Codes : CC -- Correctly Classified  
 MC -- Misclassified  
 UC -- Unclassified (Rejected)

Figure 5.1 Detailed structure of the proposed character recognition system.

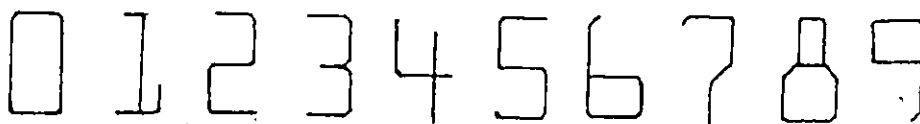


A row of ten handwritten numerals from 0 to 9. The digits are written in a cursive, slightly slanted style with varying stroke thickness and some irregularities, characteristic of a typical handprint.

a) Typical handprint numerals of set A.

A row of ten chosen numeric character models from 0 to 9. These characters are more uniform and stylized than the handprints, appearing as clean, slightly rounded, and consistent in appearance.

b) Chosen numeric character models of set B.

A row of ten standard numeric character fonts from 0 to 9. These characters are highly uniform, rectangular, and resemble a standard digital or monospace font style.

c) Standard numeric character fonts of set C.

Figure 5.2 Standard numeral models from different data bases.

## 5.2 Results from recognition experiments

Data set A : A training set which consisted of 500 samples was used, and the same 500 samples were tested and classified based on the highest score generated from the discriminant functions. After an error rate of 1.2 percent was obtained (no rejection process was added, and classification results are shown in Table 5.1a), the classification scheme described in section 4.4 was designed for decision-making. A set of experiments based on this scheme was performed :

a) A set of 500 samples was trained and tested, and a confusion matrix which indicates the classification results is shown in Table 5.1b.

b) A training set consisting of the first 25 samples of each numeric characters was used and tested, an error rate of 0.4 percent was obtained. The remaining 250 samples were tested, a rejection rate of 0.4 percent was observed with no misclassified samples. Table 5.1c is shown the confusion matrix and the overall classification results for the entire set.

Both the above experiments misclassified the same sample, which is the numeral '8' but was substituted as numeral '9'. Figure 5.3 is shown this misclassified sample and the rejected sample of the testing set from experiment b. The numeral '1' was rejected because it was written in a way (left-skewed) different from it's typical model which caused much confusions with other numerals. Though no perfect recognition has been obtained from

the experiments, the results compare favourably with those achieved by investigators who used the same data base [31], [33], [45].

Confusion matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	49								1		
1		50									
2			49			1					
3				50							
4					50						
5			1			49					
6							50				
7								50			
8	1				1		1		47		
9										50	

Results :

Number of character(s) correctly classified = 494 (98.80%)

Number of character(s) misclassified = 6 ( 1.20%)

Table 5.1a Classification results from data set A (500 training samples -- classification was based on the highest score generated from the discriminant functions).

Confusion matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	50										
1		50									
2			50								
3				50							
4					50						
5						50					
6							50				
7								50			
8									49	1	
9										50	

Overall results :

Number of character(s) correctly classified = 499 (99.80%)

Number of character(s) misclassified = 1 ( 0.20%)

Number of rejected character(s) = 0 ( 0.00%)

Table 5.1b Results from data set A (500 training samples -- Classification scheme was used).

Confusion matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	50										
1		49									1
2			50								
3				50							
4					50						
5						50					
6							50				
7								50			
8									49	1	
9											50

Overall results :

Number of character(s) correctly classified = 498 (99.60%)

Number of character(s) misclassified = 1 ( 0.20%)

Number of rejected character(s) = 1 ( 0.20%)

Table 5.1c Results from data set A (250 samples for training).

```

      11111
     111111111
    1111111111111
   11111      111
  1111      1
 111      111
 11      1111
 11      11111
 111      11111
 1111      11111
   1111 1111
    1111 1111
     1111111
      111111
       11111
        111111
         1111 11
          111 11
           1111 111
            111 11
             111 11
              111 11
               111 11
                111 111
                 11 111
                  11 111
                   11 11

```

Misclassified numeral

```

 111
 111
 111
 111
 11
 11
 11
 11
 111
 111
 111
 111
 1111
 111
 1111
 1111
 111

```

Rejected numeral

Figure 5.3 Misclassified and rejected samples of data set A.

Data set B : By using the classification scheme for decision-making, a training set consisting of 300 samples of each numeric character (ten from each author, i.e., 150 left-handed samples, 150 right-handed samples) was used and tested. An error rate of 0.17 percent and a rejection rate of 0.45 percent were observed. In this test, four samples of numeral '9' were misclassified as '4', and one sample of numeral '8' was misclassified as '6'. The remaining set (3000 samples) was tested, an error rate of 0.10 percent and a rejection rate of 0.5 percent were obtained which included two samples rejected by the contour tracing process. In this testing set, one sample '4' was misclassified as '9', one sample '9' was misclassified as '4', and one sample '0' was misclassified as '2'. Since the numerals '4' and '9' are quite similar in shape, classification process for this pair is difficult. When the first two highest score classes from discriminant functions belong to this two classes, classification based on the highest discriminant score is used, and the input character will be rejected when no hole was detected. The confusion matrix and the overall classification results for this data set are shown in Table 5.2a.

Figure 5.4 (a) and (b) is shown some misclassified and rejected samples from this experiment, and the two samples rejected during the contour tracing process are shown in Figure 3.4 of chapter 3.

The data set has also been tested and classified based on the highest score generated from the discriminant functions. The

Confusion matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	591		1								8
1		599									1
2			600								
3				599							1
4					596					1	3
5						599					1
6							593				7
7								600			
8							1		597		2
9					5					590	5

Overall results :

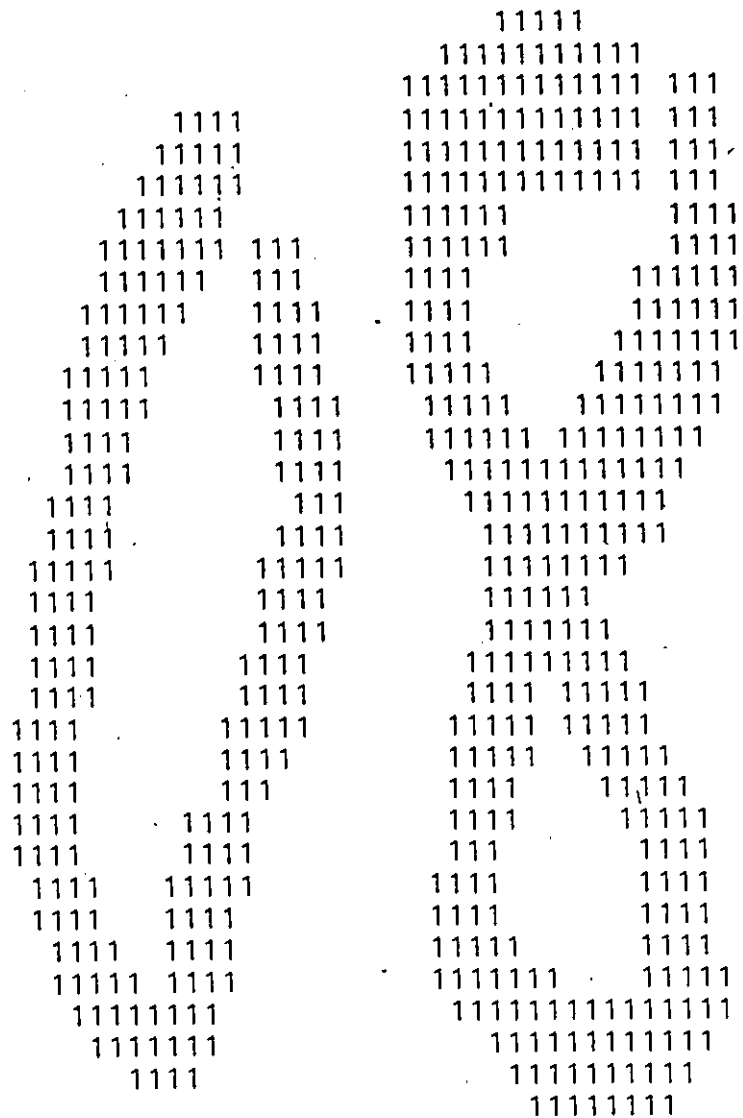
Number of character(s) correctly classified = 5964 (99.40%)

Number of character(s) misclassified = 8 ( 0.13%)

Number of rejected character(s) = 28 ( 0.47%)

Table 5.2a Classification results from data set B (3000 training samples).

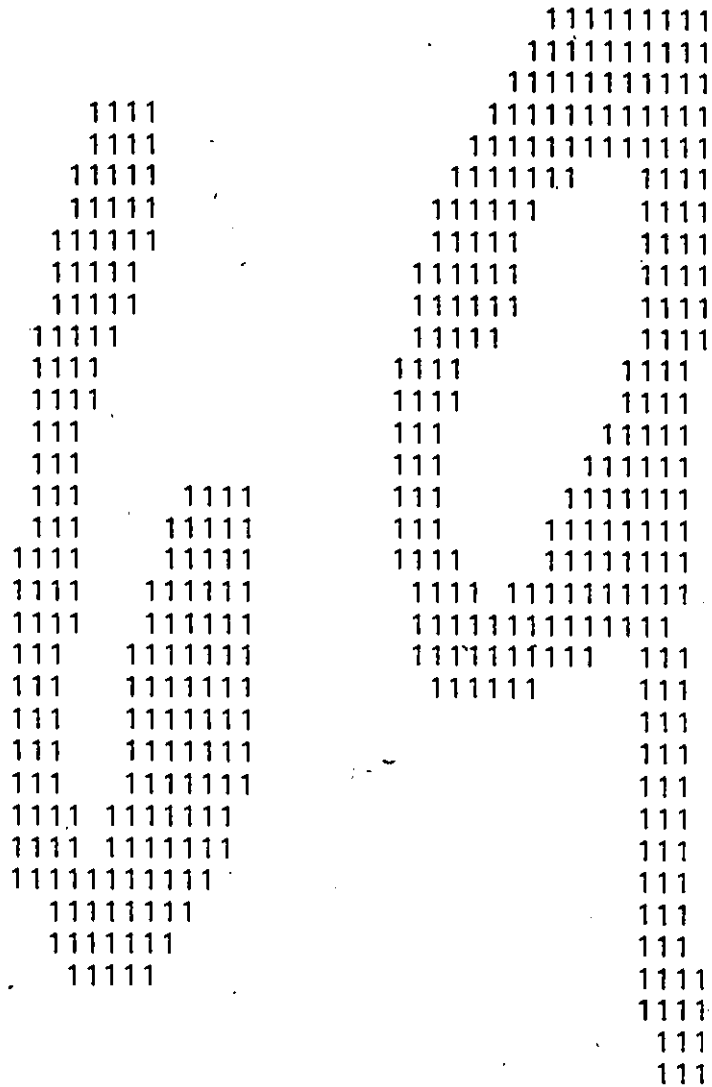




Rejected

Misclassified

Figure 5.4a Misclassified and rejected samples of data set B.



Rejected

Misclassified

8

Figure 5.4b Misclassified and rejected samples of data set B.

Confusion matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	297						3				
1		300									
2			300								
3				300							
4					300						
5			2			297			1		
6	1						299				
7								300			
8									300		
9					9					291	

Results :

Number of character(s) correctly classified = 2984 (99.47%)

Number of character(s) misclassified = 16 ( 0.53%)

Table 5.2b Classification results of 3000 training samples from data set B (Classification was based on the highest score generated from the discriminant functions).

Confusion matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	293						4		3		
1		299									1
2			300								
3				299							1
4					299					1	
5						300					
6	1						299				
7								300			
8			1	7					292		
9	1				1					298	

Results :

Number of character(s) correctly classified = 2979 (99.30%)

Number of character(s) misclassified = 19 ( 0.63%)

Number of rejected character(s) = 2 ( 0.07%)

Table 5.2c Classification results of 3000 testing samples from data set B (Classification was based on the highest score generated from the discriminant functions).

same 3000 samples were trained and tested, an error rate of 0.53 percent was obtained. The remaining 3000 samples were then tested, an error rate of 0.63 percent and a rejection rate of 0.07 percent were observed (this rejection rate was obtained from samples rejected by the contour tracing process). Tables 5.2b and 5.2c are shown the classification results for the training and testing sets respectively. The overall classification results for the entire set from this experiment are shown as follows :

Number of character(s) correctly classified	=	5963 (99.38%)
Number of character(s) misclassified	=	35 ( 0.58%)
Number of rejected character(s)	=	2 ( 0.03%)

In the above experiments, most of the misclassified samples were the numeral '9' substituted as '4', or vice versa, this is due to the fact that the shapes of these two models are quite similar to each other. The other samples which were misclassified or rejected mostly because they are written in a way which is different from their respective standard models. In addition, the results obtained from these experiments compare favourably with [32] who used this same data base.

Data set C : Based on the classification scheme, a training set consisting of 1500 samples was used and tested. After the error and rejected rates of 0.40 and 0.20 percent were obtained, the method was tested on the remaining 1500 samples, and the error and rejected rate with both equal 0.2 percent were

obtained. Since the numeral '2' and numeral '5' in this data base are symmetric in shapes, they might contain the same amount of generated features, and thus these two numerals constituted the most misclassification rates in the experiments. Table 5.3a is shown the confusion matrix and the overall classification results for the entire set.

The same experiment was used and the classification was based on the highest score computed from the discriminant functions, and no rejection process was added. An error rate of 0.13 percent was obtained from the 1500 training set, and a rejected rate of 0.2 percent was obtained from the remaining 1500 testing set with no misclassified samples. The overall classification results are shown in Table 5.3b.

For both experiments, three characters were rejected during the contour tracing process, the common rejected samples and some misclassified samples are shown in Figure 5.5.

Confusion Matrix :

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	300										
1		300									
2			297			3					
3				297		1					2
4					300						
5			4			295					1
6						1	299				
7								300			
8									297		3
9										300	

Overall results :

Number of character(s) correctly classified = 2985 (99.50%)

Number of character(s) misclassified = 9 ( 0.30%)

Number of rejected character(s) = 6 ( 0.20%)

Table 5.3a Classification results from data set C (1500 training samples).

Confusion Matrix.:

I \ O	0	1	2	3	4	5	6	7	8	9	REJECT
0	300										
1		300									
2			300								
3				298							2
4					300						
5						299					1
6						1	299				
7								300			
8				1					299		
9										300	

Overall results :

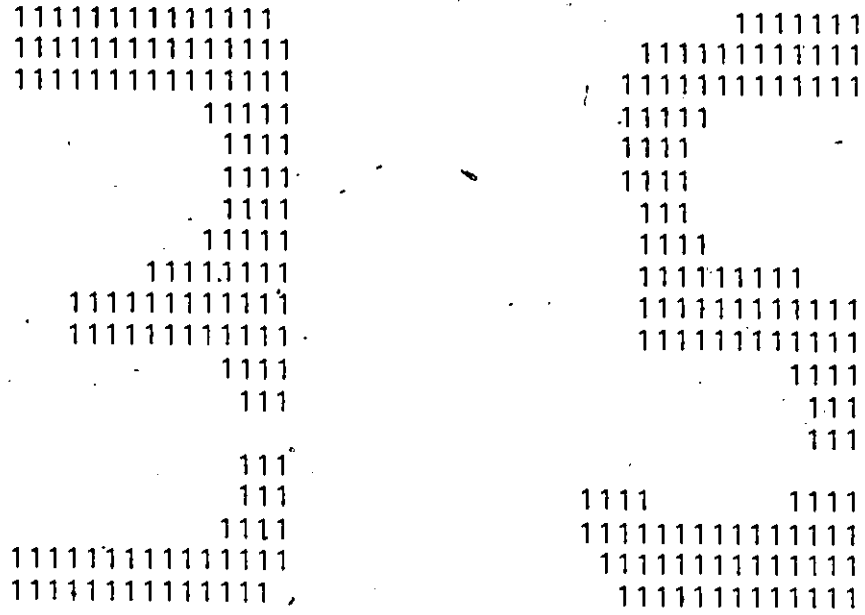
Number of character(s) correctly classified = 2995 (99.83%)

Number of character(s) misclassified = 2 ( 0.07%)

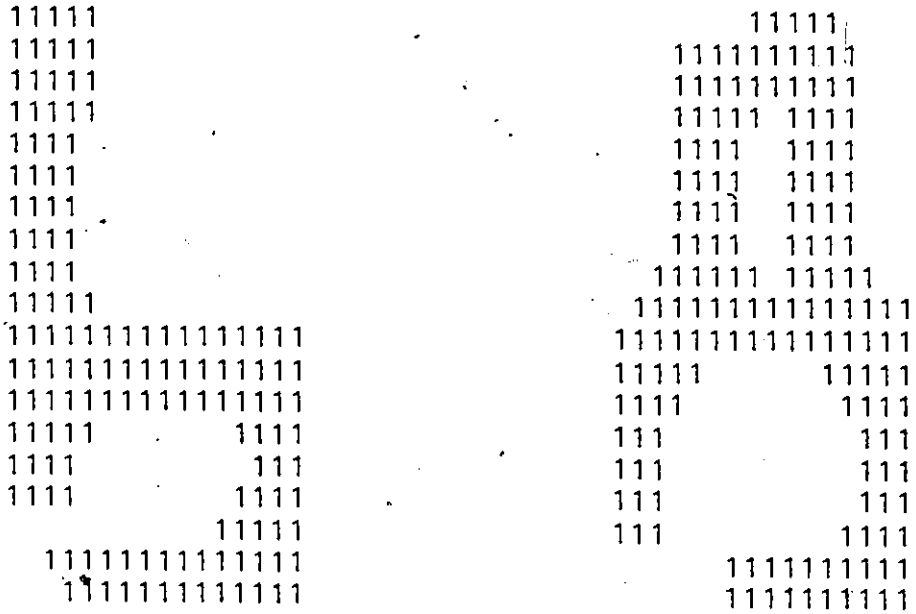
Number of rejected character(s) = 3 ( 0.10%)

Table 5.3b Classification results from data set C (1500 training samples, classification was based on the highest score class generated from the discriminant functions).





Rejected samples



Misclassified samples

Figure 5.5 Rejected and Misclassified samples of data set C. \* \*

## CHAPTER SIX

## CONCLUSIONS AND DISCUSSIONS

## 6.1 Conclusions

A character recognition system for the automatic recognition of handprinted and machine-printed numeric characters has been developed and implemented on a digital computer. The design of the system is based upon the smoothing operation which enhanced the quality of input data, the inner and outer boundary feature extractions, and a classification scheme which incorporates discriminant analysis and nearest neighbor classification techniques for decision-making. High recognition rates resulted from experiments with several data bases have established the good performance of this system.

## 6.2 Discussions and Comments

Besides the successful performance of the proposed system, some details for further development in the system in order to ensure its reliability are suggested as follows :

- 1) In the system, numeric characters were used for recognition. In order to further establish its reliable and powerful performance, extension to the recognition of all alphanumeric characters can be performed.

- 2) Most of the samples misclassified and rejected by the system contain breaks in their strokes. Certain modifications in the linking process such as increasing the threshold for the black points counter can be tried.
- 3) The position measurements grid used in the feature extraction process is useful for constrained characters which are usually written in less skewed form. In order to extend the scope to recognize unconstrained characters, certain preprocessing techniques must be performed so as to correct the skewness (left or right inclined) of the character. The occurrence of certain features in each defined position can then be precisely obtained.
- 4) The hole detection algorithm developed in this study may not work well when the input character is too thin or represented in skeleton form, certain modifications should be made.
- 5) The features employed in the system may contain some redundant information which can affect the computed decision functions. Investigations in selecting optimal features for classification may improve the recognition performance of the system.
- 6) In the classification scheme designed for decision-making, a priori knowledge was set up based on the basic structure of the numerals. For unconstrained characters, the recognition rates of the system may decrease when the characters are not written in their standard form, such as the numeral '6' is written as '6', or the numeral '0' is written as '0' etc. By increasing several typical class

models of each character, a classification scheme based on the a priori structures of each class model can be reset.

- 7) Efficient procedures may be incorporated in the present system to resolve the confused pairs of characters such as closed-opening numerals '4' and '9', '2' and '5', etc.
- 8) The observed results from OCR-A characters have indicated that the experiment based on the classification scheme gives more confusions than that based on the discriminant functions. This phenomenon may indicate that fine-tuning technique is not necessary for machine-printed (fixed fonts) characters. In order to confirm this, more machine-printed characters should be applied to the system for recognition.
- 9) The system can classify approximately fifteen characters per second in the CDC Cyber 172 digital computer, higher speed can be obtained by optimizing or refining the written computer programs.

## REFERENCES

1. Anderson, T. W., "Introduction to multivariate statistical analysis," John Wiley and Sons, 1958.
2. Andrews, H. C., Introduction to Mathematical Techniques in Pattern Recognition, New York: Wiley, 1972.
3. Auerbach on Optical Character Recognition, Princeton: Auerbach Publishers, 1971.
4. Brill, Evan L., "Character recognition via Fourier descriptors," presented at WESCON, Session 25, Qualitative Pattern Recognition Through Image Shaping, Los Angeles, California, pp. 1-10, August 1968.
5. Brill, E. L., R. P. Heydorn and J. D. Hill, "Some approaches to character recognition for postal address reader applications," Proc. Automatic Pattern Recognition, pp. 19-40, 1969.
6. British Computer Society, Character Recognition, London, 1971.
7. Chang, S. K. and G. Nagy, "Deposit-slip-first check reading," IEEE Trans. Systems, Man and Cybernetics, vol. 7, pp. 64-68, 1977.
8. Clemens, Jon K. and Mason, Samuel J., "Character recognition in an experimental reading machine for the blind," in Recognizing Patterns, Kolers, Paul A. and M. Eden (eds.), M. I. T. Press, pp. 156-167, 1968.
9. Cooley, W. W. and P. R. Coones, Multivariate Data Analysis,

New York, John Wiley and Sons, 1971.

10. Deutsch, E. S. "Thinning algorithms on rectangular, hexagonal, and triangular array," Comm. of the ACM, vol. 5, #9, pp. 827-837, Sept. 1972.
11. Diday, E. and J. C. Simon, "Clustering Analysis," in Digital Pattern Recognition, K. S. Fu (ed.), New York:Springer, pp.47-92, 1976.
12. Dinneen, G. P., "Programming pattern recognition," Proceedings on Western Joint Computer Conference, pp. 94-100, 1955.
13. Duda, R. O. and P. E. Hart, Pattern classification and Scene Analysis, New York: Wiley, 1973.
14. Earnest, L. D., "Machine recognition of cursive writing," in Information Processing, C. M. Popplewell (ed.), Amsterdam. The Netherland : North Holland, pp. 462-466, 1963.
15. Eden, M. and M. Halle, "Characterization of cursive handwriting," Proc. Fourth London Symp. on Information Theory, C. Cherry (ed.), Butterworths Scientific Publications, London, pp.287-299, 1961.
16. Fischer, G. L. Jr., D.J. Pollock, B. Raddack and M. E. Stevens (eds.), Optical Character Recognition, Spartan Books, Washington, D. C., 1962.
17. Freeman, H. "On the encoding of arbitrary geometric configurations," IRE Trans. Electron. computer, EC-10, #2, pp. 260-268, 1961.
18. Freedman, M, David, "Optical character recognition," IEEE Spectrum, pp. 44-52, March 1974.

19. Freyer, W. D. and G. E. Richmond, "Two dimensional spatial filtering and computers," Proc. Nat. Electronics Conf. 18, p. 529, 1962.
20. Frishkopt, L. S. and L. D. Harmon, "Machine reading of cursive script," Proc Fourth London Symp. on Information Theory, C. Cherry (ed.), Butterworths Scientific Publications, London, pp. 300-316, 1961.
21. Fu, K. S. (ed.), Digital Pattern Recognition, Communication and Cybernetics, vol. 10, New York : Springer, 1976.
22. Genchi, H., K. Mori, S. Watanabe, and S. Katsuragi, "Recognition of handwritten numeral characters for automatic letter sorting," Proc. IEEE, vol. 56, pp. 1292-1301, 1968.
23. Granlund, G. H., "Fourier preprocessing for handprint character recognition," in IEEE Trans. Computers, vol. C-21, pp. 195-201, 1972.
24. Gray, Peter J., "Optical scanning, OCR, and MICR," Automatic Data Processing Handwork, The Diebold Group (ed.), pp. 2-117 - 2-125, 1977.
25. Greanias, E. C., P. F. Meagher, R. J. Norman, and P. Essinger. "The recognition of handwritten numerals by contour analysis," IBM J. Research and Develop., vol. 7, pp. 14-22, Jan. 1963.
26. Gudesen, A., "Quantitative analysis of preprocessing techniques for the recognition of handprinted characters," Pattern Recognition, vol. 8, pp. 219-227, 1976.
27. Harmon, Leon D., "Automatic recognition of print and script," Proc. of the IEEE, vol. 60, #10, pp. 1165-1176, October, 1972.

28. Hussian, A. B. S., G. T. Toussaint, and R. W. Donaldson, "Results obtained using a simple character recognition procedure on Munson's handprinted data," IEEE Trans. computers, vol. C-21, pp. 201-205, 1972.
29. IBM System/360, Scientific Subroutine Package, Programmer's Manual, H20-0205, 1966.
30. Klecka, W. R., "Discriminant analysis," in Statistical Package for the social science, N. H. Nie, C. H. Hull, J. G. Jenkins, K. Steinbreuner and D. H. Bent (eds.), New York : McGraw-Hill, pp.434-467, 1975.
31. Knoll, Arnold L., "Experiments with 'Characteristic Loci' for recognition of handprinted characters," IEEE Trans. computer, pp. 366-372, April 1969.
32. Kwan, C. C., "A study of the selection and recognition of handprinted characters," Master of Computer Science Thesis, Concordia University, 1977.
33. Kwon, S. K. and D. C. Lai, "Recognition experiments with handprinted numerals," Joint Workshop on Pattern Recognition and Artificial Intelligence, pp. 74-83, 1976.
34. Lachenbruch, Peter A., Discriminant Analysis, Hafner Press, 1975.
35. Munson, J. H. "Experiments in the recognition of hand-printed text : Part I -- Character Recognition," Fall Joint Computer Conference, AFIPS conference proceedings, vol. 33, Washington, D. C. : Thompson, pp. 1125-1138, 1968.
36. O' Callaghan, J. F.; "A coding approach to pattern recognition," Pattern Recognition, vol. 2, pp. 199-214, 1970.



37. Oram, D. O., " Optical character recognition," Honeywell Computer Journal, vol. 2, #3, Fall, 1968.
38. Pavlidis, T., Structural Pattern recognition, vol. 1, New York, Springer, 1977.
39. Pávlidis, T. and F. Ali, " Computer recognition of handwritten numerals by polygonal approximations," IEEE Trans. Systems, Man, and Cybernetics, vol. SMC-5, #6, pp. 610-614, Nov. 1975.
40. Pavlidis, T. " Polygonal approximations by Newton's method," IEEE Trans. on computers, vol. C-26, #8, pp. 800-807, August 1977.
41. Persoon, Eric and K. S. Fu, " Shape discrimination using Fourier descriptors," IEEE Trans. on Systems, Man, and Cybernetics, vol. SMC-7, #13, pp. 170-179, March 1977.
42. Rao, T. CH. Malleswara, " Feature extraction for fingerprint classification," Pattern Recognition, pp. 181-192, July 1976.
43. Richard, C. W. and H. Hemami, " Identification of three-dimensional objects using Fourier descriptors of the boundary curve," IEEE Trans Systems, Man, Cybern., vol. SMC-4, pp. 371-378, July 1974.
44. Sheinberg, I., " Optical character recognition for information management," Pattern Recognition, L. N. Kanal (ed), pp. 31-40, Washington, D. C. : Thompson, 1968.
45. Siy, Pepe and C. S. Chen, " Fuzzy logic for handwritten numeral character recognition," IEEE Trans. Systems, Man, and Cybernetics, pp. 570-575, Nov., 1974.
46. Stefanelli, R. and A. Rosenfeld, " Some parallel thinning

algorithms for digital pictures," JACM. 18, 2, pp. 255-264, 1971.

47. Stevens, M. E., " Automatic character recognition - A state-of- the-art report," National Bureau of Standards, Tech. Note 112, Washington, D. C., 1961.

48. Suen, C. Y., " Optical character recognition - the state of the art," Canadian Datasystems, vol. 6 , pp. 40-44, May 1974.

49. Suen, C. Y., " Advances in optical character recognition," Proc. Canadian Computer Conference , pp. 263-268, May 1978.

50. Suen, C. Y., M. Berthod and S. Mori, " Advances in recognition of handprinted characters," Proc. Fourth Int. Joint Conf. on Pattern Recognition, Nov. 1978.

51. Takeshi, A., N. Masayuki, and C. Yoshimurau, " Recognition of handprinted characters using the information processing of close planar curves," Systems, Computers and Controls, vol. 7, #6, pp.67-75, 1976.

52. Triendl, Ernst E., " Skeletonization of noisy handwritten symbols using parallel operations," Pattern Recognition, vol. 2, pp. 215-226, 1970.

53. Tou, J. T. and R. C. Gonzalez, Pattern recognition Principles, New York : Addison-Wesley, 1974.

54. Toussaint, G. T. and R. T. Donaldson, " Algorithms for recognizing contour-traced handprinted characters," IEEE Trans. Computers, pp. 541-546; June, 1970. vol. 1, # 1, pp. 43-65, 1972.

55. Udupa, K. J. and I. S. N. Murthy, " Some new concepts for encoding line patterns," Pattern recognition, vol. 7, pp.

225-233, 1975.

56. Unger, S. H., " Pattern detection and recognition," Proc. IRE, vol. 47, pp. 1737-1752, Oct., 1959.

57. Zahn, Charles T. and R. Z. Roskies, " Fourier descriptors for plane closed curve," IEEE Trans. Computers, vol. C-21, #3, pp. 269-281, March 1972.