COMPREHENSIVE STUDY OF THE TECHNIQUES USED FOR ONLINE HANDWRITING CHARACTER RECOGNITION

Thesis submitted in partial fulfillment of the requirement for

The award of the degree of

Masters of Science

In

Mathematics and Computing

Submitted by

RAMANJEET KAUR

Roll no. - 30703017

Under

the guidance of

Mr. Rajiv Kumar

JULY 2009

School of Mathematics and Computer Applications

Thapar University

Patiala-147004 (PUNJAB)

INDIA

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled ***"Comprehensive Study Of The Techniques Used For Online Handwriting Character Recognition"*** *in partial fulfillment of the requirements for the award of degree of Master of Science, School of Mathematics and Computer Applications, Thapar University, Patiala is an authentic record of my own work carried out under the supervision of Mr. Rajiv Kumar.*

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

Ramanjeet Kaur
(Signature of Student)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Rajiv Kumar

(Mr. Rajiv Kumar)

Supervisor
SMCA,Thapar University

Patiala

Countersigned by:

SSBhatia 15.7.09

Dr. S.S. Bhatia

(Professor & Head)

School of Mathematics & Computer Applications

Thapar University, Patiala

RKSharma 22/7/09

Dr.R.K.Sharma

Dean of Academic Affairs

Thapar University

Patiala

ABSTRACT

Online automatic recognition of handwritten text has been an on going research problem for four decades. It has been gaining more interest lately due to the increasing popularity of hand held computers, digital note-books and advanced cellular phones, more accurate electronic tablets, more compact and powerful computers, and better recognition methods. In Online system input is taken as the (x,y) co-ordinates of the sampled points along with the time information. One can extract the order of the writing (where did the writer start, and where did he or she stop writing, what direction did the pen take) and also the speed of the writing.

The first part of my work gives an introduction to handwriting recognition. The topics considered include; types of handwriting systems; difference between *Online* and *Offline* handwriting recognition; three different approaches to online handwriting recognition; online handwriting recognition system; and finally literature survey.

Second part explains the online handwriting recognition method based on dominant points in strokes. This technique is based on sequential handwriting signals. In this approach, an online handwritten character is characterized by a sequence of dominant points in strokes and a sequence of writing directions between consecutive dominant points. The directional information is used for character pre-classification and positional information is used for fine classification. Both pre-classification and fine classifications are based on dynamic programming matching using the idea of band limited time warping.

Third section discusses about the prototype (template) based online handwriting recognition method. An online handwriting system must be able to recognize a wide variety of handwriting styles, while attempting to obtain a high degree of accuracy when recognizing data from any one of those styles. As the number of writing styles increases, so does the variability of the data's distribution. We then have an optimization problem: how to best model the data, while keeping the representation as simple as possible? If we can identify N different styles of writing individual characters (referred to as lexemes), these can then be modeled as N relatively simple independent distributions. This technique used a string matching distance measure for the

recognition of online handwriting which takes the advantage of lexemes to reduce the number of prototypes that must be stored. A method of lexeme representation is shown.

Fourth part presents the online handwriting recognition using structural based method, in which structure of the input is extracted and finally matched with the structure of models already stored in a model database to determine the class of input character by using elastic structural matching.

In the last, I have discussed comparative study on the basis of advantages and disadvantages of all the techniques presented in this work.

ACKNOWLEDGEMENT

I would like to thank the following people for providing assistance, support, encouragement and inspiration during this work.

Firstly, I would like to thank my supervisor, **Mr. Rajiv Kumar, Lecturer, School of Mathematics and Computer Applications, Thapar University, Patiala**. He has provided guidance encouragement, opportunities, and knowledge at a level that few advisors are capable of. His dedication to his students and his field is a true inspiration. He has devoted a lot of time in exploring the terms used in my present work. He gave suggestions one step ahead to me.

I would also like to extend my special thanks to **Dr. S.S. Bhatia, Prof and Head, School of Mathematics and Computer Applications, Thapar University, Patiala**, providing help and necessary facilities in the department and directly or indirectly encouraging me to work harder during the whole course.

I would like to give special thank to my parents sister and the rest of the members of my family. They have always been there to support me, and I am very grateful.

Finally I would like to thank all my friends for all possible help.

(Ramanjeet Kaur)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Electronic tablets accurately capture the x-y coordinate data of pen-tip movement. Their advent in the late 1950s precipitated considerable activity in *On-line Handwriting Recognition*. This intense activity lasted through the 1960s, ebbed in the 1970s. And is being renewed now in the 1980s. The renewed interest in *On-line Handwriting Recognition* stems from a number of factors. Compared to the 1960s we now have more accurate electronic tablets, more compact and powerful computers, and better recognition algorithms. However there are additional and perhaps more important reasons. First, the recent hardware advance of combining tablets and flat displays brings input and output into the same surface. This, combination permits the use of electronic ink, providing feedback to the writer of the digitized writing. Electronic ink is the instantaneous display of the trace of the motion of the stylus tip directly under the stylus. Second, efforts in automating office work have increased interest in more natural methods of entering data into machines

1.0 HANDWRITING RECOGNITION

Traditionally, interactions between humans and computers have been based on a display, printer, keyboard, and pointing device. However, a keyboard can be very inconvenient when the device is only slightly bigger or the same size as the human palm. In addition, a keyboard is difficult to integrate in small devices and it usually determines the size of the whole apparatus. This is especially true when the number of the characters is very high as in Chinese and Japanese languages. A pointing device, for example a track ball or a pen, is insufficient or very slow when used alone in applications in which textual input is also desired.

Because of these problems, new methods for input have been developed. *Handwriting Recognition* is the more attractive input method [1].

At first sight, *Handwriting Recognition* does not appear to be a difficult problem. A recognition system should just choose the correct answer, usually the one that most resembles the written one, from a limited set of characters. Unfortunately, this approach faces a number of difficulties. The most prominent problem in handwriting recognition is the vast variation in personal writing styles. There are also a lot of variations within a writing style of one person. These variations

depend for example on the context of the writing, writing equipment, writing situation, and the mood of the writer. The writing style may also evolve with time or practice. The performance of the automatic recognition system thus depends heavily on how well the different personal writing styles and their variations are modeled. A recognition system should be insensitive to meaningless variations and still be able to distinguish different but sometimes very similar looking characters.

1.1 TYPES OF HANDWRITING RECOGNITION SYSTEM

A Handwriting Recognition system can be either "online" or "offline." According to the mode of data acquisition, handwriting recognition methodologies are categorized into two systems as *ONLINE CHARACTER RECOGNITION SYSTEMS* and *OFFLINE CHARACTER RECOGNITION SYSTEMS.* The set of characters, the writing style variations [2], [3] and the constraints on the writing [4] have a great influence on recognition methods. The distinctions re described in the following sections in more detail.

Off-line Character Recognition

The Handwriting Recognition task heavily depends on the application it is used. First applications were systems that recognized text that was originally written on the paper. In such systems paper sheets are digitized to get two dimensional images. Features for recognition are then first enhanced and then extracted from the bitmap image by means of digital image processing. This type of recognition is called *Offline Recognition* as the text is not recognized same time as it is produced but after the writing task is completed. Offline methods are not suitable for man machine communication because they do not facilitate real time interactivity. They are suitable for automatic conversion of paper documents to electric documents which then may be interpreted or post processed by computers.

On-line Character Recognition

In the case of *Online Recognition*, text is input with special equipment and it is recognized in real time. Input media is usually a tablet or a flat display that can capture information on the location and the motion of a pen or some other pointing device moving on its surface. Writing can be done with any ordinary pen or a pen like device specially designed for its purpose. Location and

possibly also pressure data are frequently sent to the computer which performs the recognition task.

Online handwriting recognition assumes that a transducer device is connected to the computer and is available to the user. The transducer converts the user's writing motion into a sequence of signals and sends the information to the computer. The most common form of the transducer is a tablet digitizer. A tablet consists of a plastic or electronic pen and a pressure or electrostatic-sensitive writing surface on which the user forms one's handwriting with the pen. Sampling the movement of the pen-tip, the digitizer is able to detect information like x and y coordinates of a sampled point, the state of whether the pen touches the surface (pen-down) or not (pen-up).The information is sent to the connected computer for recognition processing (Figure 1.1). A "stroke" in on-line data is defined as sequence of sampled points from the pen-down state to the pen-up state of the pen, and the completed writing of a character consists of a sequence of one or more strokes. A"digital ink" is the display of the strokes on the computer screen. By digital ink, the user can see what he or she writes on the tablet and it provides interactivity between the user and the computer.

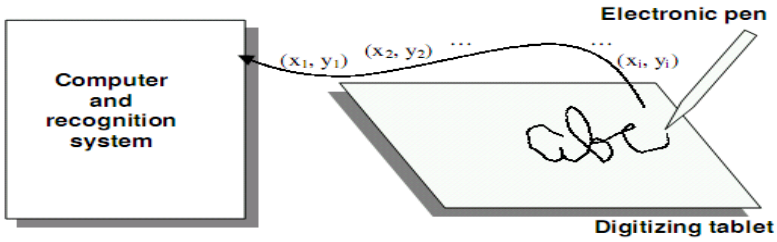


Figure 1.1 A tablet digitizer, input sampling and communication to the computer.

1.2 ADVANTAGES OF ONLINE CHARACTER RECOGNITION OVER OFFLINE CHARACTER RECOGNITION

- Online Handwriting Recognition means that the machine recognizes the writing while the user writes .The term real time or dynamic has been used in place of Online. Offline Handwriting Recognition, by contrast, is performed after the writing is completed. It can be performed days, months, or years later.

- An advantage of Online devices is that they capture the *dynamic information* of the writing.This information consists of the number of strokes, the direction of the writing for

each stroke, and the speed of the writing each stroke. Online transducers capture the trace of the handwriting or line drawing as a sequence of coordinate points. By contrast, offline conversion of scanner data to line drawings usually requires costly and imperfect preprocessing to extract contours and to thin or skeletonize them .

- Another advantage of Online over Offline data is the availibility of the *stroke segmentation* and order of writing,this is bacause pen devices include the ability to detect the states pen-down(when the pen is touching the tablet) and pen-up(when the pen is not touching the tablet.

- In dynamic handwritten data ,the trace of the writer's pen is stored as a sequence of points sampled at equally spaced time intervals. The information captured for each sample is the(x,y) coordinates of the pen on the digitizing tablet. While this pen tarce could be used to construct a static image of the writing,thus aloowing traditional OCR techniques to be applied.

- Another advantage of Online Recognition is *interactivity* .In an editing application, for example, the writing of an editing symbol can cause the display to change appropriately. Also, recognition errors can be corrected immediately.

- Yet another advantage is *adaptation.* When the user sees that some of his characters are not being accurately recognized, he can alter their drawing to improve recogniion. Thus, the user adapts to the recognition system.

The present is with reference to Online Character Recogintion. So now it is better to discuss this system first.

1.3 ONLINE CHARACTER RECOGNITION SYSTEM

Online character recognition procedure is basically very simple:after preprocessing of the raw data,some features are extracted from the unknown character which is then classified to the class whose members have the most similar features(figure 1.2).

Online character recognnition system is explained below:

1. **Raw Data:** a typical format of on-line handwriting data is a sequence of coordinate points of the moving pen point. In addition to location, the pressure between the pen point and writing surface can be measured. Sometimes, not only the events when the pen point is actually touching the writing surface are detected and recorded but handwriting data is collected also when the pen point is hovering just above the writing surface. Connected parts of the pen trace in which the pen point is touching the writing surface, or the pressure between them exceeds some threshold value, are called strokes. (Note that sometimes trace segments defined not only by pen lifts but also by other critical points of the pen trace such as curvature maxima, velocity minima, and local horizontal minima are called strokes). If the touching detection is too sensitive and is activated with too low pressures, hooks and retraces are likely to appear in the beginnings and the ends of strokes and some strokes might be unintentionally connected . On the contrary, if the pressure limit is set too high for touching detection, strokes tend to break accidentally. The pen trace is usually sampled with a constant rate and thus data points are evenly distributed in time but not in space. When the speed of writing is slow, the sample points are located densely on the true pen trace, whereas quick writing produces more sparsely located points. Typically, writing speed slows down in sharp corners, in other points of extremal curvature, at the beginning and the end of the stroke, but also by hesitation and pausing of the writer .Sampling rate and resolution should be so high that the sampled data points represent the true pen trace faithfully. Naturally, the selection of suitable level of sampling rate and resolution depends on the writing speed and the scale of the meaningful pen trace features. If sampling rate is too low, odd corners will be introduced on the sampled pen trace and some of the real corners and miniscule trace features can be missed. In practice, sampling resolution has to be definite and that causes some errors in the recorded pen point locations. If sampling resolution is too low, the sampled pen traces are jagged. Data collection hardware can sometimes introduce erroneous points clearly out of the real pen trace. Such points are called wild points. Wild points can be introduced for example when the writer rests his or her hand on the pressure sensitive writing surface. Figure 1.2 shows some examples of characters collected with different sampling rates and resolutions. Characters in the upper row are collected with sufficiently high resolutions as there is no jagging in the captured pen traces. Characters in the lower

row are collected by using much poorer resolution and the captured pen traces are clearly jagged. The sampling rates seem to be high enough for all but perhaps the two last characters in the upper row as the captured pen traces in those cases have some corners which most probably were not in the real pen traces. Effects of varying writing speed on data point density can be seen most clearly in the second character on the left in the upper row.

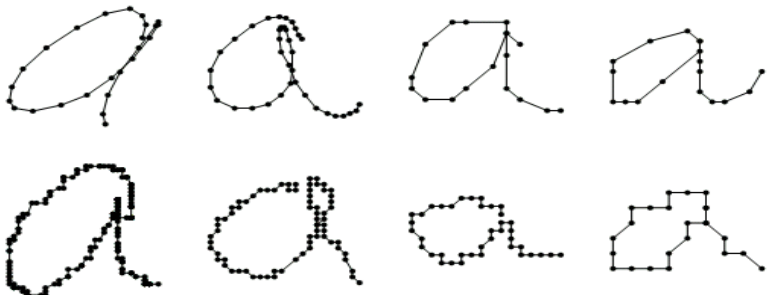


Figure 1.2: Examples of character collected with different temporal sampling rates and spatial resolutions.

2. **Preprocessing:** the goal of preprocessing is to discard irrelevant information that can negatively affect the recognition .Generally, the irrelevant information includes duplicated points, wild points, hooks, noise, etc. The whole approach is summarised in Figure1.3. The approach consists of removing duplicated points and hooks, smoothing data, and performing normalisation, as epalined below.

   - **Removing duplicate points :** Duplicated points can be removed by checking whether the coordinates of any two points are the same. If they are in the same position, one of them is kept and others are removed. After the duplicated points have been removed, the process starts removing hooks in the strokes.

   - **Elimination of Hooks:** usually,hooks occur at the beginning and/or at the end of a stroke. In addition ,hooks are usually characterized by their small length and their large changed angular variations. Based on their locations and their small length and sharp turn angles,this technique uses the equal length technique to interpolate and adjust points, and then uses the detection method of the sharp points. Finally the method physically removes the hooks of stroke,based on the sharp points.

- **Normalization:** the goal of the normalization is to remove some of the variations of handwriting styles and to simplify the shapes of symbols. Usually it includes two of the important procedures:scaling and transition.

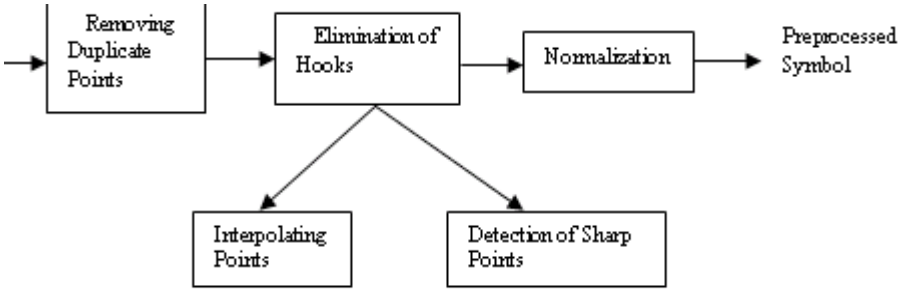


Figure 1.3:Preprocessing Steps

3. **Feature extraction:** it is a very crucial step, as the success of a recognition system is often attributed to a good feature extraction method. The Feature extractor determines which properties of the preprocessed data are most meaningful and should be used in data stages. These features may be velocity,acceleration,structural primitives, curvature data of the stylus derived from the handwriting input and structural primitives. The resulting feature vector is supposed to be the best possible representation of the input.The various methods for the extraction of feature vectors are expalined in the coming chapters.

4. **Classification:** The classification is the most important part of the online recognition system and uses the features extracted in the previous stage to identify the input character according to preset rules.

Digitizing

Raw Data

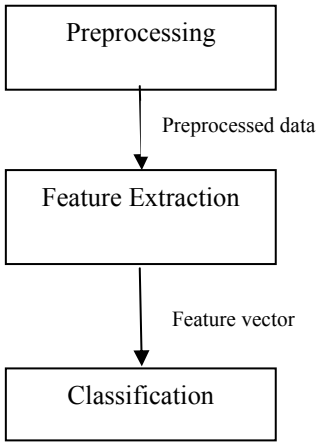


Figure 1.4 : Stages of On-line Handwriting Recognition

1.4 ONLINE CHARACTER RECOGNITION METHODS

The On-lineCharacter Recognition problem has been tried to solve with many different methods. Some of the methods are especially developed for character recognition but most of them are loaned from other fields of pattern recognition,signal processing,and image analysis.Recognition methods included in the present study are based upon *dominant points*, *prototypes* and *flexible structural matching*, and brief introduction to these techniques is as follows:

On-line Charcter Recognition using Dominant Points In Strokes

The main idea of this technique is basically to come up with a approach to on-line handwritten character recognition based on sequential handwriting signals. In this approach,an online handwritten character is characterized by a sequence of dominant points in strokes and a sequence of writing directions between the consecutive dominant points. In particular points in strokes corresponding to *local extrema* of curvature,*turning points* in online handwriting curves are detected ,and finally the classification has been done by using *band limited time warping* and *dynamic programming* .

Prtotype(Template) Based Online Handwriting Character Recognition

Handwriting is a common,natural form of communication for humans,and therefore it is useful to utilize this modality as a means of input to machines. One well known method of classifying individual characters is *prototype based matching*.In prototype-based methods, the classifcation

of an unknown handwriting sample is performed by matching it to a set of known samples, the prototypes, and selecting the class according to the most similar prototypes, This method is demonstrated for online handwriting recognition where the number of representative different styles of writing a particular character. The prototypess are then used a reference for efficient classification using decision trees.or in other way prototype based method described in this work focuses on a representation of characters that models the different writing styles.

Flexible Structural matching for Online Charcter Recognition

In the classical structural approach for OLCR(Online character Recognition),structure of input character is extracted and matched with the structure of the models stored in the model database. If a match is found the class of the matched model is given as the result of recognition for input character .In structural approach recognition accuracy largely depends on the quality of the model database,capability of primitives used in representing the different patterns uniquely ,as the structure of a character is a sequence of primitives with their direction information. Also due to flexibility, those characters which do not have an exact match varied in shape and direction in order to find approximate matching.

1.5 LITERATURE SURVEY

Character recognition has been an active research area for more than 30 years Different approaches, such as using dominant points in strokes,prototype based, structural based , and neural network aproaches, have been proposed.

B.Q.Huany, Y.B.Zhang and M.T.Kechdi[5], C.C.Tappert, Suen and T.Wakhara(1988) [6],proposed preprocessing techniques for Online Handwriting.The approach is to remove duplicate points,hooks,smoothing data and normalization.

Xiaoline.Li and Dit-yan yeung(1996) [7],Rudy Actipranta [8] have given a online handwritten alphanumeric character recognition technique using dominant points in strokes,in which pen-up,pen-down points,points coreesponding to local extrema of curvature are considered as dominant points. Su Yang and Guozhong[9] have given the idea of turning points in online handwriting curve as dominant points.

Elastic matching(C.C.Tappert(1984) [10]) works on the sequence of sample points directly by searching for an alignment of data points between an input character, and some template character .Duin et al. [11],Scott.D.Connel and Anil.K.Jain [12] have referred to such data as "featureless". The distance between an input character and a template is taken as the sum of distances between aligned points .Classification can then be done using a nearest-neighbor classifier(Fahim A.M, Ramandan M.A [13], Salem Renals [14], A.K.Jain and R.C.Dubes [15]),in which there is given a set of n data points in d-dimentional space and an integer Kand the problem is to determine a set of K points,called centers,so as to minimize the mean squared distance from each data point to its nearest centre .A "featureless" representation of offline characters has been demonstrated by Jain and Zongker [16], in which they used deformable models to find the similarity between characters.

Approach adopted by Jain et al. [17], each event is represented with three measurements: the x and y offsets with respect to some reference coordinate, and the angle of curvature of the written stroke at the sample point .We define the reference coordinate as the first sample point of the digit's first stroke.

Wakhara and Odaka(1997) [18] have developed a linear curve matching method for isolated characters which uses a distance completely different from the one used in te matching method described .The variations between the unknown character and the model character are explained by the stroke based affine transformation(SAT) components and the residual components.The SAT components are caused affine transformations,such as translation,rotation,scaling,and shearing,and they represent interclass variations,where the residual components represent interclass variations.

C.C.Tappert,suen,Toru Wakhara(1990) [1] have developed a curve matching method in which curves from an unknown are matched against those of prototype characters,and the name of the best matches the unknown is assigned to the unknown The curves matched are usually functions of time.

Elastic matching which is also called dynamic time warping(DTW), is a nonlinear matching method originally used in speech recognition. It was developed in the beginning of 1970s and was introduced as an handwriting recognition method in late 1970s. Similar methods have also

been used sucessfully in online signature verification(Martens and Claesens 1997 [19]),(Sato and Kogure 1982 [20]).A very basic elastic matching method is introdused by Tappert(1984)[10] ,Li and Yeung(1997) [21]. The method consists of three modes of operation: training, recognition, and updating.

Structural and Syntactical methods are used for various pattern recognition tasks in which the structure of the pattern is the paramount and can be used for classification. As structural approach,one of the most widely used chain coding method is Freeman's chain code [22]. It consists of eight values, 0-7, which indicate how the current point is connected to the next one.

Hung yuen [23] also given a chain coding approach for real time online recognition,in which a new type of generlized chain code(GCC) is proposed for lossless incoding of data variably spaced handwritten data.

The extended Freeman encoding schemes(Mustafa M.hammad[24]) uses the eight direction values in Freeman's chain code as primitives . Al-taani Ahmed[25] discussed the feature extraction algorithm used to extract the primaryands econdary features.then he has given overview of proposed recognition approach.

Chan et al.[26] proposed a syntactic approach to structural analysis of online handwrittem mathematical expressions .The author used definite clause grammar (DCG) to define a set of replacement rules for parsing mathematical expressions. They also proposed some methods to increase the efficiency of the parsing process. The authors tested the proposed system on some commonly seen mathematical expressions and they claimed that their method has achieved satisfactory results,making mathematical expression recognition more flexible for real world aplications.

Chan et al.[27] discussed a structural approach for recognizing on-line handwriting. The recognition process starts when getting a sequence of points from the user and then by using these points to extract the structural primitives .These primitives include different types of line segments and curves .The authors demonstrated their approach on 62 character classes (digits, uppercase and lowercase letters).

Chan and Yeung(1998) [28],C.C.Tappert(1990)[1] have proposed a structural method in which the structure of the unknown pattern can be deformed if if does not directly match to any of the structures found in the model base .The method is called Elastic Structural Matching.the structure found in the characters are :a line segment,a counter clockwise and clockwise curve,a loop and a dot .The recognition is done on the basis of the first matching modal .At first all the original pattern structure is matched against all the structures of the models.If no matching model is found,the structure of the pattern is modified.

CHAPTER 2

ONLINE CHARACTER RECOGNITION USING DOMINANT POINTS IN STROKES

ONLINE CHARACTER RECOGNITION USING DOMINANT POINTS IN STROKES

2.0 INTRODUCTION

In this technique an online handwritten character is characterized by a sequence of dominant points in strokes and a sequence of writing directions between consecutive dominant points. The directional information of the dominant points is used for pre-classification and the positional information is used for fine classification. Both pre-classification and fine classification is based on dynamic programming matching using the idea of band limited time warping.

The remainder of this technique is divided into two sections. Section 2 describes feature extraction. Section 3 presents pre-classification and fine classification algorithms.

This approach to online handwriting character recognition is based on sequential handwriting signals. The whole recognition process of this technique is described by the following figure 2.1

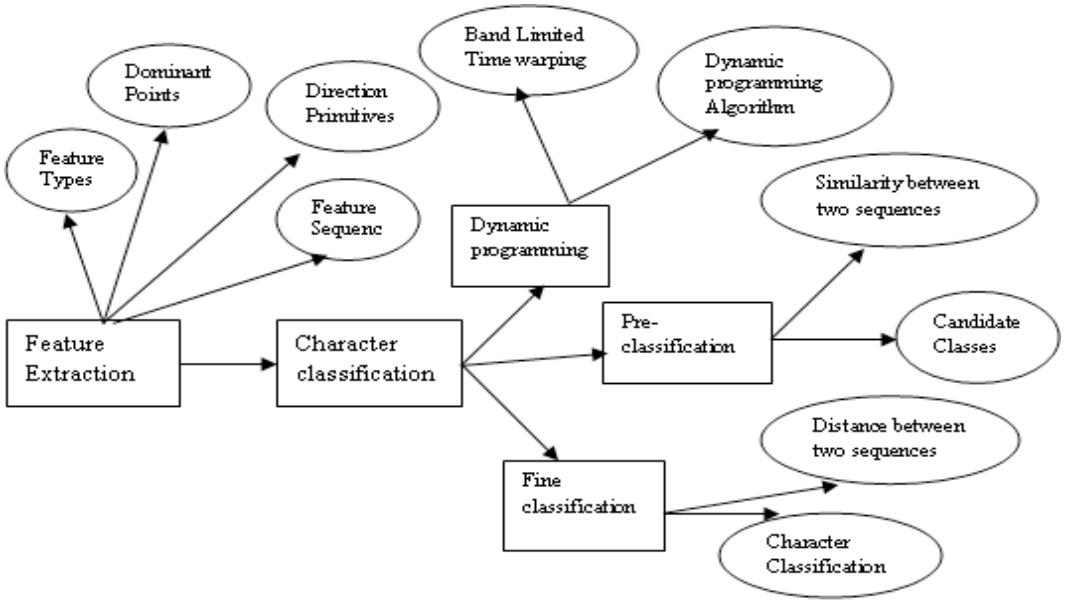


Figure 2.1: Different stages of the recognition process

2.1 FEATURE EXTRACTION

The main idea here is to come up with an efficient and reliable method for extracting features from the sequential handwriting signals .In this approach these feature points, dominant points detection is essential to pen based human-computer interaction. Many recognition based applications, for example, pen gesture recognition ,graphics recognition for computer-aided design, and handwriting recognition, are on the basis of *dominant point* .In particular, points in strokes corresponding to *local extrema* of the curvature or gradually or sharply *turning points* of the online handwritten curves are detected. Figure 2.2 illustrates the concept, where the dominant points are indicated by circles.

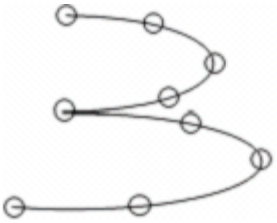


Figure 2.2: Dominant points

2.1.1 Dominant points

Dominant points are points that one of the next 2 types:

Type 1 includes points that one of the next types: (a) start point and final point of a stroke (or pen-down and pen-up points); (b) local extreme; (c) middle point that connected the previous two points in (a) and (b) [7].

Type 2 includes points corresponding to gradually or sharply turning points in the Online Handwriting curve.

Method to find Type 1 dominant points

(a)Pen-up and Pen-down points

A pressure sensitive device on the pen itself detects the beginning (Pen-up event) and end (Pen up event) of each stroke.

(b)Local Extreme

If there is a function $f(x)$ representing the handwriting curve and $f'(x1) = 0$, where $x1$ is a number including in domain of f, so x_1 will caused the value of the function maximum or

minimum. And $T(x_1, f(x_1))$ is called as critical point of the function $f(x)$. Local extreme includes ***Local Maximum*** and ***Local Minimum*** .Local means that the point is extreme for some interval in the domain function [29].

Local Maxima and Local Minima

To determine whether an extreme point is local maximum or local minimum or the others, we need to focus on the changing of derivative sign .If the sign changes from plus to minus or minus to plus, the point must be local maximum or local minimum. On the other hand, the point is not local maximum and not local minimum.

- $(x_1, f(x_1))$ is local maximum if $f'(x_1) > 0$

  For $x < x_1$ and $f'(x_1) < 0$ for $x > x_1$.

- $(x, f(x_1))$ is local minimum if $f'(x_1) < 0$ for $x < x_1$ and $f'(x_1) > 0$ for $x > x_1$

- $(x_1, f(x_1)$ is not local maximum and not local minimum if $f'(x_1) > 0$ for $x < x_1$ and $x > x_1$,or $f'(x1) < 0$ for $x < x_1$ and $x > x_1$.

Therefore it can be said that a point is called local maximum if in that point the function change from increasing to decreasing and local minimum if the function change from decreasing to increasing .Local maximum and minimum will have extreme value just in some intervals of domain function .The other way, the value of global maximum will be the largest for all domain function and the value of global minimum will be the smallest value for all domain function .And therefore these points corresponding to Local Extreme are called Dominant points.

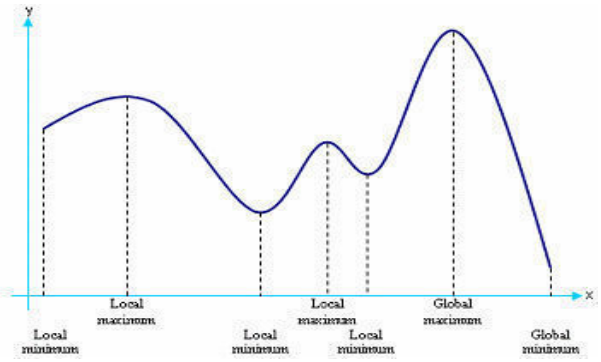


Figure 2.3: Local Maximum and Minimum Extreme

Method to find Type 2 dominant points

An on-line handwritten curve is composed of a point sequence $P_1, P_2 \cdots P_i$. Here a *dominant point is defined as either a gradually or a sharply turning point.* Firstly, we calculate the polygonal area enclosed by the curve of interest. Then, we define a variable that is the ratio of the polygonal area to the square of the corresponding chord of a curve. As illustrated in Fig. 2.4, we can imagine a rectangle whose bottom is the chord of this curve and area is equal to the polygonal area. The variable that we just defined is equal to the ratio of the height to the width of the imagined rectangle .If this ratio is bigger than a given threshold, then, we can decide that an obvious turning trend takes place. This decision criterion takes both the fluctuation amplitude and the scale of a curve into account simultaneously so that what is captured is the turning trend not the local curvature details of a curve. Only when the fluctuation on a curve is large enough compared with the scale of this curve, it can be decided that the curve contains turning points.

The computing method is summarized as follows .Let A_i represent the polygonal area enclosed by $\{P_1, P_2 \cdots P_i\}$ and $\|P_i P_j\|$ the length of $P_i P_j$.

The variable to indicate a turning trend is

$$R = H_i / \|P_1 P_i\| \quad , \qquad (1)$$

Where $H_i = A_i / \|P_1 P_i\|$ (2)

is the height of the imagined rectangle as formerly discussed via Figure 2.4, and R is the ratio of the height to the width of the imagined rectangle. If R exceeds a given threshold T, which is empirically determined on the basis of extensive tests, it can be decide that a turning point has occurred. In this case, we should search it. For the value of the variable changes with each new incoming point, we know that the turning point must be near to the latest incoming point P_i. So, it is not necessary to search all the point sequence. We only need to search a very small region $\{P_i, P_{i-1} \cdots P_{i-M}\}$, where M is a small number. Let d_{i-j} represent the distance from point P_{i-j} to line $P_i P_{i-M}$ .If

$$D_{i-j} = \max\{d_i, d_{i-1}, \cdots d_{i-M}\} \quad (3)$$

Then, we select P_{i-j} as a dominant point. For the above computations just involve $\{P_1, P_i, P_{i-1} \cdots P_{i-M}\}$, only a few of points should be kept in memory. So, the storage requirement of this algorithm is very low. After the dominant point P_{i-j} has been detected,$\{P_1, P_2, \cdots P_{i-j-1}\}$, which are the points preceding P_{i-j},should be deleted, and P_{i-j} is taken as the new starting point to conduct a new decision procedure as stated above. If $R < T$, we should do nothing but wait for the next point .When a new point P_{i+1} come in, the value of R should be adjusted by the following procedure. Firstly, we calculate the area of the triangle $\Delta P_1 P_i P_{i+1}$ via the well-known equation

$$S_i = [L_i(L_i - \|P_1P_i\|)(L_i - \|P_iP_{i+1}\|)(L_i - \|P_1P_{i+1}\|)]^{1/2}, \quad (4)$$

Where S_i denotes the triangle area and $L_i = (\|(P_1P_i\| + \|P_iP_{i+1}\| + \|P_1P_{i+1}\|)/2 \quad (5)$

It is obvious that

$$A_{i+1} = A_i + S_i \quad (6)$$

This means that the area of the new polygon $\{P_1, P_2, \cdots, P_{i+1}\}$ is equal to the sum of the area of the old polygon $\{P_1, P_2, \cdots, P_i\}$ and the area of the triangle $\Delta P_1 P_i P_{i+1}$ Then we can adjust the value of R by letting $R = A_{i+1}/\|P_iP_{i+1}\|^2$ in accordance with Eq. (1) and Eq.(2). When a new point Pi+1 comes in, only $\|P_1P_{i+1}\|$, $\|P_iP_{i+1}\|$, $L_i, S_i, A_{i+1}$and R should be updated according to the above equations .The computations are so few that can be accomplished during the sampling interval.

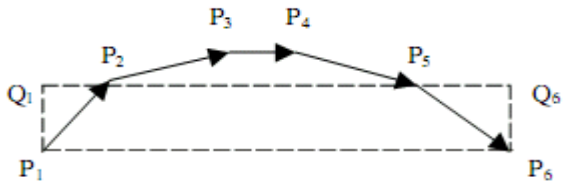


Figure 2.4 :Polygon with the imagined rectangle

2.1.2 Direction Primitives

Direction primitives use to convert moving direction into codes. There are 8 kinds of moving direction, that are E, SE, S, SW, W, NW, N, NE (Figure 2.5). The direction will code into number 0-7. After doing the coding, chain code will be obtained for every stroke (Figure 2.6).

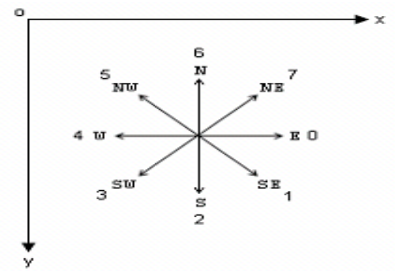


Figure 2.5: The code for 8 directions

To determine the moving direction from $A(x_a, y_a)$ to $B(x_b, y_b)$ we need to compute $\Delta x = x_b - x_a$ and $\Delta y = y_b - y_a$. If $\Delta x > 0$ then $\Delta x = 1$ and if $\Delta x < 0$ then $\Delta x = -1$ the same thing happened to Δy. Then the direction primitives d can be found depend on table 2.1.These can be defined as the vector from a dominant point to the following one.

d_i	0	1	2	3	4	5	6	7
Δx	1	1	0	-1	-1	-1	0	1
Δy	0	1	1	1	0	-1	-1	-1

Table 2.1: Determination of direction code depend on Δx and Δy

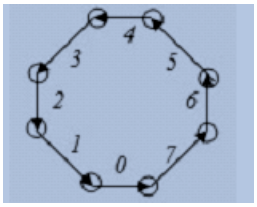


Figure 2.6: chain code

2.1.3 Feature Sequences

After feature extraction, a character C can be represented as a sequence P of dominant points and a sequence D of direction primitives:

$$P = p_1 p_2 p \cdots p_M$$

$$D = d_1 \; d_2 d_3 \cdots d_{M-1}$$

Where p_M is a dominant point, and $d_M \in$ {E, ES, S, SW, W, NW, N, NE} is the direction from p_M to p_{M+1}.

2.1.4 Example of feature Extraction

Figure 2.7 shows an example of the letter 'B' .The original handwriting with two "inked" strokes is located in the designated writing area for English letters .Local extrema of curvature in each stroke are then detected. After that, mid points between every two consecutive local extreme or pen-down/pen-up points are located .All these points are dominant points in strokes. The direction between every two consecutive dominant points is then quantized into one of the eight categories, forming the sequence of direction primitives for each stroke.

In the lower region of figure the two preprocessed strokes of the letter 'B' are displayed, with the local extrema of the curvature marked .Each stroke is also characterized by a sequence of direction primitives. The first stroke has three dominant points (no curvature extrema) with sequence '22'.The second stroke has '9' dominant points (3 extreme points) with sequence '01340134'.

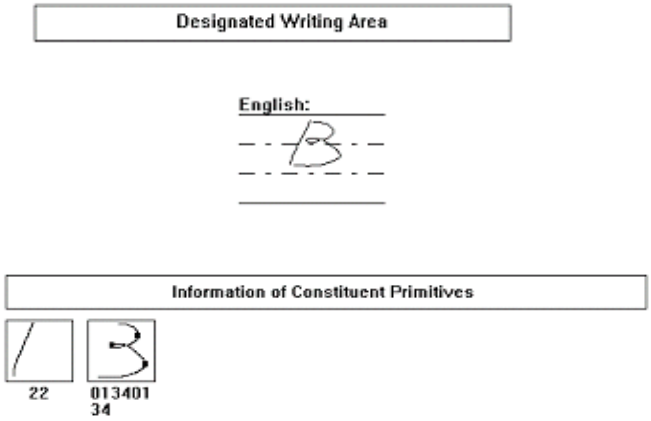


Figure 2.7:Example of feature extraction

2.2 CHARACTER CLASSIFICATION

The classification stage is divided into two steps .First, candidate selection is performed during the pre-classification step to reduce the number of candidate classes need to be investigated further in detail .Second fine classification is performed on the candidate classes that survive the pre-classification step to finally classify an input character .Both the pre-classification and fine classification steps perform matching using dynamic programming.

2.2.1 Dynamic programming for elastic matching

Band limited time warping

Time warping is a useful technique for finding the correspondence between two strings (sequences).

Given two strings, many time warps are possible. A cost (or gain) function can be defined to evaluate each warp .If time warping is intuitively seen as stretching or shrinking one string (nonlinearly) to make it look more similar to another string, then the best warp corresponds to the best compromise between minimizing the stretching and shrinking costs and maximizing the similarity between the corresponding symbols in two strings .A similarity or dissimilarity (that is distance) measure can be defined between two strings based on the best warp.

For a string of length m and another string of length n, the best warp between them can be found out using dynamic programming, and the matching process is usually represented by an m×n graph (matrix) with three basic operations(i.e. compression, expansion and substitution) associated with node [30].

Dynamic programming algorithm

Dynamic programming is a useful technique for finding the shortest path from one node to another in a graph .It is also been used successfully for string matching. In our system, we use dynamic programming approach to perform elastic matching of feature sequences.

Let's look at a particular type of shortest path problem. Suppose we wish to get from A to J in the following network (Figure 2.8).

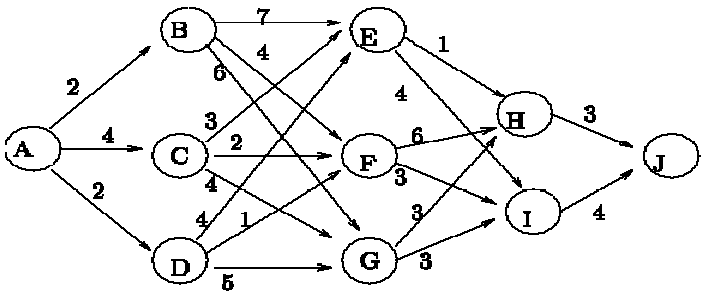


Figure 2.8 : network example for the dynamic programming

The numbers on the arcs represent distances .Due to the special structure of this problem; we can break it up into stages. Stage 1 contains node A, stage 2 contains nodes B, C, and D, stage 3 contains node E, F, and G, stage 4 contains H and I, and stage 5 contains J .The states in each stage correspond just to the node names .So stage 3 contains states E, F, and G.

Say *S* denote a node in stage *j* and, say $f_j(S)$ be the shortest distance from node *S* to the destination *J*, we can write

$$f_j(S) = \min_{\text{nodes } Z \text{ in stage } j+1}\{C_{SZ} + f_{j+1}(Z)\}$$

Stage 4.

During stage 4, there are no real decisions to make: we can simply go to destination *J*. So we get:

- $f_4(H) = 3$,by going to J.
- $f_4(I) = 3$,by going to J

Stage 3.

Here there are more choices. Here's how to calculate $f_3(f)$ .From F we can either go to H or I .The immediate cost of going to H is 6 .The following cost is $f_4(H) = 3$ .The total is 9 .The immediate cost of going to I is 3 .The following cost is $f_4(I) = 4$ for a total of 7 .Therefore, if we are ever at F, the best thing to do is to go to I .The total cost is 7, so $f_3(F) = 7$ .The next table gives all the calculations:

S_3	C $S_3Z_3 + f_4(Z_3)$		$f_3(S_3)$	Decision Go To
	H	I		
E	4	8	4	H
F	9	7	7	I
G	6	7	6	H

We now continue working back through the stages one by one, each time completely computing a stage before continuing to the preceding one .The results are:

Stage 2.

S_2	$C S_2 Z_2 + f_3(Z_2)$			$f_2(S_2)$	Decision Go To
	E	F	G		
B	11	11	12	11	E or F
C	7	9	10	7	E
D	8	8	11	8	E or F

Stage 1

S_1	$CS_1Z_1 + f_2(Z_1)$			$f_2(S_1)$	Decision Go To
	B	C	D		
A	13	11	11	11	C or D

The algorithm for dynamic programming based on a gain measure can also be derived in the similar manner.

2.2.2 Pre-classification

The preclassification step is based on finding the maximum gain for band limited time warping to define the similarity between two strings.The maximum gain is found by dynamic programming.

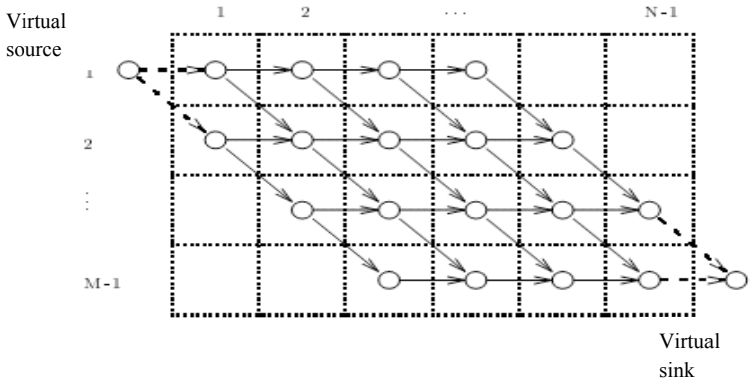
Similarity between two sequences

Take D_I and D_R be two sequences of direction primitives of an input character C_I and a reffernce character C_R,respectively.

$$D_{I=} d_{i_1} d_{i_2} d_{i_3} \cdots d_{i_{M-1}}$$

$$D_R = d_{r_1} d_{r_2} d_{r_3} \cdots d_{r_{N-1}}$$

Withoue loss of generality,assume that $M \leq N$ .Using the idea of band limited time warping, one can construct a matching graph $G_s(D_I, D_R)$ with nodes in N+1 levels,as shown in Figure 2.9.



Node (p, q) denotes matching d_{i_p} with d_{r_q}

Figure 2.9: Matching graph $G_s(D_I, D_R)$

And sink nodes are introduced so that matching of the two ends is not enforced. The gain at a node(p,q) can be interpreted as a similarity between d_{ip} and d_{rq}, which is defined in table 2.2.The gains at the source and the sink are set to zero .The maximum toatl gain G_{max} fron the source to the sink can be found by dynamic programming as discussed above.The similarity between D_I and D_R is then defined as

$$S(D_I, D_R) = \frac{G_{max}}{N-1}$$

If both $D_I$and D_R are non empty,it is obvious that $0 \geq S(D_I, D_R) \leq 1$.

	E	SE	S	SW	W	NW	N	NE
E	1	0.6	0	0	0	0	0	0.6
SE	0.6	1	0.6	0	0	0	0	0
S	0	0.6	1	0.6	0	0	0	0
SW	0	0	0.6	1	0.6	0	0	0
W	0	0	0	0.6	1	0.6	0	0
NW	0	0	0	0	0.6	1	0.6	0
N	0	0	0	0	0	0.6	1	0.6
NE	0.6	0	0	0	0	0	0.6	1

Table 2.2:Similarity between direction primitives

Candidate Classes

Suppose C_I and C_R belong to the same character set and have the same ascender descender property if the character set is the set of english letters . C_R is said to be a candidate class for

C_I if

$S(D_I, D_R) \geq T_S$ Where T_S is the threshold

Based on the above rule, we can select a (limited number of candidate classes for subsequent fine classification).in case when no candidate classes can be selected, the input character will be rejected because it is sufficiently different from all the character patterns in the current refference set.

2.2.3 Fine classification

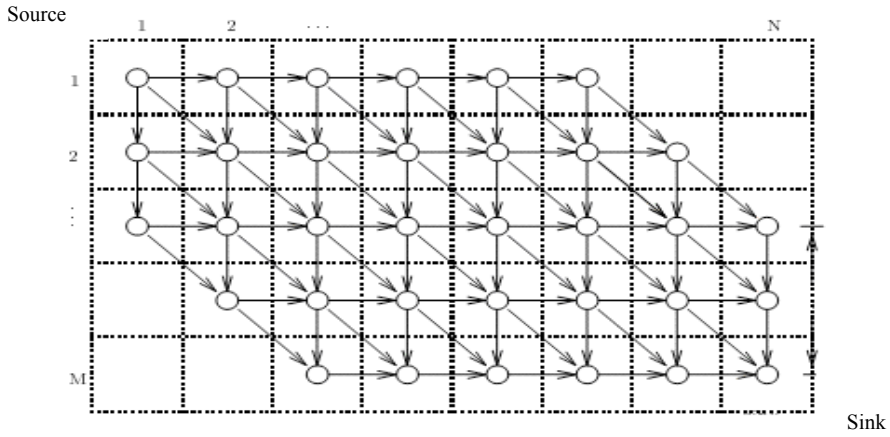
The fine classification step is based on finding the maximum cost of band limited time warping to define the distance (i.e. dissimilarity) between two strings .The minimum cost is found by dynamic programming

Distance between two sequences

Say P_I and P_R be the sequences of dominant points of the input character C_I and the reference character C_R respectively.

$$P_I = p_{i_1} p_{i_2} p_{i_3} \cdots p_{i_M}$$

$$P_R = p_{r_1} p_{r_2} \cdots p_{r_N}$$



Node (p, q) denotes matching P_{i_p} with P_{r_q}

Figure 2.10: Matching graph $G_d(P_I, P_R)$

Using the idea of band limited time warping,a matching graph $G_d(P_I, P_R)$ with nodes in N levels can be constructed(Figure 2.10)similar to that for the preclassification step.

The cost at a node (p, q) is defined as the euclidean distance between dominant points p_{ip} and p_{rq} .The minimumtotal cost C_{min} from the source to the sink can be found by dynamic probramming.With this cost,the distance between P_I and P_R is defind as

$$D(P_I, P_R) = \frac{C_{min}}{N}$$

It is obvious that $D(P_I, P_R)$ is well defind and ≥ 0 as long as P_I and P_R are both non empty.

Character classification

Let $\{P_R\}$ be the set of dominant point sequence corresponding to the set of candidate classes $\{C_R\}$ obtained from the preclassification step .The input character C_I is classified as $C_R^* \in \{C_R\}$ if P_R^* corresponding to C_R^* satisfies

$$P_R^* = \arg min\{D(P_I, P_R)\}$$

and $$D(P_I, P_R^*) \leq T_D$$

Where T_D is a threshold .Otherwise,the input character will be rejected.

CHAPTER 3

PROTOTYPE(TEMPLATE) BASED ONLINE CHARACTER RECOGNITION

PROTOTYPE(TEMPLATE) BASED ONLINE CHARACTER RECOGNITION

3.0 INTRODUCTION

In the case of prototype-based recognition systems, the better the different writing styles are covered and represented by the prototypes, the higher accuracies are achieved. However,the prototype set should not contain too many redundant prototypes as the recognition time depends linearly on its size. The prototype set can be formed automatically by applying a clustering algorithm to a large training database containing character samples written by several subjects. In such an approach, the clustering algorithm divides the training database into groups of similar character samples, or clusters, and selects a prototype(Cluster representative) for each cluster which represents all the character samples in that cluster.

The rest of the chapter is divided into three sections. Section 2 describes the string matching technique used for both lexeme representation and character classification,and Section 3 gives the data reduction and lexeme representatives ,Section 4 describes the classification using these lexeme representatives with the help of decision trees. The following Figure.3.1 shows the whole online handwriting recognition system for this technique.

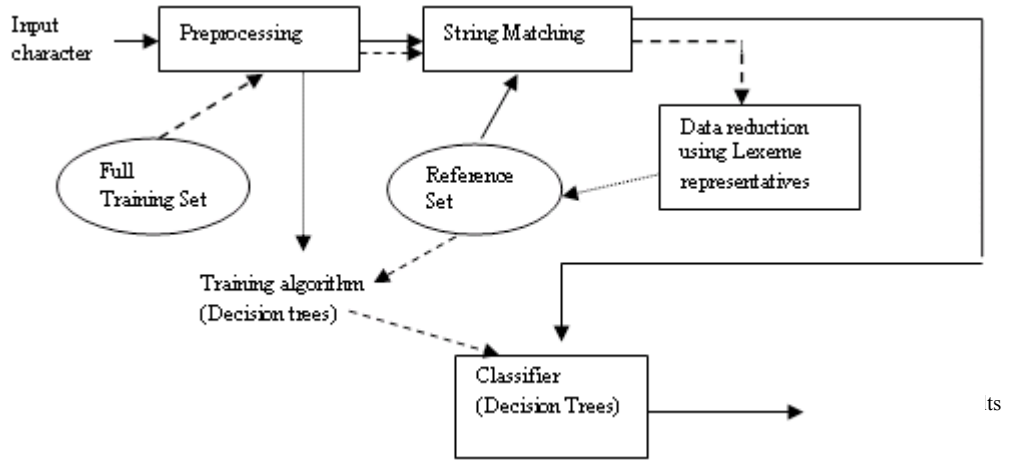


Figure 3.1: Online character recognition system. The flow of data during training is shown by the dashed line arrows, while the data flow during recognition is shown by the solid line arrows.

3.1 STRING MATCHING

A stroke is represented as a sequence of events (feature vectors, which will be described later), corresponding to the sequence of sample points in the stroke. This sequence forms a variable-length string with an average of about 62 events after preprocessing. The distance between two different strings, A and B, involves computing the distance between the corresponding pair of events e_i^A and e_j^B .This requires an alignment of the events between the two strings, and calculating the distances between the individual pairs of aligned events. This string matching technique is used to provide a distance measure between character pairs.

In this approach, adopted from [17], each event is represented with three measurements: the x and y offsets with respect to some reference coordinate, and the angle of curvature of the written stroke at the sample point. We define the reference coordinate as the first sample point of the digit's first stroke .*Table 3.1* shows this string representation for three different characters. Given an alignment of the events between two strokes, the distance, $d_E(i,j)$ between a pair of events, $e_i^A=(x_i^A, y_i^A, \Theta_i^A)$ from stroke A and $e_j^B=(x_j^B, y_j^B, \Theta_j^B)$ from stroke B,is defined as a linear combination of the respective differences between the two events of the three measurements taken for each event:

$$d_x(i,j) = \left|\left(x_i^A - x_1^A\right) - \left(x_j^B - x_1^B\right)\right| \quad (1)$$

$$d_y(i,j) = \left|\left(y_i^A - y_1^A\right) - \left(y_j^B - y_1^B\right)\right| \quad (2)$$

$$d_\theta(i,j) = \min\left(\left|\left(\theta_j^A - \theta_j^B\right), 360 - \left(\theta_i^A - \theta_j^B\right)\right|\right. \quad (3)$$

$$d_E(i,j) = \alpha d_x(i,j) + \beta d_y(i,j) + \gamma d_\theta(i,j) \quad (4)$$

where all θ are in the range (0, 360), and α, β, and γ are the weights of the linear combination. These weights are empirically determined but it is observed that placing a large weight on $d_\theta(i,j)$results in a better classification accuracy. The distance between two characters is then the sum of the distances between each pair of corresponding points from the strings, given some alignment of the points.

An alignment of the events between the two strings takes the form of a set of pairings of the events between the strings:

$$\left\{\left(e^{A}_{i^A(1)}, e^{B}_{t^B(1)}\right), e^{A}_{i^A(2)}, e^{B}_{t^B(2)} \cdots\cdots, e^{A}_{i^A(L)}, e^{B}_{t^B(L)}\right\}$$

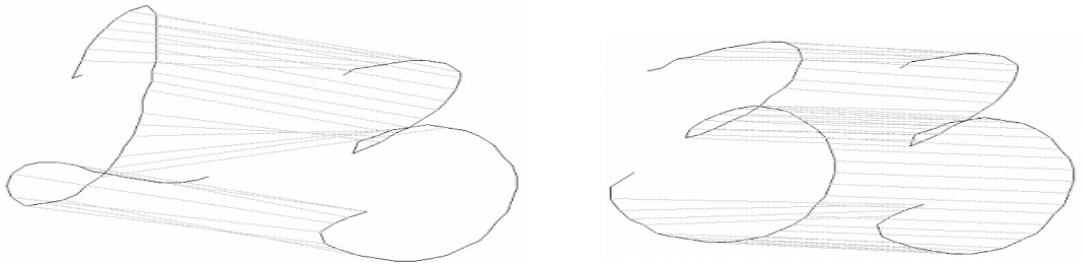
$$i^A(1) < i^A(2) < \cdots\cdots i^A(L), \quad L < N_A$$

$$t^B(1) < t^B(2) \cdots\cdots t^B(L), \quad L < N_B$$

Where N_A,and N_B are the lengths of the strings A and B,respectively,and the sequence of events from the from each string in the alignment are defind by the functions $t^A(i)$ and $t^B(i)$. Note tht the definition of an alignment allows us to entirely skip points in string $A(L < N_A)$ or $B(L < N_B)$ or both. We place a restriction on the alignment that prohibits alignments in which two or more events in one string map to a single event in the second string .In other words: $t^A(i) = t^B(j)$ if and only if $i = j$.

(0,0,223),(-2,-1,223),(-1,2,223),(0,5,182),(1,9,194),(3,13,1930,(5,16,190),(7,19,201),(10,22,239),(14,24,289), (16,21,261),(16,17,196),(16,13,180),(16,9,188),(16,5,189),(15,1,182),(15,-2,187),(14,-5,178),(13,-9,172),(13,-13,187), (12,17,193),(11,21,189),(9,-25,183),(8,-28,186),(6,-32,196),(4,-35,190),(1,-38,185),(-1,-41,207),(-4,-43,211),(-8,-44,218), (-12,-45,254),(-15,-42,255),(-16,-38,240),(-15,-34,237),(-12,-31,232),(-9,-30,220),(-5,-30,211),(-1,-31,196),(2,-3,172), (6,-34,172),(9,-35,179),(9,-35,179),(13,-36,172),(17,-37,155),(25,-36,155),(27,-35,155) (a)
(0,0,192),(2,2,192),(6,3,1920,(10,4,187),(14,5,201),(18,5,216),(22,4,232),(25,1,251),(25,-1,232),(24,-5,2000,(220,-9,199) ,(20,-12,195),(17,-15,191),(14,-18,186),(11,-20,180),(8,-23,186),(5,-25,264),((2,-27,343),(3,-23,286),(7,-21,207),(10,-19,195), (14,-18,203),(18,-17,208),(22,-18,203),(26,-19,205),(29,-21,211),(32,-24,201),(34,-28,192),(36,-31,202),(37,-35,212),(37,-39,203),(36,-43,196),(35,-47,195),(33,-50,186),(31,-54,195),(29,-57,205),(26,-59,1960,(22,-62,200),(19,-63,202),(15,-64,202), (11,-64,201),(7,-63,196),(3,-62,199),(0,-60,213),(-3,-58,2530,(-4,54,259),(-1,-51,221),(1,-49,199),(5,-47,199),(5,-47,199) (b)

Table 3.1:The string representation $(e_i = (x_i, y_i, \theta_i))$ after preprocessing of the Two digits from figure 3.2(a) the digit "2",(b) the digit "3" of Figure 3.1(a)



Distance =108 Distance =10

Figure 3.2:The alignment between the two digits, and the resulting matching score for (a) a '2'compared with a '3',(b)comparing two '3's

A restriction is placed that prohibits alignments in which two or more events in one string map to a single event in the second string. Alternate alignments are created by two methods: (1) skipping an event from the first string if it is determined that, for the given alignment, the event is spurious, (2) skipping an event in the second string if it is determined that, for the given alignment, the corresponding event in the first string is missing. In each case, we add a penalty to the total distance between the strings, respectively, called spurious penalty and missing penalty. These penalties also act as threshold values on the distance between two events in determining if a spurious or missing event exists. The distance between two strings is calculated by considering all possible alignments of events in the two strings, and finding the alignment for which the total distance is minimum using dynamic programming(expalined in third chapter) The calculation of the distance between two strings, A and B,is shown here in terms of the calculation of the distances between a set of events , e_i^A and e_j^B .

$$D(i,j) = min \begin{cases} D(i-1,j-1) + d_E(i,j) \\ D(i-1,j) + missing\ penalty, & 1 \le i \le N_A \\ D(i,j-1) + spurious\ penalty & 1 \le j \le N_B \\ D(i-1,j-1) + missing\ penalty + spurios\ penalty \end{cases}$$

$$Dist(A,B) = D(N_A, N_B)$$

For this technique missing penalty=spurious penalty. The final distance between two strings uses an additional global measurement: a stroke count difference penalty term

$$Dist_{sp}(A,B) = \frac{Dist(A,B)^2}{Norm\_Factor(N_A,N_B)} + (SP)|S_A - S_B|$$

where $S_A(S_B)$ is the number of strokes that make up digit $A\ (B)$, SP is the stroke penalty, andNorm}Factor(N_A,N_B) is the maximum possible distance between any two strings of lengths N_A and N_B scaled by a constant factor.

3.2 DATA REDUCTION AND LEXEME REPRESENTATIVES

Selection of representative prototypes(or *Lexeme* representative) from the training set is presented in this section. As the variabilty of the writing styles that must be captured by a system increases,it becomes more and more difficult to discriminate between different writing styles for each class into different subclasses,known as *lexemes[]*. Using the distances calculated by the string maching method of *Section 2* .A matrix is constructed for each class of character. Each matrix is measures the intra class distances for a particular character class characters in particular class are clustered in an attempt to find a small number of prototypes .Clustering is done using a *Squared Error Clustering.*

3.2.1 Squared Error Clustreing

In order to make use of this technique,we require that each Character be represented by a common feature vector. However in this technique data does not have such a representaion since it is in the form of a string. An approximation to a Euclidean feature space can be obtained by representing each character as the vector of distances to every character in the feature space. Clustering is then done in which the feature vector for a character is its M distances to each one of the M characters belonging to the same class.

A S*quared-Error Clustering* algorithm[31], attempts to produce the best clustering for each value of K, where K is the number of clusters into which the data is to be partitioned. Figure 3.2 shows an example of the number of clusters plotted against the mean-squared error for the character class “m”. As expected, the mean-squared error decreases monotonically as a function of K. An inspection of Figure. 3.3 shows that identication of a “knee” in the curve is not easily accomplished. This example is typical for most characters, choice of K is very difficult by this method.

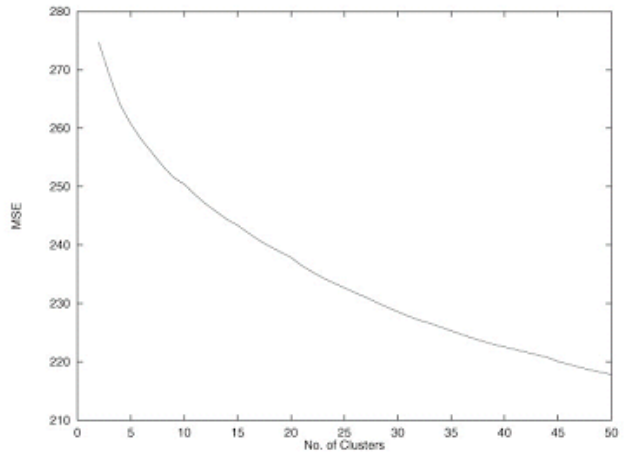


Figure 3.3: Mean squred error for different values of K,the number of clusters,for the character “m”.

Automatically determining the number of clusters that are present in a data set is a very difficult problem. We use the method proposed by [32,33] to select an “optimal” value of K. Now after the clustering the problem then ramains how to represent a cluster. Becuase the classification is highly dependent on the choice of representative character(or *lexeme representative*) for each cluster. A number of methods were attempted to select a single character which best represents its cluster. **Min Intra-Cluster Distance** was to choose the character for which the sum of distances to all other characters in the cluster was a minimum. Therefore this method choose a representative that is closest to the centre of a cluster. Another method **Max Inter-Class Distance** picked the character whose sum of distances from other classes is the maximum. Therefore this method involves the distances between the representative character and clusters from the other classes than that character’s class. Therefore,when reducing the size of the training set,we would like to find cluster representatives(or *lexeme representatives*) which mostly account for the smallest of the intra class distances and the largest of the inter class distances .The following figure 3.4 shows two *Lexemes representatives* of the digit 2 .

Figure 3.4:Two diiferent lexemes of the digit 2 .The digit in the centre of each cluster shown here,is the training sample(lexeme Representative) that lies closest to the cluster center.

3.3 ANOTHER METHOD OF FINDING LEXEME REPRESENTATIVE

Clustering algorithm ***Tree Clust*** is hierarchical.In this method clusters are represented by prototypes which are the samples having the minimum distances to the other samples in the same cluster.*Tree Clust* starts from a situation in which all the samples are prototypes,i.e form their own cluster.

As the clustering algorithm proceed,the number of clusters is reduced by merging of clusters. In this method those two clusters whose prototypes are similar to each other are merged into one. The following figure 3.5 shows the hierarchical clustering found by the *Tree Clust* algorithm.Each row corresponds to one clustering solution and the lines show which clusters are merged to get the next solution.The shown character samples are *lexeme representatives*.

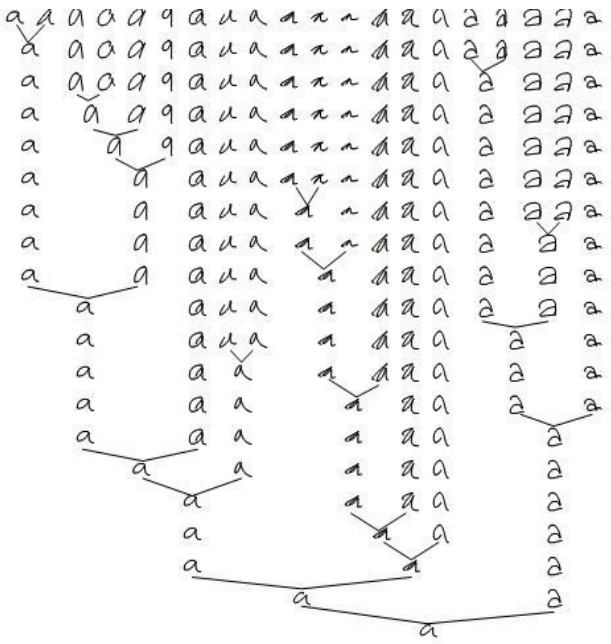


Figure 3.5:A hirarachical clustering found by *Tree Clust* algorithm

3.4 CLASSIFICATION

We discuss decision tree classifier for the classification purpose. This method can be used in combination with the data reduction technique from the previous section.

3.4.1 Classification with Decision tree classifier

Decision trees take a “divide and conquer” approach to solve a complex classifcation problem and attempt to identify those features which provide the most discriminating information .In

order to represent a character as a fixed length vector of features, the distance from a given character to each of the M *representative lexemes* (which will henceforth be referred to as the reference set characters) was measured giving the following feature Vector:

$[D_{c,1},D_{c,2},\ldots\ldots D_{c,M}]$ where Dc,i is the distance between a character, c, and the ith reference character. These features shall be referred to as the similarity features and they can be used to partition the training data at each node of a decision tree. For example, at node, n,when feature Dc,i(n) is used to partition the data based on threshold Dc,i(n), the following *Rule* will be applicable:

If Dc,i(n) $\lessgtr T_{Dc,i(n)}$, traverse the left branch of the node n otherwise, traverse the right branch of the node n.

In such a case, characters which traverse the left branch can be said to be similar to the i(n)th reference character, and therefore may belong to the cluster corresponding to this reference character, while the characters that traverse the right branch can only be said to be dissimilar to the i(n)th reference character, creating an imbalance in the division of data between the two branches. This imbalance motivates a second type of feature which can be used to partition the prototype data(*lexeme representatives*) into two groups at each node. By measuring the distance of a character, c, to two reference characters rather than only one at each node, we can ask the following question: `Is character c more similar to reference i, or to reference j?". To pose such a question as a vector of continuously valued features, we define

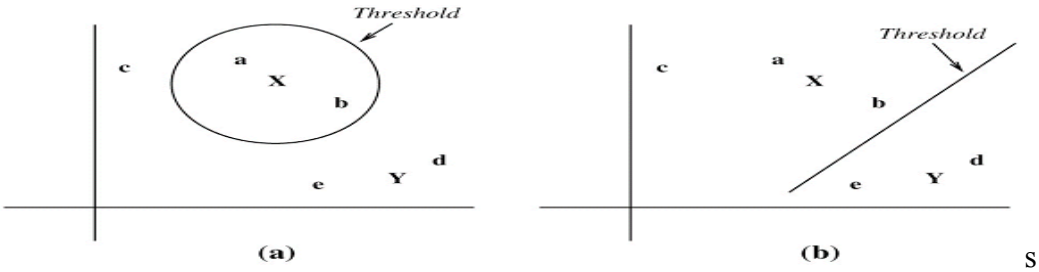
$[D_{c,i,j}]$, i=1,2……..M, j=1,2……...M,

Where $D_{c,i,j} = D_{c,i} - D_{c,j}$

These features shall be referred to as the difference features. When constructing a decision tree, at any node,n, we choose a pair of reference characters, i(n) and j(n),with corresponding feature

$D_{c,i(n),j(n)}$ and a threshold $T_{Dc,i(n),j(n)}$ for partitioning the data based on the following Rule If,$D_{c,i(n).j(n)} \lessgtr T_{Dc,i(n),j(n)}$ traverse the left branch of the node n.

otherwise, traverse the right branch of node n.Fig.3.6 illustrates these two feature representations.



S

Figure 3.6: Similarity and difference features.(a) similarity features: while trainning patterns a and b are grouped based on their similarity,to *lexeme representative* X,examples c and d are very different from each other but grouped together nonetheless:(b) Difference features: training patterns on each side of the decision threshold(difference between *lexeme representative* X and Y) are more similar to each other than to the patterns on the other side.

CHAPTER 4

FLEXIBLE STRUCTURAL MATCHING FOR ONLINE CHARACTER RECOGNITION

FLEXIBLE STRUCTURAL MATCHING FOR ONLINE HANDWRITTEN CHARATER RECOGNITION

4.0 INTRODUCTION

After we have written a character on the digitizer, what we get is only a sequence of points. In order to recognize the character,we must first extract the structural primitives from the point sequence to form a preliminary structure. Occasionally, some kinds of reconstruction may be required if certain conditions are met. Then the finalized structure, is compare with the models in the model set and try to find a match .Here flexible structural matching has been applied in order to increase the chance of finding a match. Figure. 4.1 summarizes the major stages of the whole Technique.

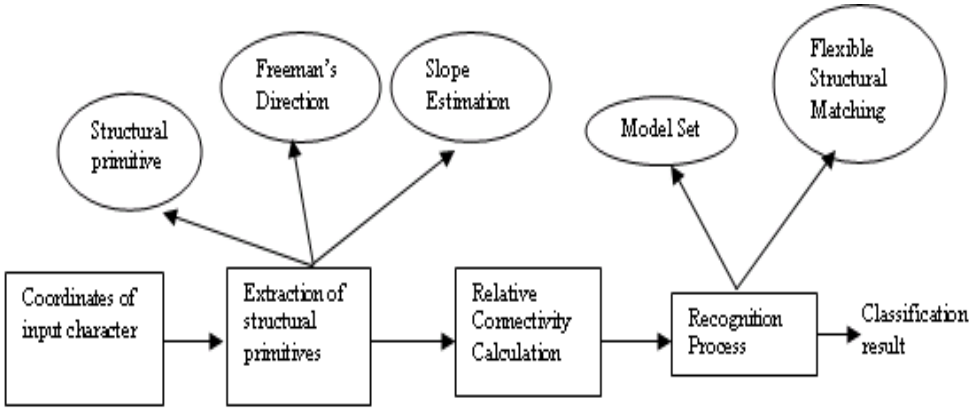


Figure 4.1:Major steps of the technique

4.1 EXTRACTION OF STRUCTURAL PRIMITIVES

4.1.1 Structural primitives

Characters are composed of line segments and curves. Every line segment or curve can be extended along a certain direction .A curve that joins itself at some point forms a loop. Hence,in this technique primitives of different types like line segments and curves with some directional information are used. Note that a single stroke may consist of several primitives. Basically, there are five types of primitives:

- line

- up (curve going counter-clockwise),
- down (curve going clockwise),
- loop (curve joining itself at some point), and
- dot (a very short segment which may sometimes be just noise; we, however, cannot simply ignore it since it may be part of a character, like in ‘i' and ‘j').

4.1.2 Freeman’s Direction Extraction

To represent the directional information the Freeman’s chain code[22] is used which consists of eight values, i.e.,0-7(corresponding to 0°,45°,…….315° respectively),to indicate how the current point is connected to the next one. To extract the Freeman’s code from point sequence, same method as explained in *Chapter 2* under the heading of *Direction Primitives* has been applied here. Finally, we will have a sequence of codes that represents the directions among the drawn pixels.Note that no directional information is associated with dots.Also,for simplicity,directions are not associated with loops at this satge. The direction of a line or a curve depends on the starting and ending points. Figure 4.2 shows some examples.

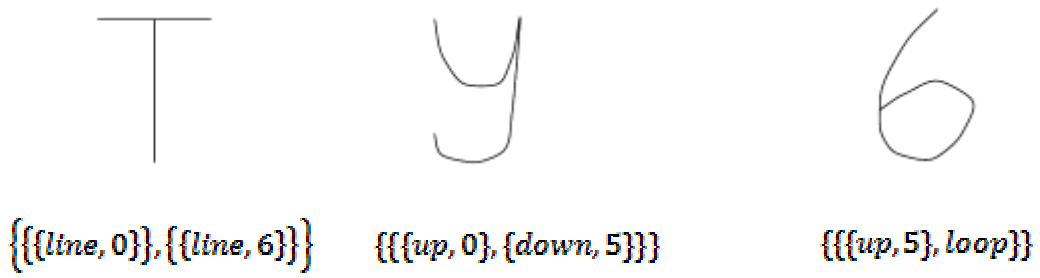


Figure 4.2: Examples of the representation for some characters

4.1.3 Slope estimation

The user draws the digit on a special window using a digitizer (or mouse). Then the coordinates (x,y) of the pixels representing the drawn digit are saved on a file. These coordinate values are used for calculating the slope values [34]. The signs of the slope values (+ and -), the zero values,and the infinity values are saved and used in the feature extraction step(Extraction of Structural primitives).Figure 4.3 shows an example,the representation of the digit 2.

The next step is to group slopes and corresponding coordinates(say X and Y).Also the sign of ΔY i.e (Y2-Y1) is used to determine the direction of writing or drawing(upward or downward) .if

$\Delta Y > 0$ and the reference point is located on the top left corner,then the directing of writing is downward,otherwise the directing of writing is upward.

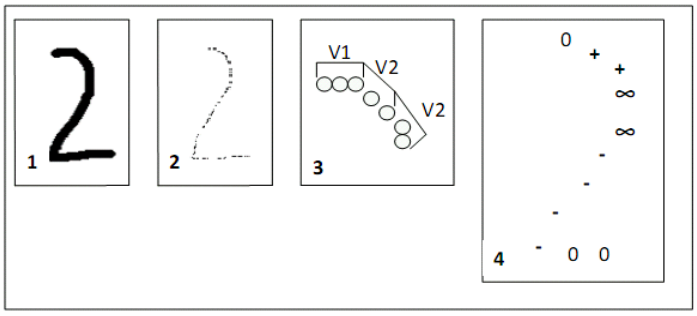


Figure 4.3:Representation of digit 2

4.1.4 Getting primitives

Method to detect lines curves and lopes is as expalined below:

Detection of line and curve

In a segment, a line or curve can be detected by counting the increments and decrements in successive slopes .If both the counts are less than a threshold then it is considered to be a line otherwise it is a curve .If increment count is more than the decrement count then it is a curve in counter clockwise direction otherwise direction is clockwise.

Detection of loop

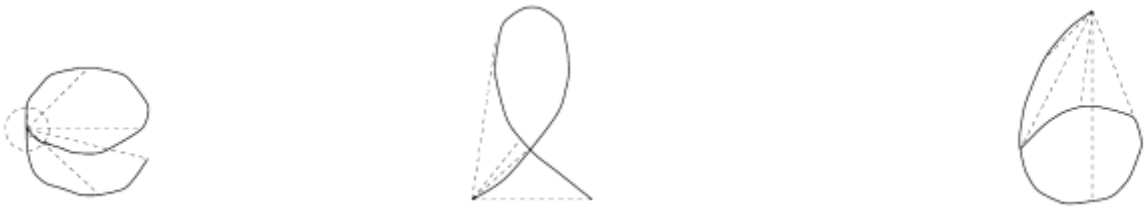
In general, a loop can be found in three different locations of a stroke:

1. at the beginning,

2. in the middle, and

3. at the end.

Figure. 4.4 gives some examples.



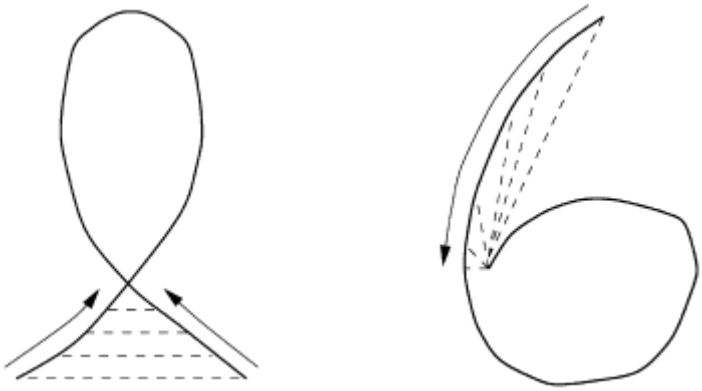
(a) At the beginning (b) In the middle (c) At the end

Figure 4.4: Different locations in which a loop can occur within a stroke

Loop can be detected with the help of original coordinates of input character not by the slopes as was in the case of line and curve .To detect the loop at the start (as shown on figure 4.4(a)), the distance of starting point from all other points in the order they are obtained has been found.If the distance first increases and then starts decreasing and goes below a certain threshold then this indicates a loop at the start of character.

When the loop is in the middle of a stroke, we can check the distance between points, starting from the two ends of the stroke .When the distance decreases, we will continue to move inward until the distance is less than a certain threshold .When this happens, a loop is said to be found.

For the case of having a loop at the end of a stroke, we apply a similar method. However, we will fix the ending point this time.Therefore to detect the loop at the end of character, distance of the last coordinate from all other coordinates is determined .If the distance first decreases up to a certain threshold and then starts increasing then it indicates the presence of a loop at the end of character. Figure. 4.5 illustrates these two cases.



(a) Extracting a loop from the middle of a stroke (b) Extracting a loop from the end of a stroke

Figure 4.5: Extracting loops from the middle and the end of a stroke

In general, loops can be detected by using one of the above methods, or a combination of some of them .For example, ‘g' contains loops both at the beginning and in the middle (sometimes, at the end) of the stroke .We, therefore, require a combination of two methods for detecting them .Figure.4.6 shows some examples of combining several methods to extract loops from a stroke.

Figure 4.6: Combination of several methods to extract loops from a stroke

4.1.5 Structure reconstruction

After obtaining the preliminary structure,we sometimes may need to either combine some lines and curves together under certain conditions, or further extract some sub-structures, such as loops, if they are detected in the original structure.

Combining lines and curves

Occasionally,a smooth stroke may be broken into parts due to either poor writing or low-quality hardware .In order to remedy this, we have to check each pair of consecutive primitives in a stroke .If some conditions are met, we will combine the two primitives together to form a new one. *Table 4.1* shows some conditions under which consecutive primitives are combined.

Primitive 1: $\{T_1, D_1\}$	Primitive 2: $\{T_2, D_2\}$	Condition	New primitive: $\{NewT, NewD\}$
$\{line, D_1\}$	$\{line, D_2\}$	$D_1 = D_2$	$\{line, D_1\}$
$\{up, D_1\}$ or $\{down, D_1\}$	$\{up, D_2\}$ or $\{down, D_2\}$	$T_1 = T_2$ = JointType	$\{T_1, NewD\}$

Table 4.1:Some conditions for combining consecutive primitives

Note that T_1 and T_2 denote the types of primitives and D1 and D2 represent their directions, respectively .The new primitive has New T as its type and New D as its direction (from the starting point of the first primitive to the ending point of the second primitive) .As shown in *Table 4.1*, in order to combine two lines together, their directions must be the same . Figure 4.7. shows an example

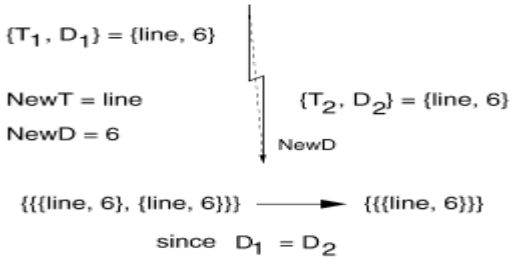


Figure 4.7:Combining two lines into one

Combining curves is slightly more complicated .First of all, we have to determine the joining type which describes how the curves are connected .For example, if the direction of the last line segment of the first primitive is 2 and that of the first line segment of the second primitive is 4, it implies that the change is from 2 to 4. As shown in Figure.4.8, there are two alternatives to change from 2 to 4 .Here we always choose the shorter path, and therefore, JointType=find_joint_type (2, 4) =up.

Figure 4.8: Alternatives in changing from 2 to 4

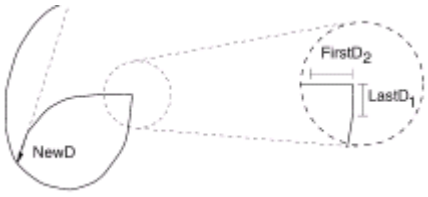
By comparing the value of JointType with the types of both curves, we are then able to decide whether the two curves can be combined .Figure 4.9. illustrates when two curves can and cannot be combined together.

After this stage, sets of characters which have only slight variations in point locations can be grouped under their corresponding structures.

$\{T_1,D_1\} = \{up,7\}$

New $T_1=T_1$

New D=6



$\{T_2,D_2\} = \{up,5\}$

joint type

=find_joint_type($LastD_1$,$FirstD_2$)

=find_joint_type(2,4)

=up

{{{up,7},{up,5}}} ⟶ {{{up,6}}}

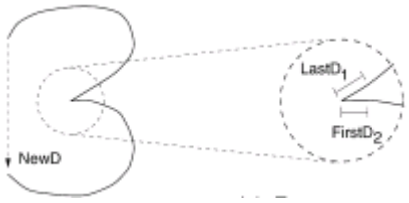
Since T_1=T_2 = joint type

4.9 (a) Two curves can be combined together

$\{T_1, D_1\}$ = {down, 7}

New T=T_1

New D=6



$\{T_2,D_2\}$ = {down,5} joint type

=find_joint_type($LastD_1$,$FirstD_2$)

=find_joint_type(5,0)

=up

{{{down,7},{down,5}}} ⟵ {{{down,6}}}

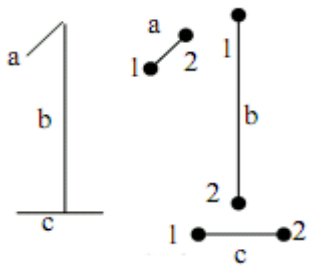
since T_1=down, joint type = up and T_1 <> joint type

(b) Two curves can not be combined together

Figure 4.9:Examples illustrating when two curves can and can not be combined together

4.2 RELATIVE CONNECTIVITY CALCULATION

For calculating connectivity between ith and (i+1)th primitive, we calculate the distances of first coordinate of ith primitive with the first and last coordinates of (i+1)th primitive .If a distance is less than a certain threshold it means that corresponding points of the strokes are connected and corresponding bit in 4-bit vector is assigned value '1' otherwise '0'.Same procedure is repeated for the last point of the ith primitive .This relative connectivity 4-bit vector (Figure 4.10) is then placed between the ith and (i+1)th primitives in the structure string for the character. Figure 4.11 shows that how ambiguities of the character having similar sequence of primitives removed using relative connectivity information.



Connectivity Codes

(a,b) = (0010)

(b,c) = (0000)

(c,a)=(0000)

Figure 4.10: Relative Connectivity code calculation

{{line 0},{line 5},{line 0}} {{line 0},{line 5},{line 0}}

(a)

{{line 0}(0010){line 5} {{line 0}(0010){line 5}

(0000){line 0}(0000)} (00100{line 0}(0000)}

(b)

Figure 4.11: (a) without connectivity code (b) with connectivity code(no ambiguity)

4.3 RECOGNITION PROCESS

4.3.1 model set

Before we can perform recognition, we need to define models for the corresponding character classes .Note that all models should be distinct .In other words, there exist no two models with the same number of strokes and the same sequence of primitives in each stroke .Also, different character classes may have different numbers of models depending on the complexity of their structures .Figure. 4.12 shows some examples. As shown in Figure. 4.12, some models look very similar but have different numbers of strokes.

(a) ‘o’ has only one model (b) ‘p’ has 5 models

(c) ‘5’ has 9 models

Figure 4.12: Examples of the models for some character classes

4.3.2 FLEXIBLE STRUCTURAL MATCHING

After extracting the structure of a character (possibly with some reconstruction steps involved), we can then match it against a set of models .However, due to different writing styles and habits, variations within the same character class are not uncommon .In order to increase the recognition rate, those characters that do not have an exact match will be slightly varied in shape and direction in an attempt to find approximate matches.

The following is the matching algorithm:

Algorithm. Elastical structural matching

1. Load the set of models in Z.
2. Extract the structure of the test character C.
3. Initialize the deformation level L to be 1.
4. Let S be the candidate set and S=deform(L, C).
5. Let M be the match set and M=match(Z, S).
6. If M is not empty, return M. Otherwise, L=L+1.
7. If L is less than or equal to the maximum deformationlevel, go to step (4). Otherwise, exit and report failureof finding an exact match.

After loading the set of models and extracting the structure of the test character (as described in Section 4.1), we start to find a match or matches.First of all, we will simply compare the structure for the unknown character against the set of models to see whether at least one match

can be found .If no match is resulted, we will deform the test structure in certain ways so as to increase the chance for finding matches.When all the deformation methods are exhausted and no matches are found, we then report failure.

Basically, there are four levels of structural deformation .

1. No deformation: The test pattern has to be exactly the same as one of the models.

2. Primitive type deformations: When there is no exact match, we will vary the primitive type in an attempt to find an approximate match .In so doing, line may become either one of its two neighboring types, i.e., up and down .

3. Directional deformations: Similarly, we may also vary the direction .To do so, we find a neighboring code of the current one .For example, {line, 5} may become{line, 4} or {line, 6}.

4. Simultaneous type and directional deformations:

When no exact pattern can be found during the previous relaxation steps, we may consider finding the nearest match by deforming both the primitive type and direction simultaneously .As a result, a much larger number of patterns will be covered.

By using fexible structural matching, some previously unmatched characters are able to find a match .This increases the recognition rate and at the same time decreases the rejection rate.

CONCLUSION

There are various methods of *Online Handwriting Recognition*, but in the present study, three techniques are investigated. These are as follows:

*Online Character Recognition using Dominant Points in strokes (say **Tech 1**)*

*Prototype (Template) based Online Character Recognition (say **Tech 2**)*

*Flexible Structural matching for Online Character Recognition (say **Tech 3**)*

Tech 1 **(Chapter 2)** is divided into two sections (say ***Tech 1.1*** and ***Tech 1.2***)

Tech 1.1 gives feature extraction, in which Dominant points and direction primitives have been extracted as features. ***Tech 1*** is totally based upon totally Dominant points. As explained in *Chapter 2* there are two types of Dominant points (***Type 1*** and ***Type2) Type 1*** includes Dominant points as (a) start point and final point of a stroke (or pen-down and pen-up points); (b) local extreme; (c) middle point that connected the previous two points in (a) and (b) .***Type 2*** includes points corresponding to gradually or sharply turning points in the online handwriting curve.

Tech 1.2 gives character classification. It is divided into two steps. Selecting candidate during the pre-classification in step 1, in step 2 fine classification is performed on the candidate classes that survive the pre-classification to finally classify an input character.

The **Advantages** and **Disadvantages** of ***Tech 1*** are explained as follows:

Advantage

A remarkable aspect of this approach is that it is easily extensible to different character sets and different writing styles. For example the system can also recognize symbols such as '+','-','$, £, if the corresponding templates are added into the reference set. Also the system can handle large character sets (such as Chinese Characters), due to its pre-classification step.

Disadvantages

a) Rather there are two types of dominant points as discussed in chapter 2, ***Tech 1*** generally use ***Type 1*****(*****Tech 1.1*****)** dominant points for the recognition process, because ***Type1*** dominant points are more suitable for the further pre-classification and fine classification steps to finally classify the input character, but the method used to detect these dominant points makes the system computationally complex. Moreover method of ***Tech 1*** is not robust to disturbance. Also Pre-classification and fine classification steps classify the input character by using band limited time warping and dynamic programming which further makes the recognition process very complex and results in the large amount of recognition time.

b) The recognition system proposed by ***Tech 1*** highly depends upon the stroke order, because in general a different stroke order requires an additional reference pattern which also makes the system computationally complex. Due to this recognition time increases.

Tech 2**(chapter 3)** is divided into three sections (say ***Tech 2.1, Tech 2.2, Tech 2.3*****)**

Tech 2.1 explains the *String Matching* method, which works on the sequence of sample points directly by searching for an alignment of data points. This string matching technique is further used for Lexeme representation and character classification.

Tech 2.2 gives the selection of representative prototypes (or *lexeme representatives*) from the training data. Clustering has been done in an attempt to find a small number of prototypes. Further by using *Min Intra Class Distance* and *Max inter Class Distance* Lexeme representatives have been chosen.

Tech 2.3 section of ***Tech 2*** gives the classification result by using *decision trees* .In this part the distance from a given character to each of the M representative Lexemes is used as a feature vectors (Similarity features).

Tech 2 also have some **Advantages** and **Disadvantages.**

Advantages

a) ***Tech 2*** is well suited for adaptive systems when some new training samples are obtained, the prototypes can be adjusted to represent them better. The initial prototype set is formed by clustering character samples collected from a large number of subjects. Thus, the

recognition system can handle various writing styles .There is no need to retain or adapt the whole recognition system but just the prototypes of same classes can be adjusted as the new samples. Incremental learning can be achieved just by including the new samples into the prototype set as soon as their class labels have been determined.

b) Recognition accuracy of ***Tech 2*** is good, due to the *clustering* and *lexeme representatives.* This is because the lexeme representatives contain relevant information for classification.

c) Common practice of ***Tech 2*** is to train a recognizer with data collected from several writers.Thus introducing several writing styles to the system expecting generlizational ability. Then if the writing style of the new user is similar to some style in the training set, the performance of the system for that writer can be adequate.

Disadvantage

Though the whole training database is guaranteed to include as many as possible writing style variations, it can make the recognition process computationally too heavy which further increases recognition time as it depends linearly on the size of the prototype set. Further reduction of training data set by using clustering and *Min Intra Class Distance* and *Inter Class Distance* also takes much time.

Tech 3**(Chapter 4)** also have three sections (say ***Tech 3.1, Tech 3.2, Tech 3.3***).

Tech 3.1 presents structural primitives (lines, curves, loops and dots). Lines and curves are detected by counting the increments and decrements in successive slopes, also the detection of loops have been explained in the chapter 3.After the detection of structural primitives structure reconstruction have been done by combining lines and curves if possible.

Tech 3.2 checks the relative connectivity of the primitives**.**

Tech 3.3 defines modal set and explains *Elastic Structural Matching* for the final recognition process.

Advantages and **Disadvantages** of tech 3 are explained below:

Advantages

a) ***Tech 3*** is model based so *training is not necessary*. New models may be added any time, therefore there is no need to spend large amount of time in training,

b) ***Tech 3*** reliably removes the *ambiguity* between the characters having same sequence of primitives and gives fairly good recognition speed under low memory requirements.

c) ***Tech 3*** is size and rotation invariant.

d) By using ***Tech 3***, some previously unmatched characters are able to find a match .This increases the recognition rate and at the same time decreases the rejection rate.

Disadvantages

a) The main problem with ***Tech 3*** is that model creation is not automatic. In other words, we still have to manually design the set of models in advance.

b) ***Tech 3*** is well suited for recognition tasks in which the structures of the patterns are paramount importance. The measured feature value or knowledge about their existence or absence do not always provide enough information for the classification of a character and some additional information about the relations between the features, or on the structure of the character is needed. Structural methods are especially well suited for Asian languages in which characters are more complex than Latin alpha numerical characters.

c) ***Tech 3*** requires human experts for making class specific rules as they are rather difficult to learn automatically.

d) To represent the directional information the Freeman chain codes (Tech 3.1) have been used, the ability of chain code to capture the 2 dimensional spatial structures of handwriting are rather limited and it heavily depends on complicated features the codes corresponds to.

Finally the overall study can be concluded in the following way:

- ***Computational complexity and recognition time*****:** As we talk about computational complexity and recognition time ***Tech 1*** and ***Tech 2*** are not very robust. As explained

above that Tech 1 is computationally very complex and takes much time for the recognition process due to huge amount of calculation work and also due to lack of response to pen movement in real time**.** On the other hand ***Tech 2*** is also not very good in this factor as explained above in ***Tech 2*** (disadvantage).

As discussed above ***Tech 3*** is modal based so training is not required, new modals can be added any time, therefore the size of the model set is not very big, also the extraction of primitives and recognition process does not take much time, therefore ***Tech 3*** is not computationally much complex as ***Tech 1*** and ***Tech 2***, therefore there is no need to spend large amount of time in training. So recognition time is very less in this case.

So in case of computational complexity and recognition time ***Tech 3*** *is best.*

- ***Learning new handwriting styles*****:** Now as the number of writing styles increases, so does the variability of the data distribution. For learning different writing styles ***Tech 1*** and ***Tech 3*** are not much suitable**.** ***Tech 3*** rather uses a elastic matching technique for classification in which if the character do not have an exact match can be slightly varied in shape and direction by using four levels of deformations ,is not very robust in learning new handwriting style, this is due to the lack of automatic model creation, however just the lowest level components of the structural handwriting models can be adapted to better represent new handwriting sampling and ***Tech 1*** is only extensible to different writing styles if the corresponding prototypes are added into the reference set. **B**ut on the other hand ***Tech 2*** is well suited for learning new writing styles as explained in advantages of ***Tech 2.***

  So ***Tech 2*** *is dominant according to this factor.*

- Nowadays the computation machines are very fast, if one can compromise the speed a bit, then **Tech 1** is the more suitable technique as this gives more precise results and dominant points are easy to detect. Therefore we can also go for the ***Tech1*** so if we think about the easiness of the user.

BIBLIOGRAPHY

1. C.C.Tappert, C.Y.Suen, and T.Wakahara (1990). "The state of art in online handwriting recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence 12(8), 787-808.*

2. http://lib.tkk.fi/Diss/2002/isbn9512262495/isbn9512262495.pdf

3. E.Bellegarda J. R. Bellegarda, D. Nahamoo, and K. S. Nathan (1994). "A fast statistical mixture algorithm for online handwriting recognition".*IEEE Transactions on Online Pattern Analysis and Machine Intelligence 16(12), 1227-1233.*

4. J.R.Ward and T. Kuklinski (1988). "A model for variability effects in handwriting with implications for design of handwriting character recognition systems". *IEEE Transactions on systems, Man, and Cybernetics 18(3), 438-451.*

5. B.Q.Huang, Y.B.Zhang, Kechdi (2007). "M-T Intelligent Systems Design and Applications", *ISDA, Seventh International Conference on Volume, Issue, 20-24 , pp.193-800.*

6. C.C. Tappett, C. Y. Suen, and T. Wakahara (1988), "On-line handwriting recognition-A survey," *in Proc. 9th Inc. Conf. Pattern Recognition, pp. 1123-1132.*

7. Li,X. and D.Y. Yeung (1997). "Online handwritten alphanumeric character recognition using dominant points in strokes". *Pattern Recognition* 30(1), 31-44.

8. http://fportfolio.petra.ac.id/user_files/99-015/103.pdf

9. Su Yang; Guozhong Dai (2002). "Frontiers in Handwriting Recognition". *Proceedings Eighth International Workshop pp.351 – 356.*

10. C.C.Tappert (1984). "Adaptative online handwriting recognition". *In International Conference on Pattern Recognition ,pp. 1004-1007.IEEE.*

11. A.K. Jain, D. Zongker (1997). "Representation and recognition of handwritten digits using deformable templates", *IEEE Transactions. Pattern Anal. Mach. Intell. 1 9 (12) pp. 1386 -1391.*

12. S. Connell, A.K. Jain (1988). "Learning prototypes for on-line handwritten digits", *Proceedings of 14th International Conference on Pattern Recognition, Brisbane, Australia, pp. 182-184.*

13. http://www.zju.edu.cn/jzus/2006/A0610/A061002.pdf

14. http://www.inf.ed.ac.uk/teaching/courses/inf2b/learnnotes09/inf2b09-learn03-lec-2x3.pdf

15. A.K. Jain, R.C. Dubes (1988). "Algorithms for Clustering Data", *Prentice-Hall, Inc.*

16. A.K. Jain, D. Zongker (1997), "Representation and recognition of handwritten digits using deformable templates", *IEEE Trans. Pattern Anal. Mach. Intell. 19 (12) pp. 1386-1391.*

17. A.K. Jain, L. Hong, S. Pankanti, R. Bolle (1997), "An identity-authentication system using fingerprints", *Proc. IEEE 85 (9) 1365-1388.*

18. T.Wakhara and K. Odaka (1997). "Online cursive kanji character recognition using stroke based affine transformations". *IEEE Transactions on Pattern Analysis na machine Intelligence 19(12), 1381-1385.*

19. C.C.Tappert, C.Y.Suen, and T. Wakhara (1990). "The state of art in online handwriting recognition".*IEEE Transactions on Pattern Analysis and Mavhine Intelligence 12(8), 787-808.*

20. R.Martens and L. Claesen (1997). "Dynamic programming optimization for online signature verification". *In proceeding of International Conference on Document Analysis and recognition,pp. 653-656.IEEE.*

21. Y.Sato and K.Kogure (1982). "Online signature verification based on shape,mption, and writing pressure". *In Proceedings of International Conference on Pattern Recognition,Volume 2,pp. 823-826. IEEE.*

22. H.Freeman and L. S. Davis (1977). "A corner finding algorithm for chain coded curves". *IEEE Transactions on Computers, 287-303.*

23. Hung Yuen (1996). "Acoustics, Speech, and Signal Processing", *IEEE International Conference on Volume 6, pp. 3426-3429 vol. 6 Digital Object*

24. http://www.icgst.com/gvip/Volume5/Issue7/P1150527004.pdf

25. http://www.informatica.si/PDF/32
26. Kam-Fai Chan and Dit-YanYeung (2000). "An efficient syntactic approach to structural analysis of on-line handwritten mathematical expressions". *Pattern Recognition, Vol. 33, pp. 375 - 384.*

27. K.F.Chan and D.-Y. Yeung (1998). "A simple yet robust structural approach for recognizing online handwritten alphanumerical characters". *In Proceedings of the sixth international workshop on frontiers in handwriting recognition, pp. 229-238.*

28. Kam-Fai Chan and Dit-Yan Yeung (1999). "Recognizing on-line handwritten alphanumeric characters through flexible structural matching". *Pattern Recognition, Vol 32, pp. 1099 – 1114.*

29. H.G.Campbell & R.E. Spencer (1977). "Finite Mathematics and calculus". *New York: Macmillan Publishing Co., Inc.*

30. http://www.ralphniels.nl/pubs/niels-dtwintuitive.pdf

31. http://130.203.133.121:8080/showciting;jsessionid=87830E4C0744DDBD440D001432CD45E9?cid=946480

32. F. Alimoglu, E.Alpaydin (1997). "Document Analysis and Recognition", *Proceedings of the Fourth International Conference on Volume 2, pp. 637 – 640.*

33. S. Madhvanath, G. Kim, and V. Govindaraju (1999). "Chain code Contour Processing for Handwritten Word Recognition". *IEEE Trans. On Pattern Analysis and Machine Intelligence, 21 (9), pp. 928 - 932.*