

M. Tech. (Computer Science) Dissertation Series

Online Handwritten Bangla Character Recognition

**a dissertation submitted in partial fulfillment of the
requirements for the M. Tech. (Computer Science)
degree of the Indian Statistical Institute**

By

Tamaltaru Pal

under the supervision of

Utpal Garain



INDIAN STATISTICAL INSTITUTE
203, Barrackpore Trunk Road
Kolkata – 700 108

Abstract

Handwritten text is a common, natural form of communication for humans. Therefore, as a useful means of input to machines, this mode can be utilized. This thesis deals with the online handwriting recognition system for Bangla script. A template-based recognition system is demonstrated here. An efficient classifier has been designed and a similarity measure, which is quite robust against wide varieties of writing styles, has been proposed. The method has been initially applied for recognition of characters in Bangla script. Our classifier achieves an overall 96.34% accuracy on a dataset of considerable size alphanumeric characters (60 classes — 10 numerals, 29 full letters, 12 parts of a letter and 9 vowel marker).

Key words: *Online handwriting, Pen-based input device, Writing Direction, Trajectory Length, Template based classifier, Similarity measure.*

Acknowledgement

The author would like to sincerely thank the following people for providing assistance, support, encouragement, and inspiration during the work being done and writing of this dissertation. First, I'd like to thank my advisor Mr. Utpal Garain. He has provided guidance, encouragement, opportunities, and knowledge at a level that few advisors are capable of. Second I'd like to thank Prof. B. B. Chaudhuri, Head, CVPR Unit, ISI and the other faculty members of the CVPR Unit, ISI for their support and assistance. They have provided laboratories, other accessories, data and advice that have been extremely helpful in the completion of this dissertation.

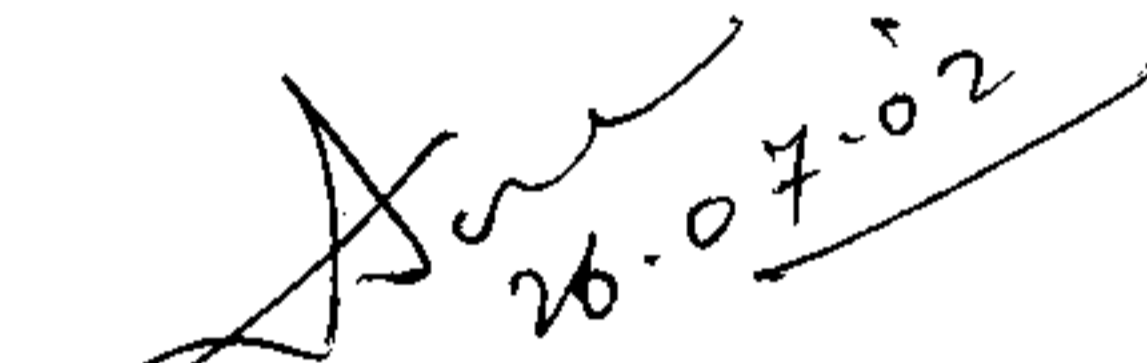
No student can survive without the help of his or her fellow students to discuss ideas, share opinions, and to make time spent in the laboratory an all around enjoyable experience. In particular, I'd like to thank Dr. Joydeep Mitra, Diptendu Bikash Dhar, Partha Mukherjee, Mitil Roy, Anirban Roy and other classmates of our M. Tech (Computer Science) batch.

I'd like to give special thanks to my wife. She has always been there to support me, and I am very grateful. Finally, I'd like to thank my friends and other supporting staffs of the CVPR Unit, ISI, Kolkata.

Certificate

This is to certify that the thesis entitled "**Online Handwritten Bangla Character Recognition**" submitted by **Mr. Tamaltaru Pal**, towards partial fulfillment of the requirement for *M. Tech in Computer Science* degree of the *Indian Statistical Institute, Kolkata*, is an acceptable work for the award of the degree.


Supervisor


26-07-02
External Examiner

Dated: 26th July, 2002.

Contents

1. Introduction		
1.1. Introduction	1
1.2. Background	2
2. Methodology		
2.1. Data Acquisition	6
2.2. Preprocessing	7
2.3. Our Approach	8
2.3.1. Feature Extraction	11
2.3.2. Data Reduction	14
2.3.3. Character Recognition	15
3. Results	18
4. Conclusions	21
Appendix A	23
Appendix B	25
Appendix C	27
References	29

1. Introduction

1.1 Introduction

Interfaces like tablet-n-stylus are now-a-days becoming very popular and will play an important role in man-machine interface in the near future. Small hand-held computers like personal digital assistant are accepting pen-based inputs. There are several advantages to use a pen as an input device. A pen can be very easily portable. It can do the basic operations of a conventional mouse other than its own job. Over all a pen gives the comfort of natural writing and by means of this device we can reliably digitized one's natural handwriting. Moreover, text data for scripts having larger alphabet size like Bangla, is very cumbersome to entry through the keyboard. These are our motivation to develop an online handwritten Bangla script recognition system.

Bangla, the 4th most popular language in the world is derived from Sanskrit, an ancient Indo-Aryan language. About 200 million people in the eastern part of Indian subcontinent speak in this language and a considerable proportion of this Bangla speaking population use Bangla script as their writing media. Moreover, almost the same script is used for writing another east Indian language — Assamese.

A lot of work [1,2,3] has been reported for off-line recognition of some printed Indian scripts. Also S. D. Connell et al. [4] do some work for on-line

recognition of Devanagari script. But, to the best of our knowledge, no work for on-line recognition of Bangla script has been done.

The problem of handwriting recognition has now been explored for more than three decades. For handwritten input modality, the recognition accuracy must be sufficiently high so that by minimal human interventions an user can reliably and commercially use it. Recently the technology progressed to the point where consumer products based on handwriting recognition have become affordable and commercially available. Many types of problems with varying complexity exist in the domain of handwriting recognition, e.g. — how the handwriting data will be presented to the system, how and where the data can be unambiguously broken into pieces (individual characters or words), who will use the system, etc. Our problem is to recognize isolated Bangla script characters. However, online handwriting recognition becomes a challenging work in the area of pattern recognition problem due to a wide variation of human writing style.

1.2 Background

Handwriting recognition can be divided into two categories — on-line and off-line. This division is done on the basis of representing the data to the system. For the first kind of recognition system user's handwriting is digitized by a scanner at a later time and the data is presented to the system as an image while for the second kind, handwriting is digitized by a tablet and a stylus at the time of writing and strokes are captured as they are being formed, by sampling the pen's position at evenly spaced time intervals. A

pressure sensitive switch on the tip of the pen detects *pen-up* / *pen-down* status and discriminates stroke segmentations.

This section describes some of the issues and techniques involved in on-line handwriting recognition. A lot of survey papers exist [5,6] for the techniques of recognizing of handwritten data. Bellegarda et al. divides on-line handwriting recognition techniques into the following five categories — (i) Primitive Decomposition, (ii) Motor Models, (iii) Elastic Matching, (iv) Stochastic Models and (v) Neural Networks. The advantages and disadvantages are summarized and reported in [7].

Primitive decomposition identifies sub-strokes like loops, dots, crossovers, arcs, ascenders, and descenders etc., which are common building blocks for characters. This method generally requires a pre-segmentation of the strokes into sub-stroke pieces. Word recognition is performed by matching identified sub-stroke sequences to previously observed sequences for words using such methods as dictionary lookup or hidden Markov models.

Motor models are a technique commonly used in what is known as *Analysis by Synthesis* in which models of stroke segments are created along with rules for connecting them to form characters. Motor models represent these stroke segments as parameterized models of the motion of the pen tip, simulating the physical properties of human hand motion.

Elastic matching works on the sequence of sample points directly by searching for an alignment of data points between an input character, and some template character. 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 neighbour classifier. Jain and Zongker [8] demonstrate a *featureless* representation of off-line characters. They used deformable models to find the similarity between characters.

Stochastic models are often used in a similar fashion as elastic matching in that the data is represented in terms of a temporal sequence. The most common such method is to represent each class using a hidden Markov model. These models are often created using features extracted from the individual sample points, or from the points that are contained within a sliding window which slides along the sample point sequence thereby producing a sequence of features.

Time delay neural networks are often used to recognize characters or character segments as a sliding window passes over their temporally sampled sequence. Features extracted from the sample points in the sliding window are passed to the input layer of a feed-forward neural network. The activation level of each output node, one per class identity, approximates the likelihood that the sequence of points in the sliding window belongs to that class. This then produces a sequence of likelihood values, which can be used to find the best sequence of character identities using methods such as non-linear *dynamic time warping* algorithms [9] and hidden Markov models [10]. Some neural network classifiers have combined both stroke based features and OCR type features, in which a character is converted into a pixel array and features are taken as the pixel values in different regions of the image.

Some results from the recent literatures, which deal with the isolated character recognition problem, where segmentation is assumed to have been done correctly) has been summarized and reported in [11]. The techniques mentioned above mostly deal with handwriting in English. A few work exist for dealing with Japanese Kanji [12] and Korean Hangul [13], etc. On the other hand, oriental scripts in general and Indian script in particular are largely neglected.

In this thesis a novel technique for online character recognition of Bangla script is proposed. We have viewed the problem of developing an algorithm is as that of designing a technique for teaching a child how to write alphabets of a script. It, basically, addresses how to control pen movements in a proper way to write a character and this in turn, refers to the functionality of our human motor system. The trajectory information and time evaluation of pen co-ordinates play major role in capturing these functionalities.

2. Methodology

2.1 Data Acquisition

For data acquisition we have used a tablet-n-stylus type device from **FRONTECH**. The commercial name of the product is **SmartPen** version 1.4. It has a tablet with writing area 11.88 inch x 8.88 inch, on which one can write freely by a stylus. The dimension of the stylus is that of a common writing pen (6 inch long & 0.5 inch thick), which provides the writer the comfort of writing on a piece of paper.

By the above hardware, we acquire the trace of a writer's pen 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 position on the digitizing tablet and the status of the pen, which can be either *pen-down* or *pen-up*. Mandler et. al. [14] show that using the above information about the pen dynamics, better recognition accuracies can be achieved than merely applying OCR techniques.

However, we can formalize the acquired data as follows. Let the hardware transmits a sequence of sample points $\{P_i\}_{i=1}^N$ equally spaced in time interval. The value of N can varies from 10 to 175 for our Bangla script characters. Each sample point $P \in A$, where $A = \mathfrak{R} \times \mathfrak{R} \times \{0,1\}$ has three components x, y and s , where $(x, y) \in \mathfrak{R}^2$ represents the pen coordinates on

the digitizing tablet and $s \in \{0,1\}$ represents the pen status – 0 means *pen-down*, 1 means *pen-up*.

2.2 Preprocessing

Strokes recorded by a digitizing tablet usually contain a small amount of noise. These noises are introduced by the digitizing device itself. The sampled curves become jagged when written with this device. Beside this, since the device records samples in equal time interval, it records a particular pen position more than once if curves are slowly drawn. Hence we have to remove these repetitions of pen positions. We use a smoothing technique to reduce these kinds of noises. To describe the smoothing technique we utilize the same notation as described in the previous section.

Let $\{P_i\}_{i=1}^N$ be the sequence of points recorded by the device. Also let $c = P_i - P_j$, where $j = i + 1, i + 2, \dots$ and P_i & P_j are two pen positions. Here $c \in \mathfrak{R}^2$ has two components (dx, dy) . We discard the pen position P_j , if the following condition is satisfied.

$$(dx)^2 + (dy)^2 \leq m^2 \quad \dots\dots \quad 2.1$$

where $dx = (x_i - x_j)$ and $dy = (y_i - y_j)$. Otherwise we retain the position P_j and rename it as P_i and repeat the process till $j < N$. If we choose m as 0, we can remove all repeated samples. If we choose m as 1, we can remove repeated samples as well as some jaggedness from the curve. While writing a stroke, it may so happen that the digitizer may sample two positions either side by side or up and down as shown in Figure 2.1, which give rise to

spurious horizontal or vertical *writing directions*. The *writing direction* is explained later in proper place. In the above situations as shown in the figure, the distance between the two sample points is exactly 1. So, if we set m as 1 and discard one of the sample points, we may get rid of spurious horizontal or vertical directions. Hence, we shall be able to smooth the input strokes by some extent.

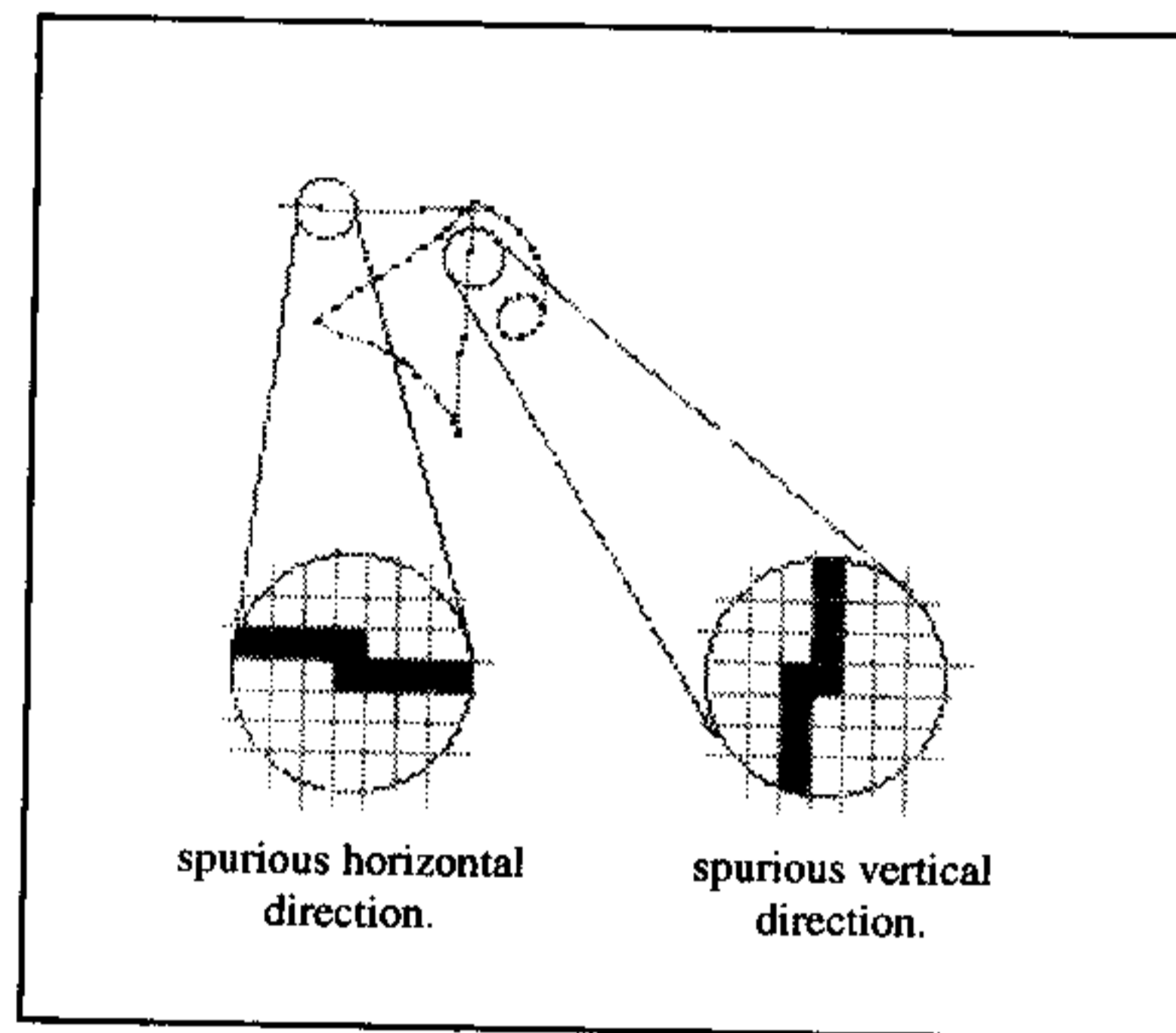


Figure 2.1. Horizontal and vertical jaggedness of a stroke.

2.3 Our Approach

Bangla scripts contain 51 basic and about 250 compound characters. Besides these, there are about 10 markers that may be attached to the left, right, top or bottom of basic or compound characters. Online recognition of these characters imposes several problems like stroke number, stroke connection and shape variations etc. Many characters are composed of multiple strokes. Figure 2.2 presents some handwritten Bangla script characters. In the figure, the first five rows show sample of five different writers, while the last row indicates corresponding printed character shapes. It

is also important to note that shape of the characters and stroke connections may vary.

However, we have minutely observed the handwritten Bangla character set and find 60 distinct stroke categories of which 10 are numerals, 29 single stroke complete characters, 12 parts of a character and 9 markers. See Appendix A for detail description. We restrict ourselves to recognize only the basic isolated Bangla script characters. Hence we discard the possibilities of 250 compound characters. Using these 60 strokes we can compose multi-stroke basic characters as well as some multi-stroke compound characters. Vide Table A.2 at Appendix A.

৫	অ	এ	ক	ট	ম
৫	অ	এ	ক	ট	ম
৫	অ	এ	ক	ট	ম
৫	অ	এ	ক	ট	ম
৫	অ	এ	ক	ট	ম
৫	অ	এ	ক	ট	ম

Figure 2.2. Some character examples from our data set

Another problem involves the stroke order variations. Figure 2.3 shows a Bangla character written in three different orders. The left-most column shows the first stroke, which is same for all the three sets. Stroke-

order variations come into view in the second column onwards and the final complete character is shown in the right-most column.

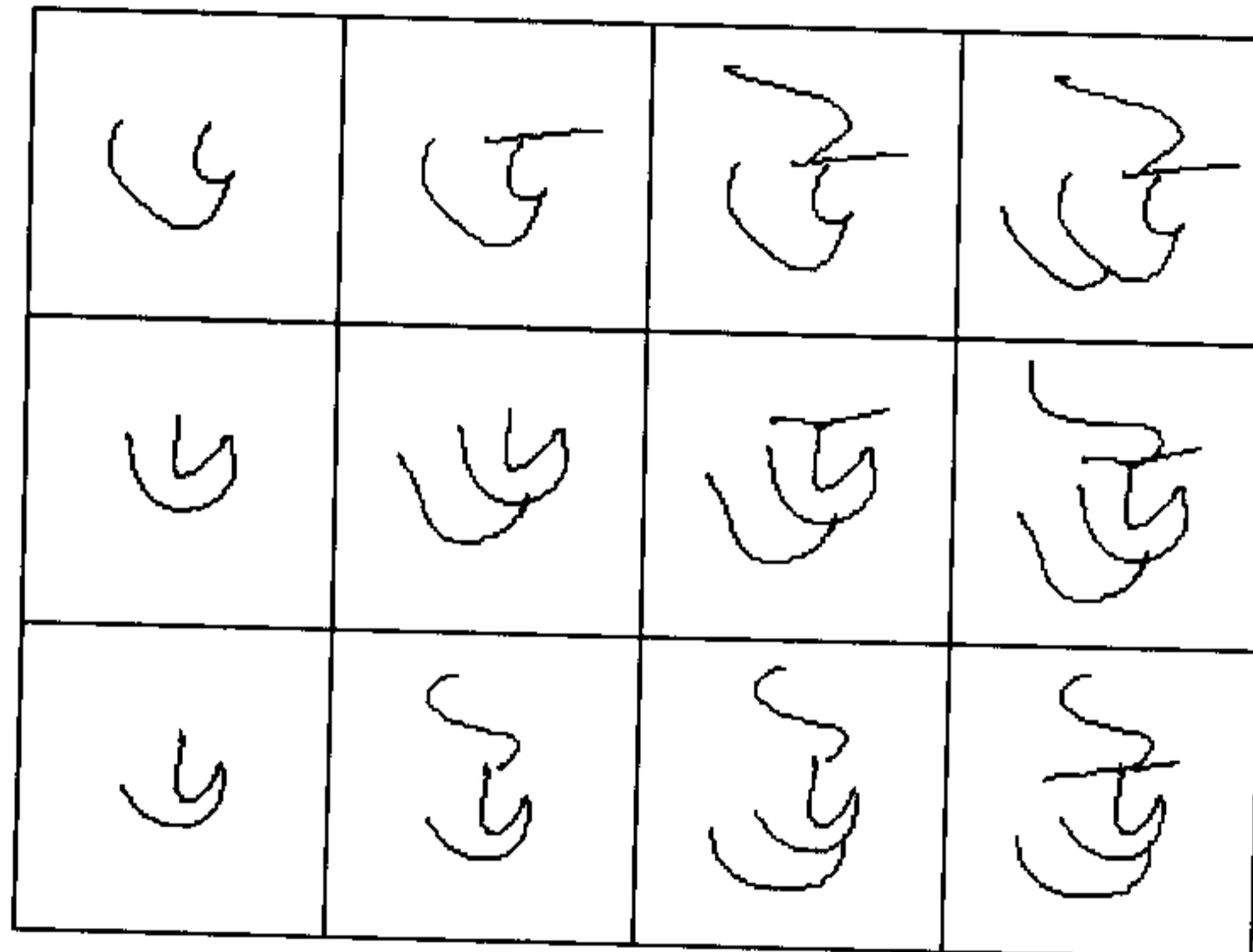


Figure 2.3. Examples of different stroke orders

This thesis proposes a new techniques that is robust against stroke connections as well as shape variation while maintaining reasonable robustness against stroke order variations. Three principal aspects of the proposed algorithm are —

1. Pen trajectory information is converted into a normalized vector, which stores *writing direction* and *trajectory length*.
2. A new distance measure is used to match input and template features.
3. A set of rules is implemented to tackle stroke order and stroke number variations.

2.3.1 Feature Extraction

Our method uses the neuromotor characteristics of handwriting. Basically, we follow the approach of learning of handwriting of a child. While learning a character, a child is taught to draw curve or straight lines by giving instructions like move downwards, move left, move diagonally to bottom-left, create loop, or lift the pen at some other position, etc. Starting position for the next stroke is also taught to a child and (s)he follows such instructions till the character is completed. The relative lengths of the different parts of a stroke are also taught, besides the starting positions and the directions of pen movements. These aspects are used as features. Some characters' writing style is illustrated in the Figure 2.4.

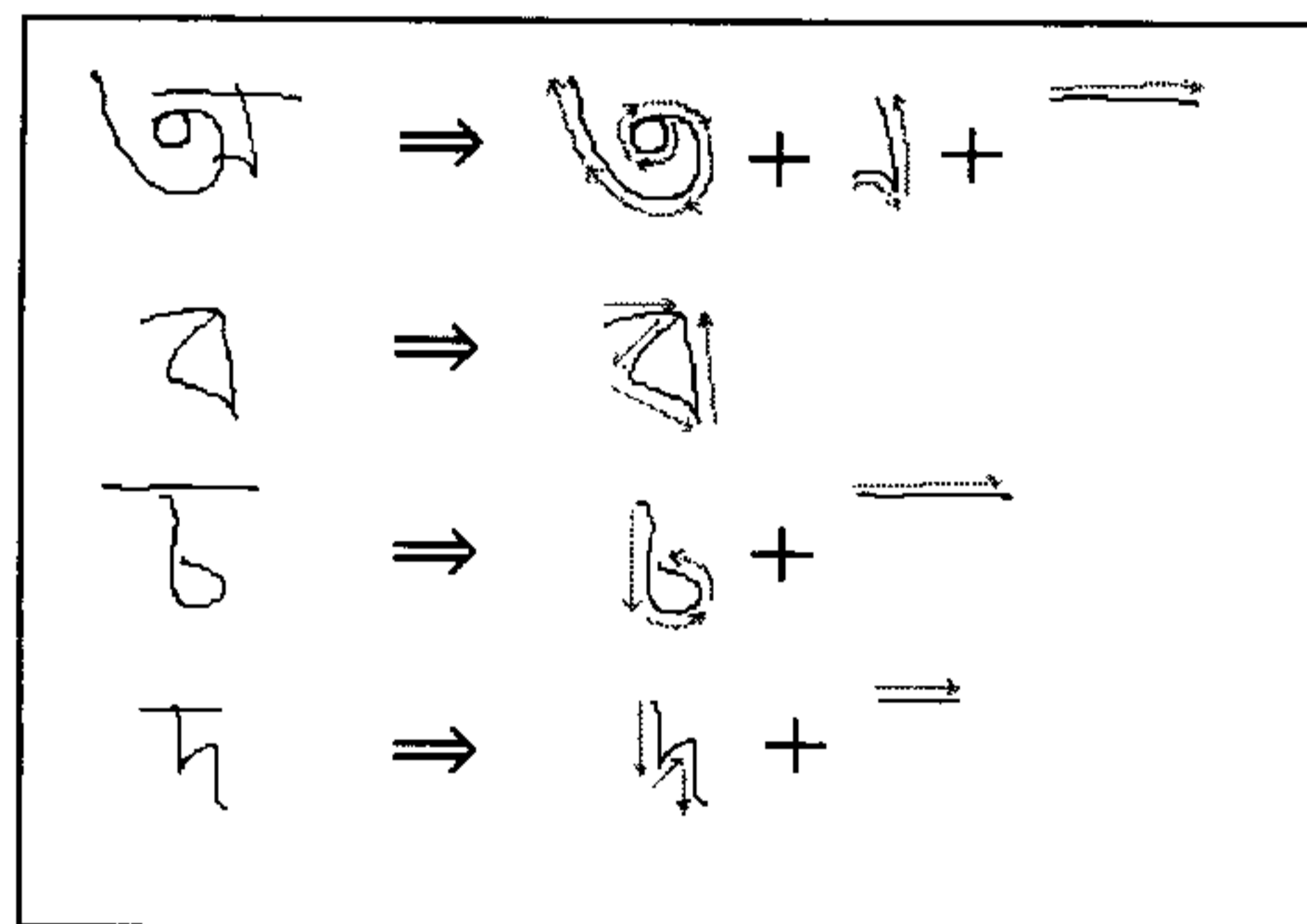


Figure 2.4. Writing style of some characters

To demonstrate our feature extraction process we again use the same notation $\{P_i\}_{i=1}^N$ as described in the data acquisition section. Now we describe how to extract *writing directions*.

Writing direction between two consecutive sample positions is an integer code (like 8-direction Freeman code [15]), which is obtained from the angle between the positive horizontal direction and the line joining the sample positions. Put the first sample point at the centre, then find the number of the region at which the second sample point falls, as shown in the Figure 2.5.

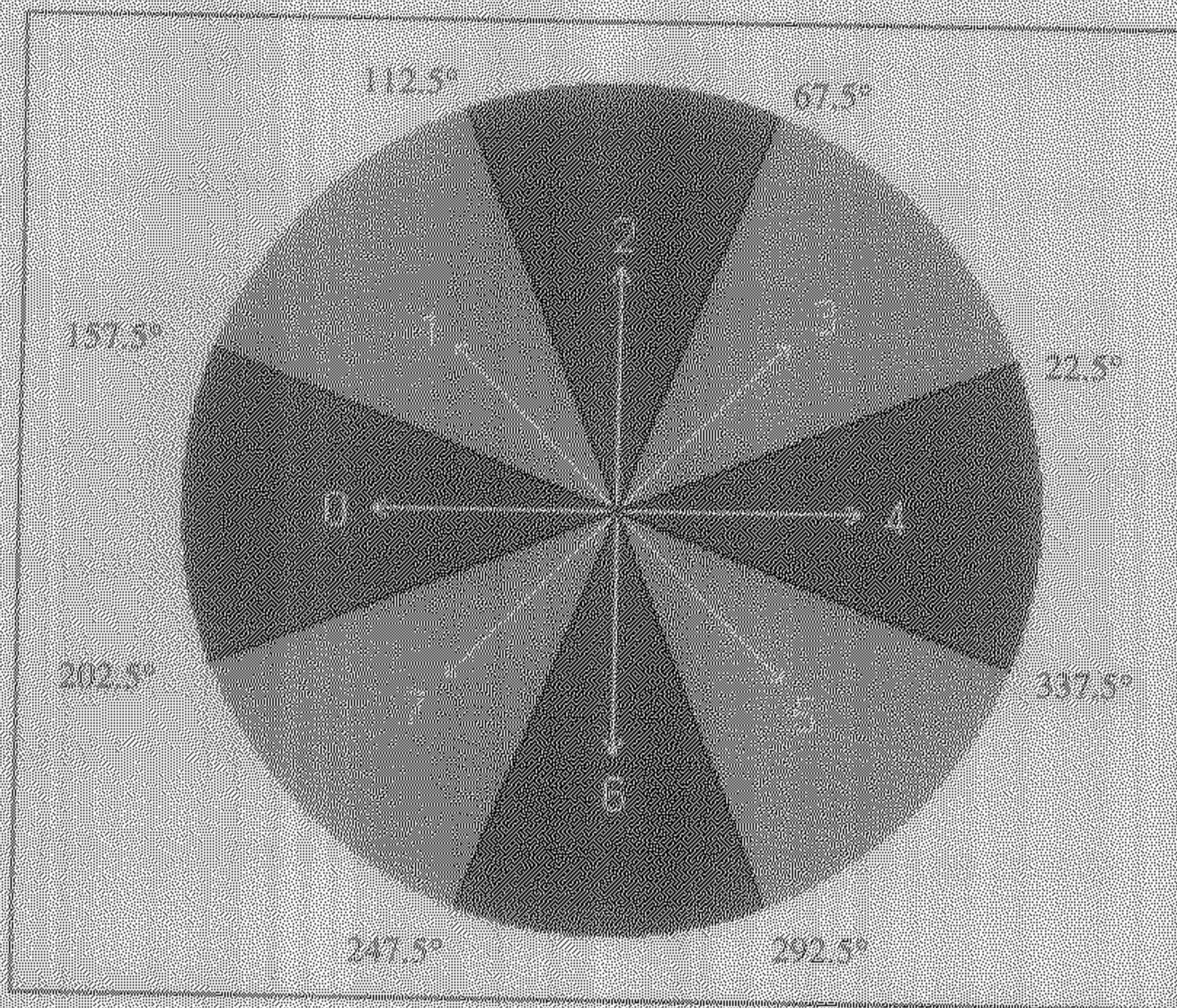


Figure 2.5. Writing direction code

Let $s = P_i - P_{i-1}$, where $i = 1, 2, \dots, N - 1$, $s \in \mathbb{R}^2$ and s has two components (dx, dy) . Now, we find the angle using the following equation.

$$\phi_i = \pi + \text{sign} \cdot \cos^{-1} \left(\frac{dx_i}{dl_i} \right) \quad \dots \quad 2.2$$

where $dx_i = (x_i - x_{i-1})$ and $dy_i = (y_i - y_{i-1})$.

$$dl_i = \sqrt{(dx_i)^2 + (dy_i)^2}$$

if $(dy_i < 0)$ then $\text{sign} = -1$
 else $\text{sign} = 1$

To increase computational efficiency, in Equation 2.2 \cos^{-1} is used to find the angle instead of \tan^{-1} . Because \tan^{-1} signals an error known as '*divided by zero*' if dy_i becomes zero. To get rid of this problem we have to check whether dy_i become zero. Note that, in our problem, very frequently dy_i may become zero. Other than the above problem \tan^{-1} has another difficulty. It returns the angle between $-\frac{\pi}{2}$ to $\frac{\pi}{2}$, which does not cover the whole space. Interestingly, the parameters dx_i, dy_i, dl_i , etc. are already calculated during the preprocessing phase (vide Equation 2.1) and are reused here.

At this point, note that, we have measured two quantities for each consecutive pair of sample points —

1. ϕ_i , the angle between positive horizontal direction and the line segment joining the sample points,
2. dl_i , the Euclidean distance between the sample points.

In reality, instead of this angle, *writing direction* is used to train a child as mentioned earlier. For this reason we convert each ϕ_i into a *writing direction*.

From ϕ_i , using the following equation we can able to find the integer code, d_i .

$$d_i = \left(\left\lfloor \left(16 \cdot \frac{\phi_i}{2\pi} + 1 \right) \right\rfloor \bmod 16 \right) / 2 \quad \dots\dots \quad 2.3$$

where mod is the modulo operator.

Now, we normalize dl_i , which is local *trajectory length*, using the Equation 2.4.

$$dl_i = \frac{dl_i}{\sum_{j=1}^{N-1} dl_j} \quad \dots\dots \quad 2.4$$

Our feature vector is then defined by the couplet (d_i, dl_i) , $i = 1, 2, \dots, N - 1$, where d_i is the *writing direction* and dl_i is the *normalized local trajectory length*.

2.3.2 Data Reduction

Selection of representative prototypes from the training set is presented in this section. Here, we mainly describe a method of editing a training set of templates.

The goal of nearest neighbour editing algorithm is to select only those templates from the training set that fall along the class boundaries in the feature space. These selected prototypes then make up the reduced training set. Common methods of identifying this reduced set involve first deriving the Voronoi Tessellation of the training set, and then removing any training example whose cell does not border the cell of a training pattern of a differing class [16].

A Voronoi Tessellation requires that the training patterns be represented as feature vectors in some d -dimensional Euclidian space. Since our patterns have only their inter-pattern distances, we have used the following algorithm to identify the border patterns. This method relies only on knowing the inter-class nearest neighbour to a character example.

T_0 = Original training set
 T_R = Reduced training set
 C = Set of class labels
 $C(x)$ = The class of pattern x .

Set $T_R = \emptyset$

$\forall x_i \in T_0$

$\forall c \in C$

Find $a = \max_j J(x_i, x_j)$, where $C(x_i) = C(x_j)$

and $b = \min_k J(x_i, x_k)$, where $C(x_i) \neq C(x_k)$

If $(a > b)$

If $x_j \notin T_R$

$T_R \leftarrow x_j$

Algorithm for constructing the Reduced Training Set.

The output of this algorithm, set T_R , contains only those patterns, which have been identified as the nearest neighbour of another pattern, which is across a class boundary; in other words, those training patterns which are near the class boundaries. However, this method retains many more training patterns than the clustering method.

2.3.3 Character Recognition

Our system recognizes characters using two-level approach. For detail description of the recognition system see Appendix B. At first individual strokes are classified by template matching. At next level, using the knowledge of forming characters our system recognizes the character. Knowledge of forming characters means — what are the constituent strokes

of a character, how and in which order they may be connected to form a character, etc.

Let $f_T = (d_i^T, dl_i^T)_{i=1}^K$ denote the feature vector for an input stroke T .

Also, let $f_S = (d_j^S, dl_j^S)_{j=1}^K$ denotes the feature vector for a template stroke S .

Here we try to find out a dissimilarity measure between T and S . Note that,

$$\sum_i dl_i^T = \sum_j dl_j^S = 1.$$

If $K = L (= N)$, T and S can be compared by

$$J(T, S) = \sum_{i=1}^N (d_i^T \dot{-} d_i^S) \quad \dots\dots \quad 2.5$$

where the operator $\dot{-}$ is defined by the Table 2.1. From the Figure 2.5 it is justified that the difference between the writing direction 0 and 7 must be 1, instead of 7. Similarly, the difference between 0 and 6 must be 2, the difference between 1 and 7 must be 2 and so on.

$\dot{-}$	0	1	2	3	4	5	6	7
0	0	1	2	3	4	3	2	1
1	1	0	1	2	3	4	3	2
2	2	1	0	1	2	3	4	3
3	3	2	1	0	1	2	3	4
4	4	3	2	1	0	1	2	3
5	3	4	3	2	1	0	1	2
6	2	3	4	3	2	1	0	1
7	1	2	3	4	3	2	1	0

Table 2.1: Definition of the $\dot{-}$ operator.

The above table can be put in the following analytical form also.

$$d' \circ d'' = 4 - |4 - |d' - d''|| \quad \dots\dots \quad 2.6$$

In reality, K is rarely equal to L . Define $C_k^T = \sum_{i=1}^k dl_i$ and $C_l^S = \sum_{j=1}^l dl_j$.

Now, we sort the union of $(C_k^T)_{k=1}^K$ and $(C_l^S)_{l=1}^L$ together in increasing order. Let the new sequence be $(C_r)_{r=1}^R$, where $C_R = 1$. Obviously, the *writing direction* codes d_i^T of T and d_j^S of S are constant over the interval $C_r - C_{r-1}$. Hence, we can redefine f_T as $(d_r^T, dl_r)_{r=1}^R$ and f_S as $(d_r^S, dl_r)_{r=1}^R$, where $dl_r = C_r - C_{r-1}$. Now the Equation 2.5 becomes

$$J(T, S) = \sum_{r=1}^R dl_r \cdot (d_r^T \circ d_r^S) \quad \dots\dots \quad 2.7$$

Note that, the metric property of J is still remain in Equation 2.7, since $\sum_r dl_r = 1$.

Individual strokes of a character are recognized by the dissimilarity measure given in Equation 2.7. However, recognition is confirmed by using a higher level knowledge where (i) linear / curved stroke, (ii) loop, (iii) sharp change in direction, (iv) stroke beginning / ending position, etc. are considered. A rule base against the formation of each character is maintained in a two dimensional array. The stroke order variation problem is tackled by introducing multiple entries in the above table for a single character. Each such entry defines different stroke sequence for the same character. Consulting the above table our system can recognize the character.

3. Results

We have collected raw data using a tablet of dimension 11.88 inch x 8.88 inch from **FRONTECH**. The sampling rate of this device is 90 points per second. We develop and execute our system on a PC (from **IBM** x86 Family 6 Model 8 Stepping 10) with O/S **Microsoft Windows 98** Ver. 4.10.2222A.

To build up our training set all the 60 strokes as tabulated in Appendix A are considered. We have collected data from thirty native writers who are from different professions — scientists, technical assistants, students, clerks, etc. We have collected two samples of each stroke from each writer. So we have a training set of 60 x 2 x 30 i.e., 3600 strokes.

At first, our experiment concerns the performance of the two coding schemes with eight and sixteen writing directions, respectively (vide Section 2.3.1). For this purpose, we have created two models Λ_8 and Λ_{16} trained with eight and sixteen writing direction codes, respectively. Their performance on the training set reveals that the 8-direction coding model Λ_8 performs better than 16-direction coding model Λ_{16} , which is expected (vide Table 3.1). Though the model Λ_{16} minimizes quantization error, it is more sensitive than the model Λ_8 to insignificant and irrelevant variations of pen movement. The recognition rate for the model Λ_8 is 96.34% while that for the model Λ_{16} is 93.94%.

Models	Recognition rate (in %).	
	Bangla Characters	Bangla Numerals
Λ_8	96.16	97.43
Λ_{16}	93.82	95.45

Table 3.1. Performance of two models on writer-dependent data.

For writer-dependent data (i.e., training set is created by a particular user and the testing is performed on the handwriting of that particular writer) the performance of our system, as expected, is better than that for writer-independent data (i.e., training set is our combined set of strokes from all thirty writers). The results are presented in the Table 3.2.

Writer #	Recognition rate (in %).	
	Bangla Characters	Bangla Numerals
1	98.67	99.12
2	98.42	99.18
3	98.15	99.06
4	98.05	99.30

Table 3.2. Performance for writer dependent data.

Test on combined data set (characters as well as numerals) shows 96.34% accuracy for Bangla. Our result seems good though till now no benchmark data set is available to test the classifier's performance. But this high accuracy strongly supports our model for selecting features and designing the classifier. Moreover, low complexity of the classifier gives recognition throughput as high as 60 characters per second on a 750MHz, Pentium-4 machine. Note that, the accuracies reported here are on discrete characters and only the top choice is considered.

There are two kinds of recognition errors —

- Misclassification
- Rejection

For our system the first kind of errors occurs approximately at a rate of 2.35% while the second kind of errors occurs only at a rate of 1.31%. Our system may confuse to recognize characters having high shape similarity. The character pairs having the shape similarity are listed in Figure 3.1. If a multi-stroke character is written with very uncommon stroke order or stroke shape variation is very unnatural, our system then cannot recognize the character.

ক ফ ক ফ	খ ঘ খ ঘ	গ ণ ন গ ণ ন
ঙ ভ ঙ ভ	চ ঢ চ ঢ	ঠ ব ঠ ব
ড ড ড ড	ত ত ত ত	থ য থ য
ম স ম স	ল শ ল শ	ং হ ং হ
	৩ ১ ৩ ১	

Figure 3.1. List of characters having high shape similarity

4. Conclusions

The scheme proposed for online handwritten Bangla script recognition is a new one. It observes at the human motor functionality to develop the feature vectors. Our scheme is also sufficiently general. With some minor extension, our proposed scheme may be applied to other alphabet-based languages like Devanagari, Oriya, Panjabi, Gujrati, Tamil, Telugu, etc. Dissimilarity measure between two strokes is also very robust. The scheme is capable of producing commercial level of accuracy.

Future scope

Further improvements of our system performance may be done in the following directions. Instead of only one top choice of recognized characters more than one choice (say n) may be considered. Using word level contextual information (like spelling or character bigram probability) we then able to find the more correct choice of recognized character. To improve the recognition of characters written with unnatural stroke order variations we may add more rules to the rule base (vide last paragraph of section 2.3.3).

Since handwritten text is more natural form of communication for human, in near future many software products may integrate these kinds of recognition systems to their interfaces. Though data is entered into computers by the keyboard, still today many tasks exist where people tend to prefer

handwriting to keyboard entry. In a classroom, note-taking task still be done more efficiently by hand for most users. Using a pen along with this kind of system, if an user can able to enter data instead of using a combination of the mouse and keyboard it will be time saving. Finally, communicating with computers by some language script having large number of symbols, this kind of system will be very helpful.

Appendix A

Stroke No.	Shape	Description	Stroke No.	Shape	Description
1	১	Numeral one	31	য	Consonant character
2	২	” two	32	ল	”
3	৩	” three	33	শ	”
4	৪	” four	34	ষ	”
5	৫	” five	35	স	”
6	৬	” six	36	হ	”
7	৭	” seven	37	ং	”
8	৮	” eight	38	এ	Vowel character
9	৯	” nine	39	ও	”
10	০	” zero	40	ঞ	Lower part of ঙ
11	ক	Consonant character	41	।	Right part of অ, ঞ, ঞ
12	খ	”	42	ূ	Upper part of ট, ঙ
13	গ	”	43	ৃ	Lower part of উ
14	ঘ	”	44	ৄ	Right part of জ
15	ঙ	”	45	৅	Right part of ঞ
16	চ	”	46	৆	Lower part of প
17	ছ	”	47	ে	Upper part of প
18	ট	”	48	ৈ	Lower part of ং
19	ড	”	49	৉	Lower part of ূ
20	ঢ	”	50	৊	Upper part of ি
21	ণ	”	51	।	Vowel marker আ
22	ত	”	52	৊	Vowel marker ঙ
23	থ	”	53	৊	Vowel marker উ
24	দ	”	54	৊	Alt. form of marker উ
25	ধ	”	55	৊	Vowel marker উ
26	ন	”	56	৊	Vowel marker ঞ
27	ফ	”	57	৊	Vowel marker এ
28	ব	”	58	।	Consonant marker য
29	ভ	”	59	।	Consonant marker র
30	ম	”	60	।	Lower part of র

Table A.1 : Description of different strokes of Handwritten Bangla Characters.

Appendix B

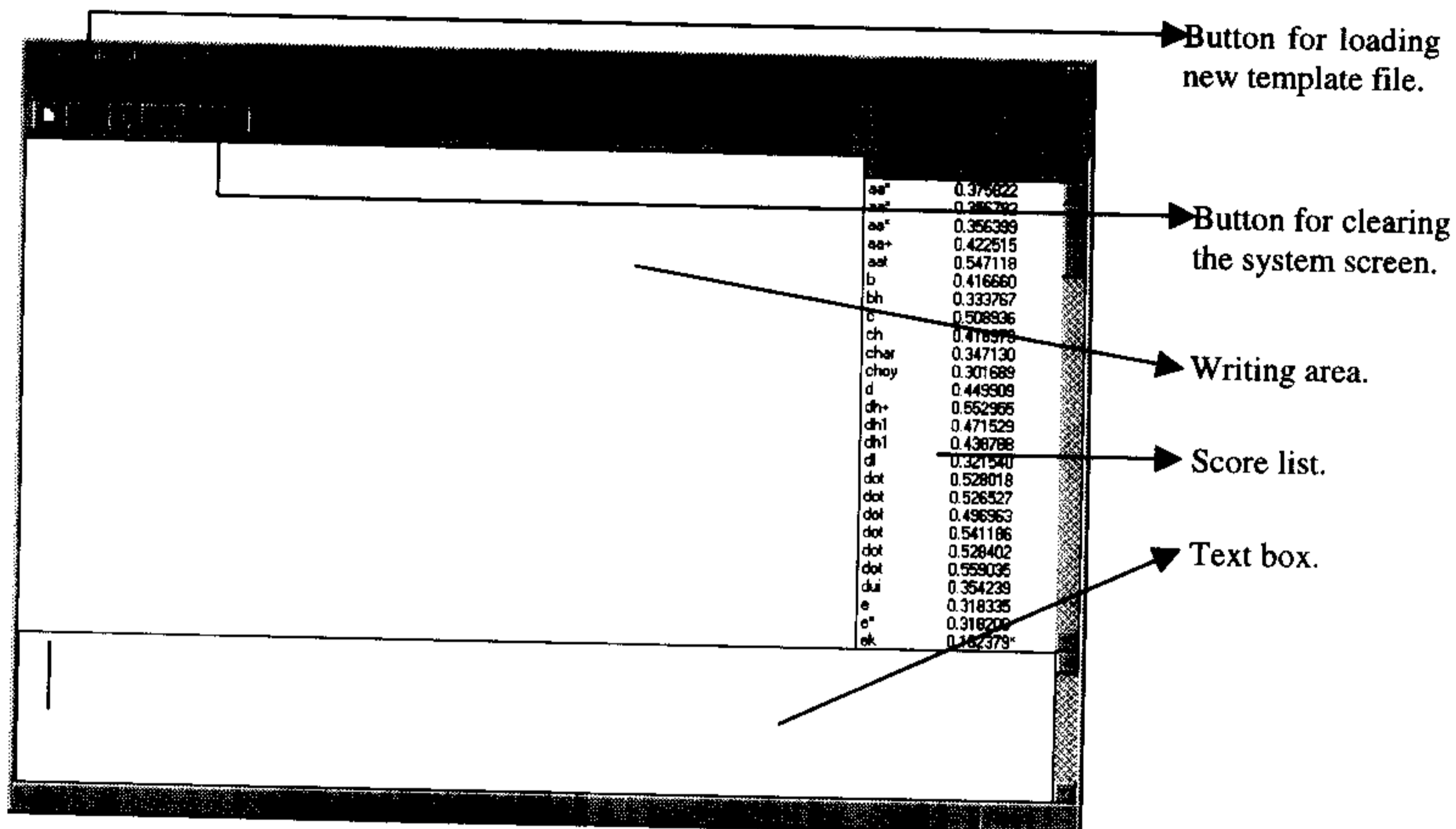


Figure B.1: Explanation of different parts of the interface of our system

Writing area: This is the place where user can get feedback of his/her handwriting, while (s)he writes on the tablet.

Text box: This is a text editor, where recognized characters are appended sequentially. Since it is an editor, user can easily make necessary corrections in the recognized text.

Score list: This is a list of all the stroke templates along with their dissimilarity measure from the current input stroke. This list has no use of the end user of the system.

How the system works

When the user touches the tablet by the stylus our system starts to sample pen position in equal time interval until the user lifts the stylus from the

tablet. As soon as the stylus is left, our system stops sampling and recognize the stroke just written. The recognized stroke number along with other relevant information is then stored in an array. After writing the whole character, user has to wait for 500 milliseconds. Within this time period if no input come to the system, it will try to recognize whole character using the rule base table (vide last paragraph of Section 2.3.3).

Some characteristics

- Using a very simple file format, templates may be stored in a file. User can load new set of templates, by just giving the corresponding file. Format of template file is given at Appendix C.
- When user completes writing, the recognized characters will appear in the text box, where user can make necessary corrections.
- A clear button is provided for removing all inputs and outputs from the screen so that user can start afresh.

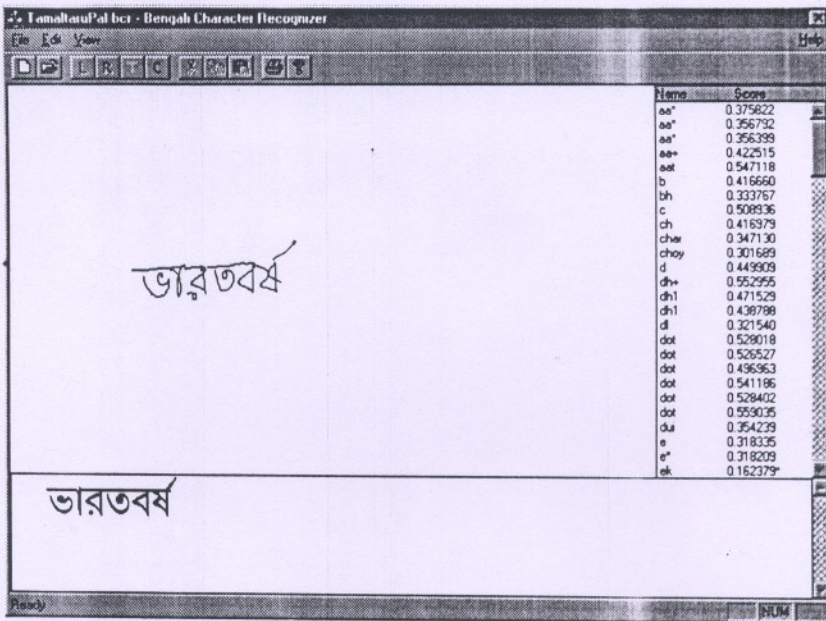


Figure B.2: Snapshot of our system in action

Appendix C

```
12

one 8
5 0.408093 7 0.171958 0 0.067168 1 0.113654 3 0.093870 4 0.050376
5 0.061296 6 0.033584

two 3
5 0.300936 7 0.302938 5 0.396126

three 9
0 0.012381 1 0.027685 3 0.115011 4 0.037143 5 0.085260 6 0.177204
7 0.088927 1 0.425038 3 0.031352

zero 9
7 0.219374 6 0.106934 5 0.046387 4 0.095597 3 0.064359 4 0.017288
2 0.174179 1 0.125480 0 0.150403

four 10
1 0.014610 0 0.010331 1 0.011550 7 0.091183 5 0.282302 7 0.099408
0 0.036158 1 0.075421 3 0.289166 0 0.089870

four 17
3 0.019859 2 0.017763 1 0.069872 0 0.044407 7 0.041131 6 0.014043
7 0.009930 6 0.022203 5 0.074768 4 0.093464 5 0.071740 6 0.045127
7 0.100311 0 0.039966 1 0.054681 2 0.104670 3 0.176066

five 11
7 0.153528 6 0.061024 5 0.043302 4 0.080456 7 0.116673 0 0.115456
1 0.038636 2 0.105451 3 0.163935 4 0.061024 5 0.060516

six 6
6 0.234008 3 0.137856 6 0.140473 7 0.088465 0 0.138256 1 0.260942

seven 9
7 0.145240 1 0.091267 2 0.050036 3 0.103115 4 0.050036 5 0.109914
6 0.410127 0 0.016679 1 0.023587

eight 4
6 0.377354 3 0.187902 1 0.176238 4 0.258506

nine 10
2 0.046660 3 0.039385 4 0.046660 5 0.052168 6 0.062214 7 0.104335
3 0.205101 2 0.087380 1 0.338708 5 0.017389

nine 11
7 0.012380 0 0.011073 1 0.024759 3 0.057101 4 0.022145 6 0.117571
2 0.293636 7 0.012380 5 0.267332 7 0.107445 0 0.074178
```

Figure C.1: Contents of a typical template file.

The entry in the first line represents the number of templates present in the file. Then after a blank line, entries for templates are kept. The entries are separated by blank lines. Each such entry consists of stroke name and the number of template vectors followed by the template vectors, which are nothing but couplets of *writing direction* and normalized *trajectory length* (vide last paragraph of Section 2.3.1).

References

- [1] B. B. Chaudhuri and U. Pal, "A Complete Printed Bangla OCR System", *Pattern Recognition*, vol. 31, no. 5 pp. 531-549, May, 1998.
- [2] B. B. Chaudhuri and U. Pal, "An OCR System to Read Two Indian Language Scripts: Bangla and Devanagari", *Proc. 4th Int. Conf. Document Analysis and Recognition*, Ulm, Germany, pp. 1011-1015, August, 1997.
- [3] V. Bansal and R. M. K. Sinha, "On how to Describe Shapes of Devanagari Characters and Use them for Recognition", *Proc. 5th Int. Conf. Document Analysis and Recognition*, Bangalore, India, pp. 410-413, 1999.
- [4] Scott D. Connell, R. M. K. Sinha, and A. K. Jain, "Recognition of Unconstrained On-Line Devanagari Characters", *Proc. 15th ICPR*, pp. 368-371, 2000.
- [5] R. Plamondon and S. N. Srihari, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63-84, January, 2000.
- [6] C. C. Tappert, C. Y. Suen, T. Wakahara, "The State of the Art in On-Line Handwriting Recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 179-190, August, 1990.
- [7] E. J. Bellegarda, J. R. Bellegarda, D. Nahamoo, and K. S. Nathan, "A Probabilistic Framework For On-Line Handwriting Recognition", *Proc. IWFHR III*, Buffalo, New York, pp. 225-234, May, 1993.

- [8] A. K. Jain and D. Zongker, "Representation and Recognition of Handwritten Digits Using Deformable Templates", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 12, pp. 1386-1391, December, 1997.
- [9] S. Jaeger, S. Manke, J. Reichert, and A. Waibel, "Online handwriting recognition: the NPen++ recognizer", *Int. Journal on Document Analysis and Recognition*, vol. 3, no. 3, pp. 169-180, 2001.
- [10] M. Schenkel, I. Guyon, and D. Henderson, "On-Line Cursive Script Recognition Using Time Delay Neural Networks and Hidden Markov Models", *Proc. ICASSP '94*, vol. 2, pp. 637-640, 1994.
- [11] S. D. Connell and A. K. Jain, "Template-based Online Character Recognition", *Pattern Recognition*, vol. 34, no. 1, pp. 1-14, 2001.
- [12] M. Kobayashi, S. Masaki, O. Miyamoto, Y. Nakagawa, and Y. Komiya, "RAV (reparameterized angle variations) algorithm for online handwriting recognition", *Int. Journal on Document Analysis and Recognition*, vol. 3, no. 3, pp. 181-191, 2001.
- [13] J. H. Kim and B. Sin, "Online Recognition of Korean Hangul Characters", *Handbook of Character Recognition and Document Image Analysis*, pp. 381-396, Eds. H. Bunke and P. S. P. Wang, World Scientific Publishing Company, 1997.
- [14] E. Mandler, R. Oed, and W. Doster, "Experiments In On-Line Script Recognition", *Proc. 4th Scandinavian Conf. Image Analysis*, pp. 75-86, June, 1985.
- [15] H. Freeman, "On the digital computer classification of geometric line patterns", *Proc. National Electronics Conf.*, vol. 18, pp. 312-324, 1962.
- [16] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, Wiley – Interscience, 1973.