

Spatial Similarity based Stroke Number and Order Free Clustering

Santosh K.C., Cholwich Nattee, Bart Lamiroy

▶ **To cite this version:**

Santosh K.C., Cholwich Nattee, Bart Lamiroy. Spatial Similarity based Stroke Number and Order Free Clustering. International Conference on Frontiers in Handwriting Recognition, Nov 2010, Kolkata, India. inria-00516726

HAL Id: inria-00516726
https://inria.hal.science/inria-00516726

Submitted on 11 Sep 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Spatial Similarity based Stroke Number and Order Free Clustering

Santosh K.C.*, Cholwich Nattee† and Bart Lamiroy‡
*INRIA Nancy-Grand Est, ‡Nancy Université INPL
LORIA - Campus Scientifique, BP 239 - 54506 Vandoeuvre-lés-Nancy Cedex, France
Email: Santosh.KC, Bart.Lamiroy@loria.fr
†School of ICT, SIIT, Thammasat University, Bangkadi Campus, Pathumthani 12000, Thailand
Email: cholwich@siit.tu.ac.th

***Abstract*—In this paper, we present an innovative approach to integrate spatial relations in stroke clustering for handwritten Devanagari character recognition. It handles strokes of any number and order, writer independently. Learnt strokes are hierarchically agglomerated via Dynamic Time Warping based on their location and their number and stored accordingly. We experimentally validate our concept by showing its ability to improve recognition performance on previously published results.**

***Keywords*-Stroke Spatial Information: Stroke Number and Order free; Hierarchical Clustering; Devanagari Handwriting Recognition**

I. Introduction

A. Motivation

Pencil and paper can be preferable for anyone during a first draft preparation instead of using keyboard and other computer input interfaces, especially when writing in languages and scripts for which keyboards are cumbersome. Those for Devanagari, for instance, are cumbersome to use, although they are used by millions of people. In this paper, we develop an on-line writer independent natural handwritten character recognition system for Devanagari by considering the number of strokes and their spatial relation. We validate our concept with the help of 25 native writers for 36 classes of characters and achieve recognition rate of more than 95%.

B. Structure of Devanagari Script

Devanagari is used to write several Indian languages including Sanskrit, Nepali, Hindi, Marathi, Pali, Kashmiri, Sindhi, and sometimes Punjabi. It is written from left to right with a horizontal line on the top which is known as *shirorekha* from which the text(s) is(are) suspended. In order to clearly establish the semantics of our reasonment, we introduce the following vocabulary distinction: the term *"character"* refers to a token having a lexicographic meaning while *"symbol"* refers to one of its handwritten graphical representations. In Devanagari, significant shape variations may occur between the symbols representing the same character. In addition, visually very similar *symbols* – even from the same writer – may represent different *characters* due to individual writing styles. Consequently, confusion is likely to occur with their handwritten *character* counterparts like, (क, फ), (छ, ध), (य, प), (स, झ), (ढ, द), (इ, ड), (ठ, ट), (न, त), (व, त), (च, थ) and (ढ, ट) *etc*. Fig. 1 shows a few samples of discrete natural handwritten characters with *structure similarity*. Besides *writing units* like stroke direction, stroke number and order, speed in writing, tilting angle, and stroke size (big or small) are always varied. Overall, unconstrained Devanagari writing is more complex than English cursive [1].

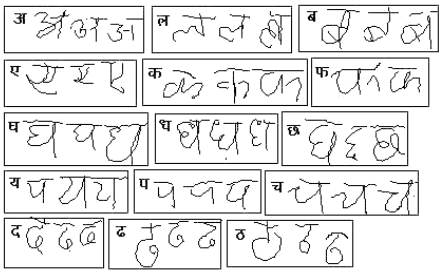


Figure 1. Few samples

C. Related Work

Research on on-line handwriting provides an extremely rich state of the art literature and more specifically, template based approaches have been dominant [2]–[6]. Our approach is inspired by Cluster generative Statistical Dynamic Time Warping (CSDTW) technique [5] but, varies distinctly that lies in the clustering technique. As mentioned earlier, Connell *et al.*, 2000 [1] illustrated the difficulty with the cursive nature of *Devanagari* writing. Due to its structural complexity, Niranjan *et al.*, 2005 [7] focused on structural properties and separated *shirorekha* based on directional code such as *east* or *west*. The work is writer dependent and is limited to natural handwriting where curve sequence can be *shirorekha*. In the literature, there has been extensive use of a unique feature and varies according to the nature of the script. Use of *shirorekha* in [8] to make *Devanagari* difference with other scripts, and *hat* feature [9] in Urdu handwriting for higher recognition confidence are two concrete examples. Besides, it is important to integrate relative positioning

knowledge of strokes by referencing such a unique feature that leads to better classification.

Handling stroke spatial relation however, in on-line handwriting is not the new approach [10]–[12]. But, the approaches are not globally extended. Swethalakshmi *et al.*, 2007 [13] mentioned spatio-structural feature based on existing method – zoning information, to recognise Devanagari and Tamil scripts, and surprisingly provided 95% and 92% recognition rates respectively. The approach however, is very sensitive to handwriting with asymmetric structure, symbols having very long ascender and/or descender for instance. In this paper, we adopt pairwise strokes spatial relation by fixing *shirorekha* as a reference within a symbol as in [8]. To the best of our knowledge, employing stroke spatial relation for Devanagari characters conveys a novel idea.

D. Proposed Technique

Especially because of the structure of Devanagari, it is necessary to pay attention for correctly structuring the data to ease and speed up comparison between the handwritten strokes, rather than just relying on global recognition techniques that are based on the whole form of the character as well as stroke number and order. This in turn affects efficiency and time complexity of the recogniser. In these respects, we develop a method based on clustered strokes and their spatial information. These aspects compared to previous works [5], [11], [13] mainly consist of the following four phases. Our learning module includes the first three phases and remaining, classification module as described in [6].

- organise the symbols representing the same character are into different groups based on the number of strokes

For a specific class of character, it is interesting to note, however, that writing symbols with the same number of strokes, generally produce a visually similar and easier to compare structure. This is how we motivate to group symbols based on the number of strokes, and find spatial relations.

- find the spatial relation between strokes

The importance of the location of the strokes is best observed by taking a few pairs of characters that often lead to confusion of interpretation: (भ, म), (ध, घ), (थ, य) *etc.* The first character in every pair has visually two distinguishing features: its particular location of the *shirorekha* (more to the right) and a small curve in the text. There is no doubt, one of the two features is sufficient to automatically recognise but small curves are not a robust feature in natural handwriting. Therefore, finding the location of the *shirorekha* only can avoid confusion. Our stroke based spatial relation technique is explained further in section II-A2.

- agglomerate (pairwise) similar strokes from the specific location within a group

In every group, a representative symbol is synthetically generated from pairwise similar strokes merging, which are positioned identically with respect to the *shirorekha* by using 'Dynamic Time Warping' (DTW) algorithm. In a similar way, learnt strokes are stored according to the location of the strokes.

- stroke-wise matching for classification

We align individual test stroke with the learnt strokes having both identical number of strokes and spatial properties. Overall, symbols can be compared by fusion of matching information from individual strokes.

Our learning module is developed in section II: it mainly includes stroke clustering method. Section III covers our classification module and section IV gives an overview of the obtained results and performance evaluation. We conclude the paper in section V

II. Learning

The digitiser captures a series of strokes during pen movement. A string of coordinates (pen-tip positions) from pen down to pen up movement represents a stroke. There are m-set of strokes in a symbol $\mathbf{S}$. Each stroke $\mathbf{s}^j$ consists of a string of coordinates.

$$\begin{aligned} \mathbf{s^j} &= \left[\mathbf{p}_1^j, \mathbf{p}_2^j, \ldots, \mathbf{p}_l^j\right], \quad \mathbf{p}_k^j = \left(x_k^j, y_k^j\right) \\ \mathbf{S} &= \left[s^1, s^2, \ldots, s^m\right] \end{aligned}$$

Strokes directly collected from users are often incomplete and noisy that jointly represnting a particular character. We use a three step of basic pre-processing.

- Size Normalisation: We scale the symbol $\mathbf{S}$ into a $[0,1] \times [0,1]$ unit square.
- Noise Elimination: Elimination of noisy sequences enhances recognition accuracy as well as speed but, it is a difficult task in cursive writing. We employ a two step noise elimination process.
  - *deletion of shorter sequences*: We delete shorter sequences ($\leq$ 5 coordinates) as it does not give any information about the character.
  - *chopping of undesirable hook/cusp*: If the trajectory path turns sharply (angle changes drastically i.e., $80^\circ - 100^\circ$) both at ascender and descender of a sequence then the sequence is chopped. This is done for 5-10 coordinates. Fig. 2 shows the complete idea of noise elimination. A drastic change in the tangent angle from the point p_3 gives rise to a cusp, is therefore chopped up to the last point.
- Co-occurrence Coordinates Deletion: We delete co-occurrence of coordinates $\mathbf{p}_k = \mathbf{p}_{k+1} = \mathbf{p}_{k+d}$ for some k and d as in [5]. It smooths shape and reduces delay.

Feature selection is able to distinguish the classes and it has no doubt that it varies from one script to another [8],

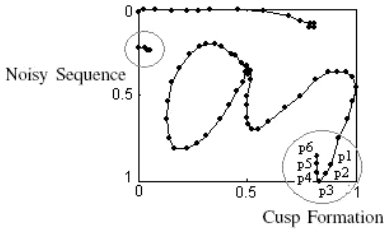


Figure 2. A sample of pre-processing

[14]–[16]. In this work, a feature vector sequence of every j^{th} stroke is [6], [16],

$$\mathbf{F^j} = \left[\left(\mathbf{p}_1^j, \alpha_{\mathbf{p}_1^j,\mathbf{p}_2^j}\right), \ldots, \left(\mathbf{p}_l^j, \alpha_{\mathbf{p}_{l-1}^j,\mathbf{p}_l^j}\right)\right]$$

where, $\alpha_{\mathbf{p}_{l-1}^j,\mathbf{p}_l^j} = \arctan\left(\frac{y_l^j - y_{l-1}^j}{x_l^j - x_{l-1}^j}\right)$.

A. Stroke Clustering

Basically, clustering is a process of grouping items which are similar in some way [17]. Items of one group are dissimilar with other items belonging to other groups.

As mentioned in the section I-D, we first group the symbols based on the number of strokes used to complete a particular character. Then, we agglomerate similar strokes from the specific location within the group. To use spatial relation between the strokes it is important to take proper reference. In this paper, we use *shirorekha* as is expressed in [8]. The question of how we identify *shirorekha* from the pool of strokes within a character is explained in the following section.

1) Shirorekha Identification: Basically, location of the *shirorekha* is assumed on the top portion and the text(s) is(are) clearly determined with respect to the *shirorekha*. But in natural writing, it may not always be strictly horizontal and not always exactly on the top (Fig. 3). It may sometimes intersect with the text and be a small curve. In general however, symbol has at least two kinds of strokes: straight and curve. Naturally, a straight stroke is more likely to represent a *shirorekha*. To identify whether a stroke is straight one, we check the straightness characteristic S_t for every j^{th} stroke,

$$S_t = \frac{d\left(\mathbf{p}_1^j, \mathbf{p}_l^j\right)}{\sum_1^{l-1} d\left(\mathbf{p}_i^j, \mathbf{p}_{i+1}^j\right)} = \begin{cases} 0.8 \leq St \leq 1 & \text{:Straight} \\ \text{otherwise} & \text{:Curve} \end{cases}$$

where, $d(\mathbf{p}_1^j, \mathbf{p}_2^j) = \sqrt{(x_1^j - x_2^j)^2 + (y_1^j - y_2^j)^2}$.

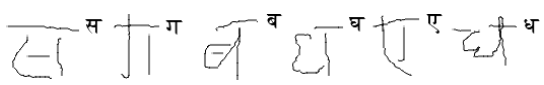


Figure 3. Two kinds of strokes

For the two-stroke handwritten characters, it is easy to identify the *shirorekha* but, confusion occurs in the presence of more than two strokes since there may potentially be many straight sequences. Fig. 3 shows a sample of it. In such a case, we check the following conditions one after another:

- spatial relation between the straight sequences,
- width of the sequence i.e., $w = x_{max} - x_{min}$, and
- global stroke direction from initial to the end coordinate i.e., $\theta_{\mathbf{p}_1,\mathbf{p}_l} = \arctan\left(\frac{y_l - y_1}{x_l - x_1}\right) \times \left(\frac{180^\circ}{\pi}\right)$.

We select the *shirorekha* as the stroke which is located on the top with respect to others. We find a possibility of being two strokes on the top due to re-writing *shirorekha* for instance. For this case, we take the one which has largest width. It is quite rare to get first two conditions identical, but it occurs in few cases and we check for minimum global stroke direction.

2) Pairwise Stroke Spatial Relation: We use six spatial predicates (2×3 regions) for directional relations:

$$\begin{bmatrix} \textit{Top Left (T-L)} & \textit{Top (T)} & \textit{Top Right (T-R)} \\ \textit{Bottom Left (B-L)} & \textit{Bottom (B)} & \textit{Bottom Right (B-R)} \end{bmatrix}.$$

To confirm the location of the stroke, two models are used. They are

- Projection Model: Minimum Boundary Rectangle (MBR) [18] and
- Angle based Model: Bi-centre [19] and angle histogram [20].

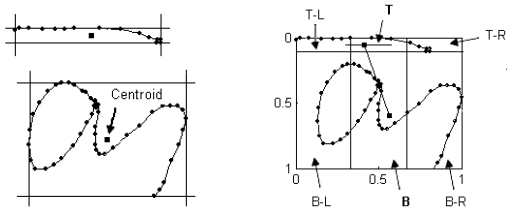
Based on [21], we start with checking fundamental topological relations such as, *Disconnected* (DC), *Externally Connected* (EC) and *Overlap/Intersect* (O/I) relations considering two strokes $\mathbf{s^j} = \left\{\mathbf{p}_k^j\right\}_{k=1\ldots l}$ and $\mathbf{s^{j'}} = \left\{\mathbf{p}_{k'}^{j'}\right\}_{k'=1\ldots l'}$ as follows,

$$\mathbf{s^j} \cap \mathbf{s^{j'}} = \begin{cases} 1 & \text{if } (\mathbf{p}_k^j \cap \mathbf{p}_{k'}^{j'} \neq \emptyset) \Rightarrow \text{EC, O/I} \\ 0 & \text{otherwise} \Rightarrow \text{DC.} \end{cases}$$

We use border condition from the geometry of the MBR. It is straight forward for DC strokes while, is not for EC and O/I relations. In the later case, we check the level of centroid. For example, if a boundary of the shirorekha is above the centroid of the text, then it is confirmed that the *shirorekha* is on the top portion. This procedure is applied to all of the six previously mentioned predicates.

On the other side, we use discretised angle [22] made by two centre of the strokes. The bi-centre model cannot produce any measure in case of centroid coincidence (an overlapping case, for instance) as well as it does not convey proper direction in case of truly concave text (curve sequence). Therefore, angle histogram approaches are found to be appropriate in those cases at the risk of time complexity.

Fig. 4 shows an example for a two-stroke क. Further, for symbols with two *shirorekha*, the first *shirorekha* is the reference for another based on the order.



(a) MBR model (b) Bi-centre model

Figure 4. Pairwise spatial relation for a two-stroke क (see Fig. 2)

3) Agglomerative Hierarchical Clustering: We find a new cluster from pairwise merging of similar strokes based on DTW [23] which are positioned identically with respect to the shirorekha. The averaging is done by taking discrete warping path along the diagonal DTW-matrix. This pairwise merging process is repeated until it reaches the cluster threshold. We vary threshold from one group to another since the number of symbols is different. However, the total number of cluster representatives is fixed at the end for every class. Fig. 5 shows the proposed clustering for a two-strokes vowel अ.

B. Template Management

As mentioned earlier in section I-D, we respect the importance of stroke location and manage threshold carefully. For example, in case of *shirorekha*, the number of clusters is equal to the number of different locations: *Top*, *Top-Right* and *Top-left*. In a similar way, we store the structured learnt strokes according to the location in a group. Fig. 6 shows a template management for different groups of handwritten vowel अ by taking a small sample.

C. Weight Determination

Our approach allows for the determination of the probability that strokes are at specific location in a specific group. The probability of occurrence of a template in every specific location can be obtained by using the number of strokes and users. We shall use this information as a weight in section III. It gives the trend of the users' writing style by showing, for example, how often the *shirorekha* is exactly at the top position. For better understanding, let us take the group of handwritten characters having two strokes अ in Fig. 5. The probability for the *shirorekha* being on top is the number of *shirorekha* on top divided by the number of users, which is 1/6 = 16.7%. Similarly, the probability of the *shirorekha* being at the top-right is 5/6 = 83.4%, and being at the top-left is 0%. For the text part, the probability of it being at the bottom is 1/6 = 16.7%, at the bottom-right is 0% and at the bottom-left is 5/6 = 83.4%.

III. Classification

We use simple stroke-by-stroke template matching process. Our aim is to classify a given unknown symbol by giving it the corresponding character label.

Let $\mathcal{T}$ be the set of template strokes and an unknown test symbol S composed of n strokes; $S = \{s_i\}_{i=1\ldots n}$. For each known character c, the strokes $\{s_i\}$ are matched with the corresponding ones (*i.e.* those positioned similarly with respect to their *shirorekha*) from the templates in $\mathcal{T}_c$. The set of corresponding templates is $\mathcal{T}_c|_{s_i} = \left\{\tau^t_{c,i}\right\}$. Every matching produces a matching score that is represented in the distance array,

$$\mathbf{M} = \begin{pmatrix} \mathbf{D}_1 \\ \vdots \\ \mathbf{D}_n \end{pmatrix} = \begin{pmatrix} \mathbf{T}^1_1, \mathbf{T}^2_1, \ldots, \mathbf{T}^\kappa_1 \\ \vdots \\ \mathbf{T}^1_n, \mathbf{T}^2_n, \ldots, \mathbf{T}^\kappa_n \end{pmatrix}.$$

$\mathbf{T^c_i}$ contains the matching scores between s_i and the corresponding templates mentioned above (κ = number of classes). $D^t_{c,i}$ refers to matching score via DTW. We then take weighted matching scores,

$$\overline{D^t_{c,i}} = D^t_{c,i}/weight(c,i)$$

where, $weight(c,i)$ corresponds to template $\tau^t_{c,i}$ that gives the probability of occurrence as explained in the section II-C. In order not to miss the similar strokes, we put threshold $\epsilon_{c,i} = min_t\left(\overline{D^t_{c,i}}\right) + \varepsilon_i(fixed)$, for every stroke s_i and count the number of matching scores close to minimum value:

$$W_{c,i} = \sum_{l=1}^{t} C\left(\overline{D^l_{c,i}}, \epsilon_{c,i}\right)$$

where,

$$C(x,y) = \begin{cases} 1 & \text{if } x < y \\ 0 & \text{otherwise.} \end{cases}$$

Finally, test symbol is labeled as,

$$L = argmin_c \sum_{i=1}^{n} \frac{min_t\left(\overline{D^t_{c,i}}\right)}{W_{c,i}}.$$

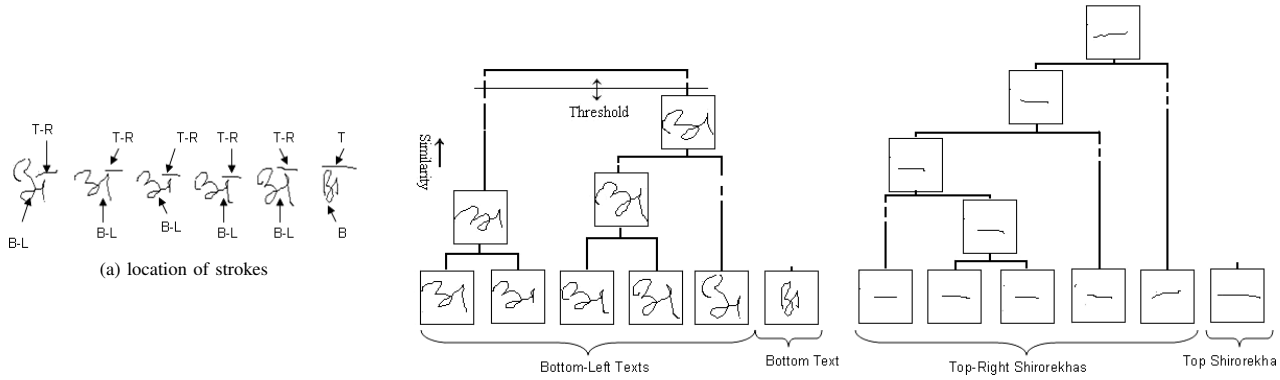
IV. Experiment

A. Dataset

The dataset was composed of 1800 symbols representing 36 characters, from 25 writers. Each writer was given the opportunity to write each character twice. 15 writers were used for training. The remaining 10 writers were for testing. No directions, constraints, or instructions were given to the users. Our dataset can be downloaded from the IAPR TC-11 website `http://www.iapr-tc11.org/`.

B. Experimental Results

We confronted our approach to the one expressed in [6] which does not take spatial information into account. Table I shows the experimental results for both training and test data sets. Our error rate was only 1% for training symbols, and 5% on the test symbols. Further, matching with structured learnt strokes reduces running time to 12 sec per character



(a) location of strokes

(b) pairwise merging based on location

Figure 5. Clustering for a two-stroke vowel अ

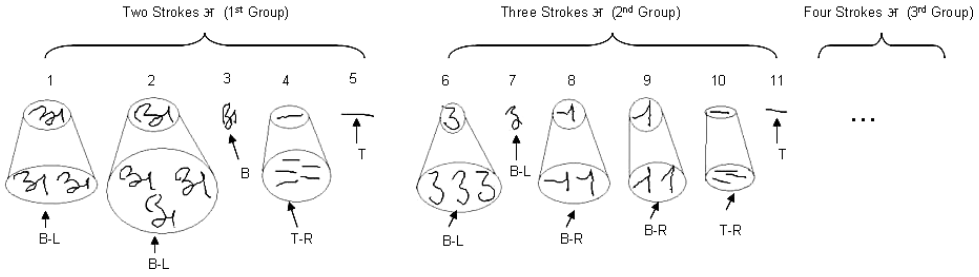


Figure 6. Learnt strokes – a sample for different groups based on the location

in average from 32. Besides the size of the strokes, time complexity is also depend on number of strokes used for a test symbol. In our database, most of the symbols consists of 2 – 3 strokes while, a few consists of 4 and 5 strokes and some times 6. Therefore, it took more time for test symbols with 2 – 3 strokes than 4, 5 and 6 strokes. We used MATLAB in Windows platform. Speed can therefore be improved considerably with the help of a high level language.

Table I
ERROR RATES FOR BOTH TRAINING AND TEST DATA

Dataset	Test Chars.	Mis-recognitions	Rejections	Average Error
Training	Consonants Vowels	15 | 9 4 | 1	4 | 2 3 | 1	2% | 1%
Test	Consonants Vowels	61 | 29 10 | 4	17 | 5 2 | 3	12.5% | 5%

Index: without spatial information [6] | with spatial Information

C. Experimental Error Analysis

We analyse frequent origins of errors such as structure similarity, tremor handwriting as well as re-writing *etc*.

Very interestingly, one of the major confusions due to *structure similarity* was significantly reduced by using spatial information. Few examples are still worth to be analysed. Most of the pairs mentioned in section I like (भ, म), (ध, घ), (थ, य) were correctly recognised. Fig. 7 is one of the examples to illustrate the importance of location of

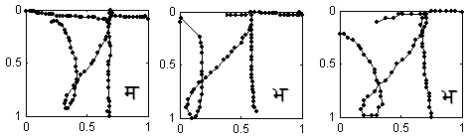


Figure 7. Recognised similar pair: म, भ

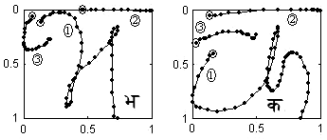


Figure 8. Symbols with re-writing stroke

shirorekha since the curve in the ascending part of the handwritten character भ is vague.

Tremor handwritings with very *short/long ascender* and *descender*, and *mis-writing* strokes were comparatively recognised. Besides, there is no improvement in the recognition of symbols with re-writing strokes.

Re-writing strokes are often the corrections made by users. In Fig. 8, the 3^{rd} stroke is referred to as rewriting stroke. Such a stroke is used to complete the structure of the character. *Re-writing stroke* individually, does not contain any information about the shape. But in off-line context, it gives visually a complete structure of the symbol while it is difficult to analyse on-line *re-writing strokes*. Therefore, the

recogniser was heavily influenced by those extra *re-writing strokes*. Overall, rejections of test symbols were usually due to re-writing and tremor (incomplete) handwritings.

V. Conclusions and Further Work

We have presented a novel concept of clustering based on stroke spatial relation for natural handwritten Devanagari characters. We have validated the efficacy of the clustering for writer independent strokes of any number and order. The proposed clustering approach is flexible and can be extended to any script. We plan to extensively test and experiment our approach on other datasets.

Further, we plan to use spatial relation model by extending it to 3×3 regions including *Middle* for syllable level Devanagari recognition including compound symbols. Compound symbols refer to more complex structure formed by the composition of two consonants. Besides, we work on stroke recovery such as merging of re-writing stroke with the closest one to make it meaningful. Also, we are interesting to use Inductive Logic Programming (ILP) as in [24], [25].

Acknowledgements

The work is partially based on Master Thesis: TC-MS-2006-01, conducted in Knowledge Information & Data Management Laboratory, School of ICT, SIIT, Thammasat University.

References

[1] S. D. Connell, R. M. K. Sinha, and A. K. Jain, "Recognition of unconstrained on-line devanagari characters," in *ICPR*, 2000, pp. 368–371.

[2] J. Hu, M. K. Brown, and W. Turin, "Hmm based on-line handwriting recognition," *IEEE PAMI*, vol. 18, pp. 1039–1045, 1996.

[3] M. Schenkel, I. Guyon, and D. Henderson, "On-line cursive script recognition using time delay neural networks and hidden markov models," *MVA*, vol. 8, no. 4, pp. 215–223, 1995.

[4] S. D. Connell and A. K. Jain, "Template-based online character recognition," *PR*, vol. 34, pp. 1–14, 1999.

[5] C. Bahlmann and H. Burkhardt, "The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping," *IEEE PAMI*, vol. 26, no. 3, pp. 299–310, 2004.

[6] Santosh K. C. and C. Nattee, "Stroke number and order free handwriting recognition for nepali," in *PRICAI*, 2006, pp. 990–994.

[7] N. Joshi, G. Sita, A. G. Ramakrishnan, V. Deepu, and S. Madhvanath, "Machine recognition of online handwritten devanagari characters," in *ICDAR*, 2005, pp. 1156–1160.

[8] A. M. Namboodiri and A. K. Jain, "Online handwritten script recognition," *IEEE PAMI*, vol. 26, no. 1, pp. 124–130, 2004.

[9] S. Malik and S. A. Khan, "Urdu online handwriting recognition," in *IEEE Sympo. on Emerging Techno.*, 2005, pp. 27–31.

[10] Z. Luo and C.-H. Wu, "A unit decomposition technique using fuzzy logic for real-time handwritten chinese character recognition," *IEEE Trans. on Industrial Electronics*, vol. 44, no. 6, pp. 840–847, 1999.

[11] S. Marukatat and T. Artieres, "Handling spatial information in on-line handwriting recognition," in *IWFHR*, 2004, pp. 14–19.

[12] F. Bouteruche, É. Anquetil, and N. Ragot, "Handwritten gesture recognition driven by the spatial context of strokes," in *ICDAR*, 2005, pp. 1221–1225.

[13] S. Hariharan, S. C. Chandra, and C. V. Srinivasa, "Spatiostructural features for recognition of online handwritten characters in devanagari and tamil scripts," in *ICANN, LNCS*, 2007, pp. 230–239.

[14] M. Blumenstein, B. Verma, and H. Basli, "A novel feature extraction technique for the recognition of segmented handwritten characters," in *ICDAR*, 2003, p. 137.

[15] B. Verma, J. Lu, M. Ghosh, and G. R., "A feature extraction technique for on-line handwriting recognition," in *IEEE IJCNN*, 2004, pp. 1337–1341.

[16] D. Okumura, S. Uchida, and H. Sakoe, "An hmm implementation for on-line handwriting recognition - based on pen-coordinate feature and pen-direction feature," in *ICDAR*, 2005, pp. 26–30.

[17] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," 1999.

[18] D. Papadias and T. Sellis, *Relation Based Representations for Spatial Knowledge*. National Technical Univ. of Athens: PhD Thesis, 1994.

[19] K. Miyajima and A. Ralescu, "Spatial organization in 2d segmented images: representation and recognition of primitive spatial relations," *Fuzzy Sets Syst.*, vol. 65, no. 2-3, pp. 225–236, 1994.

[20] X. Wang and J. M. Keller, "Human-based spatial relationship generalization through neural/fuzzy approaches," *Fuzzy Sets Syst.*, vol. 101, no. 1, pp. 5–20, 1999.

[21] M. Egenhofer and J. R. Herring, "Categorizing Binary Topological Relations Between Regions, Lines, and Points in Geographic Databases," in *Univ. of Maine, Research Report*, 1991.

[22] D. Mitra, "A class of star-algebras for point-based qualitative reasoning in two-dimensional space," in *Intl. Florida AI Research Soc. Conf.*, 2002, pp. 486–491.

[23] E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping to massive dataset," in *European PKDD*, 1999, pp. 1–11.

[24] A. Amin, "Prototyping structural description using an inductive learning program," *Intl. Journ. of Intelligent Sys.*, vol. 15, no. 12, pp. 1103–1123, 2000.

[25] Santosh K. C, B. Lamiroy, and J.-P. Ropers, "Inductive logic programming for symbol recognition," in *ICDAR*, 2009, pp. 1330–1334.