Devanagari Character Recognition Using Structure Analysis

Krishnamachari Jayanthi†, Akihiro Suzuki†, Hiroshi Kanai†, Yoshiyuki Kawazoe‡, Masayuki Kimura†, and Ken'iti Kido*

†Department of Information Engineering, Faculty of Engineering
‡Education Center for Information Processing
and
*Research Center for Applied Information Science
Tohoku University, 980 Sendai, Japan

A method of character recognition using prior knowledge of the script has been proposed. Devanagari, a script widely used in India at present, and found in Buddhist texts of the past, has been taken for this purpose. This study has been confined to recognizing a particular font used in a printed Buddhist text; Saddharmapundarika.

INTRODUCTION

Devanagari is the character set used in Sanskrit, Hindi and Marathi, the latter two languages being included among the official languages of India. Sanskrit is a language used by the ancient scholars in their literary compositions.

The following is the specific details of the Devanagari alphabet: 14 vowels, 33 consonants, 10 numerals and 3 special characters. Including the vowel-consonant (v-c) and consonant-consonant (c-c) combinations, the number increases enormously: the v-c combinations being 462 (14 vowels X 33 consonants) ; the c-c combinations can be formed by adding any number of consonants in any order, but in practice, the number of c-c combinations are quite limited and may not exceed fifty.

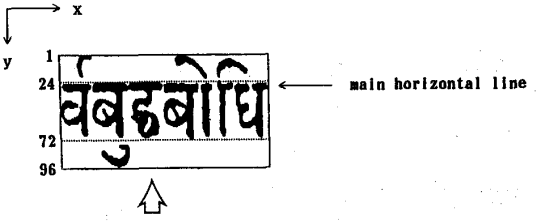
Many of the original texts of Buddhist literature are in Devanagari, in the language of Sanskrit (exactly speaking, Buddhist hybrid-Sanskrit). Given the widespread interest in the study of original Buddhist texts, and the fact that it is quite unrealistic to expect those engaged in Buddhist literature studies at present to learn an entire new script for the purpose of their research, the task of converting texts written in Devanagari into Roman script using computer system becomes necessary. Further, printed data, due to the aging of the paper, is always subject to deterioration, and the cost and effort of renewal is also high; thus the text stored in a computer database would be more easily preserved.

Sinha and Mahabala [1] once tried to recognize Devanagari automatically according to their pattern analysis system. They chose 26 symbols and extracted structual information for these characters. However, their experiment was limited in sample size, and could not give quantitative recognition rate. Although the present study has been developed independently of the result by Sinha and Mahabala, it conceptually is an extension of their work. We have employed a more sophisticated thinning algorithm, a large set of characters, a more computer suitable feature extraction method, and an exhaustive experimental recognition test, aiming to achieve a practical level of automatical Devanagari recognition. We are also studying pattern matching method to recognize printed Devanagari script, and the result will be published elsewhere[2].

DATA COLLECTION AND PROBLEMS

The present study has been performed on IBM3081-KX6 of Education Center for Information Processing, Tohoku University. The database consists of whole lines of text, and the coordinates of extracted characters for each line of text. The characters stored are in the form of a 144 by 96 matrix.

The extraction has been performed by the standard histogram method. The lower part of Fig. 1 shows the first page of the text Saddharmapundarika [3], printed in Devanagari. As is shown in Fig. 1, due to the nature of the script, each line has been divided into three longitudinal sections, the first being from pixel 1 to 24 on the y axis, the second, which is the main part, from 25 to 72, and the third from 73 to 96.



ओं नमः सर्वबुद्धबोधिसत्त्वेभ्यः । नमः सर्वतथागतप्रत्येकबुद्धार्यश्र
तानागतप्रत्युत्पन्नेभ्यश्च बोधिसत्त्वेभ्यः ॥

वैपुल्यसूत्रराजं परमार्थनयावतारनिर्देशम् ।
सद्धर्मपुण्डरीकं सत्त्वाय महापथं वक्ष्ये ॥

Fig. 1. Devanagari sample text in Saddharmapundarika. The enlarged upper part shows the present resolution and the coordinate system used.

CH2766 – 4/89/0000 - 0363 © 1989 IEEE

It is clear from Fig. 1 that most characters have a heavy horizontal line somewhere near the top, and likewise, end a few pixels above the bottom. This region is referred to as the main horizontal line (MHL). There are several characters which do not have the MHL, as also a few characters which extend below the usual lower limit. Generally, the separate features above the MHL or below the lower limit are vowel additions to the main character, and are handled separately.

We have checked the first 50 pages of the text, and found that there are 35 basic characters; 6 vowels and 29 consonants. Additionally there are 4 special characters, 10 numerals, and here we include 18 frequently occurring c-c combinations for the recognition procedure. The characters considered in the present papaer are given in Fig. 2. The vowel features to be recognised are 10. Hence the total number of characters to be recognised comes to about 100, with 10 vowel features extra.

Fig. 2. 67 Basic Devanagari characters for recognition with pronounciations.

THINNING ALGORITHM

To thin the characters to be recognized, we have used the algorithm developed by Holt et al. [4]. First, this algorithm was applied without any changes to the characters, and the result studied.

Two major features of Devanagari characters are the main horizontal line and various vertical lines. Since it is obvious that neither feature has been preserved by the Holt algorithm, it was decided to modify it in order to make use of it for the special case of Devanagari characters. In Devanagari characters MHL and vertical lines, which have almost no information, occupy large portions of character images and should be separated out for recognition. The changes made to the above algorithm are mentioned below.

a. The pre-processing stage : Before the actual thinning algorithm is applied, the character is examined for the presence of main horizontal line and vertical lines. This is done by computing the histogram along with the vertical and horizontal axes, and checking for peaks. For every character, only one main horizontal line, and up to three vertical lines, are allowed for; this is adequate, as no valid character has more than this.

b. Modifications in the algorithm: During the thinning stage, the positions already stored in the character array are used to skip the thinning checks for these coordinates in the character, i.e., for the thinned main horizontal line and vertical lines, thus these are maintained as they are. Fig. 3 shows the modified procedure; the left half of the figure shows the intermediate stage where only the MHL and a vertical line have been extracted, and the right half the last stage after the application of the Holt thinning algorithm to the rest of the character image except for the extracted MHL and VL.

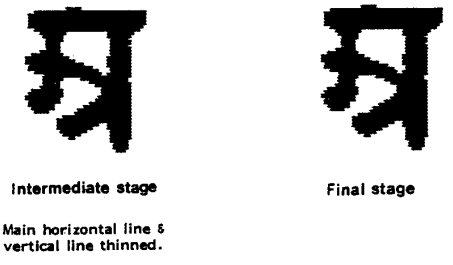


Fig. 3. Result of the modified thinning algorithm. The left part indicates the intermediate stage, and the final result in the right part.

The extraction of MHL and vertical lines before the application of thinning algorithm is necessary to preserve the structural information of the important part of Devanagari characters. It might also be possible to extract the MHL and vertical lines after application of the Holt thinning algorithm to the whole character image. Although this method is more general than the present one, the reason why we have selected the latter is to use explicitly the characteristic features of the Devanagari script and reduce the complexity of the analysis.

IDENTIFICATION OF IMPORTANT FEATURES

As Devanagari characters have widely varying sizes, unlike printed Kanji which can be confined in uniform squares, most of the known methods [5] had to be rejected as being unsuitable for Devanagari. The frequently used Devanagari characters in the text taken for study are about 100. With this small number, it is easy to locate the outstanding features and classify the characters by a binary tree based on these features.

A further reason for the choice is that this method of recognising characters by their features can be extended to other fonts apart from the type in the text under study, and subsequently to hand-written characters, too. Especially, the last advantage is remarkable compared to the pattern matching method.

The features chosen for recognition fall into two categories; (1) features which are outstanding and easy to recognise in a program; and (2) features which divide the characters into equal classes making the binary tree more balanced and more efficient. As far as possible, both the factors were kept in mind, though the latter factor was given more importance in the earlier stages of the division procedure and in the last few steps, ease of programming was taken into consideration. It was also felt at the later stages that it is better to use the features as they occur, instead of trying to force them into an artificial binary format. Hence at later stages in the classification tree, the structure is not strictly binary; branching into 3 or more leaves at node.

A cursory glance at the characters is enough to find out that the most outstanding feature of Devanagari characters is the MHL. Though it may seem at first that all the characters have it, a closer inspection will reveal that this is not so, though the majority of characters do have the MHL. Accordingly, this was chosen as the first feature to be identified. The second feature has been the presence or absence of vertical lines, with the third (in case vertical lines are present) being whether the line is the rightmost feature in the character. The other features have been the height/width ratio of the character, whether the character is narrow or broad ended, the number of free ends it has etc.. Immense care had to be taken in the case of certain features, for example, the number and position of free ends, to take care of frequently occurring printing defects like breaks.

RECOGNITION USING BINARY TREE

The above mentioned features for recognition mostly fall into a binary pattern; even the other features which do not necessary fall into a binary pattern either can be made to do so, or else have a numerical basis for decision, which is easy to program. Further, a binary tree is one of the fastest decision making processes for a computer program. With these reasons in view, it was decided to adopt a top-down binary tree for character recognition. The parameters for decision making were decided on a trial and error basis, with the view of accommodating the maximum number of characters; though this has led to a few errors, the results are generally satisfactory. As there are 67 characters for recognition, to give the full binary tree is an unwieldy exercise; accordingly, the tree has been traced up to the first three levels, with the number of characters in each branch, and following the major branch only.

At later levels, the structure of the tree is not strictly binary. To give an example of a non-binary level, if we take the number of intersections with the main horizontal line, the number of branches is three most naturally. All the 67 characters are classified according to this method, and are the last levels in the tree. Figure 4 shows the flow chart of the present recognition procedure.

By looking at the program, we can calculate the average number of levels, and the levels for the worst cases. The average turns out to be six, with the worst case examples at eight. According to the clear characteristic features extracted, the number of decisions to obtain the final candidate is only 6 in average. This proves the effectiveness of the present binary tree method for the recognition of the Devanagari script.

EXPERIMENTAL RESULTS

The effectiveness of the present method was evaluated by recognition experiments taking the first ten pages of the text as sample. The total number of extracted characters is 4863, including those not properly extracted.

In Table 1, the errors have been divided into five categories, and the final table gives the error percentage excluding the first two category errors. This is due to the fact that the errors from noise and breaks can be totally attributed to the printing defects in the text and are completely independent of the method used for recognition. Though the confusion between the two characters 'pa' and 'ya' is also mainly due to the text, with very stringent checks, these two could possibly have been told apart, and is hence included in the errors of the proposed method.

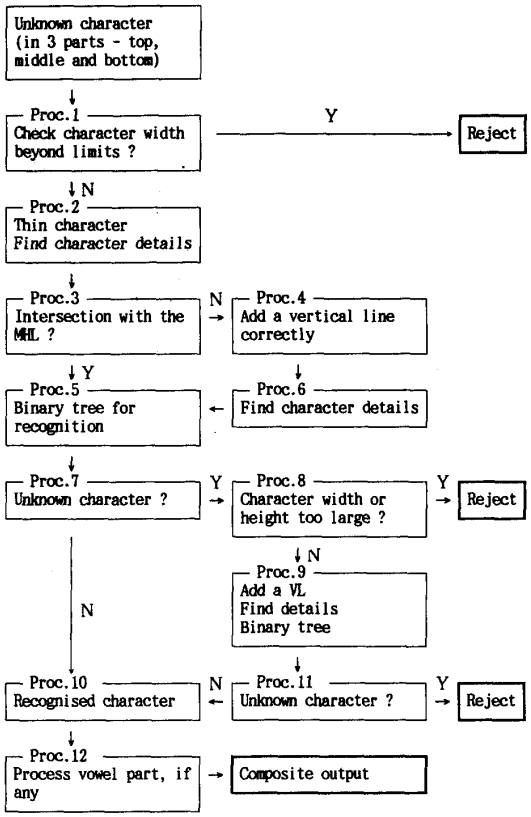


Fig. 4. Program flowchart for Devanagari script recognition by structure analysis. Reject procedures for unrecognizable characters are also shown.

CONCLUSION

The structure analysis of Devanagari characters has had a promising beginning.

Although the present study has concentrated on the font of a particular text, the same methodology can be applied to other fonts. Especially, the more recent ones, and with the current state of printing technology, the results for these are bound to be much better. We have started the next step to extend this method to hand-written Devanagari characters in ancient Buddhist manuscripts.

ACKNOWLEDGEMENTS

The authors are grateful to Professor K. Tsukamoto, Tohoku University and Asst. Professor M. Yamazaki, Sendai Technical College for their help in reading the Buddhist text.

REFERENCES

[1] Sinha R.M.K. and Mahabala H.N. : "Machine Recognition of Devanagari Script", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-9, No.8, pp.435-441 (1979).

[2] Suzuki A., Kanai H., S. Makino, Kawazoe Y., and Kido K.: "Devanagari Character Recognition Method by Using Extraction and Recognition Procedures Simultaneously", Journal of IEICE Japan, J72-D-II(1989).

[3] "Saddharmapundarika" : ed. H.Kern and Bunyiu Nanjio. St. Petersbourg, Imprimerie de L'Academie Imperiale des Sciences (1912).

[4] Holt C.M., Stewart A., Clint M. and Perrott R.H. : "An Improved Parallel Thinning Algorithm", Commun. of the ACM, Vol.30, No.2, pp. 156-160 (1987).

[5] Mori S., Yamamoto K. and Yasuda M. : "Research on Machine Recognition of Handprinted Characters", IEEE trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.4 (1984).

Table 1. Error percentages before error correction

Page no.	% error (total)	No.of errors	Categories of error				
			Noise	Break	Thinning	pa/ya	Others
1	1.42	5	1	2	2	0	0
2	1.83	10	3	3	1	2	1
3	1.88	8	4	0	1	1	2
4	3.36	22	6	9	1	2	4
5	3.56	17	5	7	0	1	4
6	4.44	24	5	7	7	2	3
7	7.98	48	3	20	9	3	13
8	7.34	31	6	7	9	3	6
9	11.73	48	6	23	12	1	6
10	5.68	25	3	12	2	3	5
Total	4.92	238	42	90	44	18	44