ACCV2002: The 5th Asian Conference on Computer Vision, 23--25 January 2002, Melbourne, Australia.

Recognition of Printed Sinhala Characters Using Linear Symmetry

H. L. Premaratne
School of Information Science, Computer and Electrical Engineering
Halmstad University, S-301 18 Halmstad, Sweden
Lalith.Premaratne@ide.hh.se

J.Bigun
School of Information Science, Computer and Electrical Engineering
Halmstad University, S-301 18 Halmstad, Sweden
Josef.Bigun@ide.hh.se

Abstract

Sinhala characters used in the Sinhala script by over 70% of the 18 million population in Sri Lanka, have been descended from the ancient Brahmi script. The Sinhala alphabet consists of vowels and consonants and the consonants are modified using modifier symbols to give the required vocal sounds. In the process of developing an OCR for the Sinhala script, characters are initially recognised through a multi-level filtering process using the Linear Symmetry [LS] feature [1]. The recognised character is then segmented to identify the associated modifier symbol/s. Since the use of LS recognises characters prior to segmentation, the most difficult task of separating touching characters is easily solved. A method to determine the skew angle of the script is also presented. Experiments conducted so far for widely used fonts of different sizes yield encouraging results.

1. Introduction

1.1 Alphabet and the Modification Processh

The objective of this research is to develop an Optical Character Reader (OCR) for the Sinhala script. The Sinhala script used by over 70% of the 18 million population in Sri Lanka has been descended from the ancient Brahmi script and evolved independently over many centuries. The Sinhala language is unique to Sri Lanka and the Sinhala characters that are generally round in shape differ from all the other Brahmi descended scripts in South Asia. The Sinhala alphabet consists of 18 vowels, 41 consonants and 17 modifier symbols. A vowel may appear only as the first character of a word and a consonant is modified using one or more of the modifier symbols to produce the required vocal sound. The total number of different modifications from the entire alphabet including the basic characters is nearly 400. Although each character possesses a distinct characteristic shape to distinguish from the others, some characters resemble with one or more of the other characters by their appearance. Some examples are given in Figure 1.

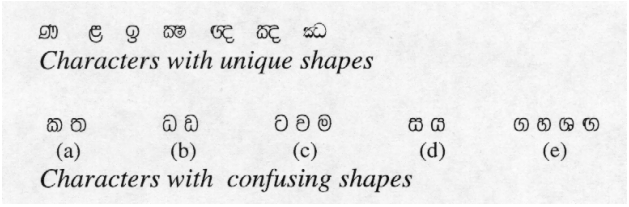


Figure 1. Distinct and similar shapes of characters

Modification of a character is carried out by simply adding one or more modifier symbols before/after/above/below the character without affecting its general shape.. However this rule is violated for a specific subset of the alphabet numbering to 10 characters, in most of the printed scripts, to give a better appearance (Figure 2). Also, in some modifications, the joint between the character and the modifier symbol is smoothed to make the modified character appear as a single unit of symbol.

Figure 2. Violation of modification rule by changing the shape of a character

1.2 Characteristics of the Script

A single line of script is organised in three horizontal layers. The middle layer contributing to approximately 50% of the total line height, mainly include fifteen (15) basic characters and Nine (9) modifier symbols. Twenty two (22) other basic characters occupy the middle layer and the upper layer, with approximately 75% and 25% of the total height of each character in each layer respectively. The middle and the lower layers include the remaining eight (8) characters, with approximately 75% and 25% of the total height of each character in each layer respectively. Four (4) modifiers occupy the upper layer while the remaining five (5) modifiers are assigned to the lower layer. The upper and the lower layers are of equal height each having 25% of the total line height. (Figure 3).

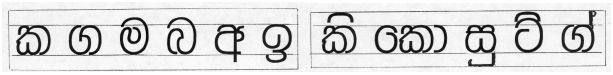


Figure 3. Three-layered structure of a line

1.3 Using Linear Symmetry as the Main Feature for Recognition.

Linear Symmetry [1] which is used as a feature of a character is the major contributor to the recognition process. Further, it is the major contributor in distinguishing confusing characters. A conventional method such as the use of an artificial neural network could still be used on top of the proposed system if the necessity for further improvement of accuracy arises.

A typical Sinhala Script may constitute of modified or unmodified characters from nearly 400 different symbols. Since it would be very difficult to handle such a large number of different classes in a neural network, segmentation of modified characters to their basic level is necessary. As explained in 1.2 above, in most of the scripts, the general shape of 10 basic characters are changed in two different ways in modification, adding 20 more different symbols to the alphabet. Therefore, the least number of classes that could be used in a neural network is still as high as 80. This reduced number of classes is achieved at the cost of a difficult task of segmentation. But, in the proposed method, each character in the script is identified prior to its segmentation. Even the 20 different modified versions of the 10 characters mentioned above are recognised through the basic character at a rate of 95%.

Most importantly, as explained in 4.1, the proposed recognition method of using linear symmetry solves the complex problem of separating vertically touching characters in the segmentation process. The proposed grouping process explained in 4.2 does not require segmentation to separate modifier symbols from the basic character, as the modifier symbols are identified within the modified character. This is made possible only if the basic character in its modified version is known prior to segmentation.

2. Determination of Skew Angle

The horizontal projection of the script shows that one of the horizontal boundaries of the middle layer carries the highest number of pixels. The reason for this characteristic could be observed by examining a line of script. Therefore, detecting this boundary and its direction could be used to determine the skew angle and thereby to align the script.

The proposed method which consists of the following three steps, effectively uses the Linear Symmetry feature (explained in 3.1 below) in determining the skew angle as it can directly deal with the orientation of text lines.

i. Construct the LS tensor of the text image.
ii. Build the level 1 of the gaussian pyramid of i above.
iii. Transform the image in ii above to frequency domain.

The resulting image in the frequency domain provides a highly prominent line corresponding to the orientation of the script lines, in the orthogonal direction.

This method was tested for various skewed scripts of different fonts, ranging from 0-90 degrees. The tests provide 98-100% accurate results.

3. Recognition Process

3.1 Theory

The fundamental theory used in the recognition process is the Linear Symmetry [1] which has been used effectively in many applications over the past few years. The Linear Symmetry [LS] is characterised by the fact that it delivers a dense orientation field along with certainties. In case of high confidence on existence of orientation, the linear orientation represents the least change of gray values in one direction and maximal change in the orthogonal direction. Hence, a Linear Symmetry tensor for an image is constructed by averaging the orientation of a local neighbourhood, for each pixel of the image.

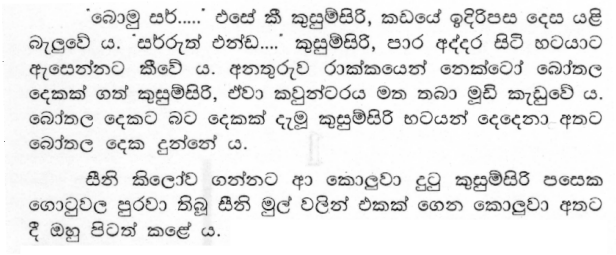


Figure 4. Original image

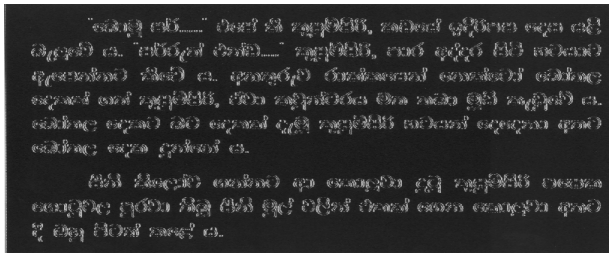


Figure 5. Linear Symmetry tensor

The LS tensor is built as follows

Four 1x4 derivative filters dx, dy and gx, gy are generated. The two derivative convolutions dxf and dyf of the original image with respect to x and y are constructed using the above pair of filters.

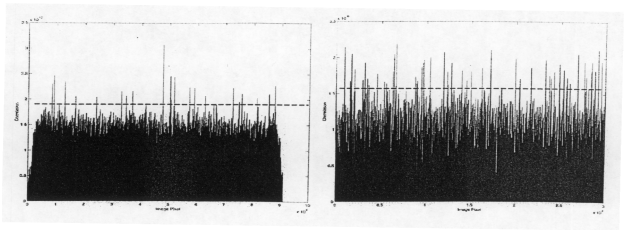
The LS tensor is then given by

LS=(dxf+j*dyf).^2 where j =√(-1)

To the best of our knowledge, there have been no significant research done on the recognition of Sinhala characters and this is the first instance that the Linear Symmetry is used for character recognition.

3.2 Recognition Procedure

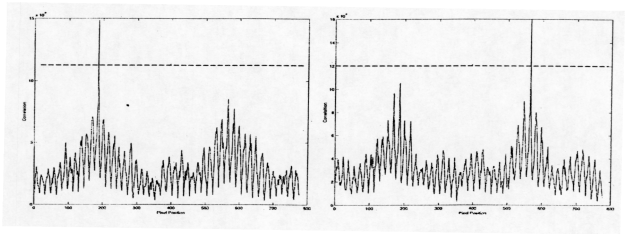
3.2.1 Testing Database. The recognition process is based on the examination of the correlation of characters in the script with each character of the alphabet through a filtering operation. The testing alphabet which consists of all the characters, is built by extracting characters from an LS tensor. Each character in the testing alphabet is filtered (one at a time) through the LS tensor of the script in order to identify its occurrences in the entire script. The plot of correlation at each pixel (Fig. 6) shows that, each occurrence of the character being tested gives a strong correlation. A suitable threshold that separates the required character from the rest of the characters in the script, is then determined. This procedure is conducted for each and every character of the alphabet. During this process, it has been observed that a total number of 35 characters amounting to 60% of the alphabet separates from all the other characters with a clear threshold (Fig. 6(a)) while the balance 40% confuse with one or more characters with similar shapes (Fig. 6(b)). Eight (8) such confusing groups have been identified.



(a) A character with a unique shape (b) A character confusing with similar shapes
Figure 6. Correlation of a character with the script

Once all the different confusing groups are identified, another level of filtering is carried out to separate each character within the confusing group. The secondary level of filtering is performed to examine the correlation of a distinct segment from one character with all the members in the group (Fig. 7). A suitable (secondary) threshold that separates each character from the rest is then determined. A further level of filtering is carried out if the confusion still occurs.

The structure of the testing database is as follows.
Character Identifier
LS Tensor of character
Primary Threshold
Flag to indicate confusing status
Secondary Threshold (for confusing characters)
Tertiary Threshold (for confusing characters)



(a) Segment of character 1 (b) Segment of character 2
Figure 7. Correlation of a distinct segment from one character with two similar characters

3.2.2 Recognition. The image is initially pre-processed to remove the background noise. A horizontal projection is carried out to determine the average line height. In this process some additional information such as height and mid point for each line is recorded (to be used in grouping at a later stage). The image is then scaled (if necessary) to match the average height of a character to that of the testing alphabet.

Recognition of a script is performed by filtering the LS tensor of each character of the testing alphabet with the LS tensor of the script. In each filtering cycle, all the occurrences of the character being tested are identified. If the testing character is a confusing one, the secondary level of filtering is carried out in order to determine the acceptance or rejection of the identified character. A tertiary level of filtering is carried out similarly.

It has been observed that, in addition to the highest value of correlation produced usually at the centre of the character, a few more relatively high values are also produced around the neighbouring pixels. This is due to the fact that the template of the testing character nearly coincides with the neighbouring pixels around its centre. This will result in recognising the same character in the image more than once. Therefore, once the filtering has been performed, non-maximums in a small neighbourhood (e.g. 3x3) are suppressed in order to eliminate the multiple acceptance of the same character.

Each Accepted character is then segmented (4.1) and grouped (4.2) and the Character Identifier, Spatial Co-ordinates and the Group are recorded. The direct output, which is automatically arranged on the order of the character identifier of the alphabet, is then sorted primarily on the row number and then on the column number.

The recognition algorithm is as follows:
Input image
*Input database-of-characters */Alphabet/**
Pre-process image
Perform Horizontal-projection
Extract Line-data
ConstructLS-tensor
Read character
While not-end-of-alphabet do
Filter characte with the LS Tensor

/ Primary Filtering /
Supress non-maximums
While not-end-of-image do
Segment occurrences above threshold
If confusing-charcater
Determine relative rhreshold
Perform secondary-filtering
/ and tertiary-filtering if necessary*/*
End-If
Determine group
Store image-coordinates and group of -each occurrence
*End-While *** not-end-of-image ****
Update output array
/ with character-id, row, column no, group.*/*
Read character
*End-While *** not-end-of-alphabet****

4. Segmentation of Script and Determination of Group

4.1 Segmentation

The main advantage in handling complex steps of segmentation is the use of Linear Symmetry for the recognition. Since a basic character is directly recognised between its modifiers, the segmentation could be initiated from the pixel at which the character has been recognised. It is a common feature in the script that two adjacent characters touch each other due to the existing noise or the occurrence of a specific modifier symbol. Two characters may touch each other with direct or indirect contact making it difficult to determine a suitable threshold for separation. As the character has already been recognised, separation could be done by using the width (and if necessary the height) of the character. Segmentation is required only to determine the modified status (i.e. the real appearance) of the recognised character. Therefore, in most of the cases, the segmentation could be limited to determine the segmented groups containing either basic characters or different modified versions of characters. Hence the segmentation of a modified character into its primary components (i.e. its basic form and the associated modifier symbols) is not essential.

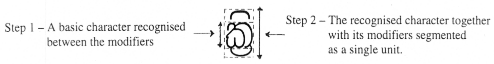


Figure 8. Recognition of a character and the consequent segmentation

Once a character is recognised at a pixel position, it is isolated from the original image into a rectangular box. In this step, by using the length and the width of the character, it is ensured that the complete character (in its modified or unmodified state) is included inside the extracted box (Fig. 9). A consequent vertical projection isolates the required character.

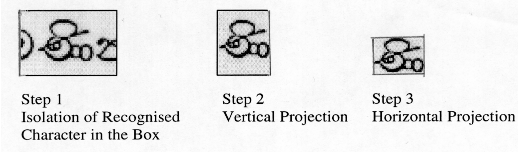


Figure 9. Segmentation of a non-touching character

If the character is being touched by another character vertically on the left or right to it, either whole or a fragment of the touching character may also be included in the isolated segment. The width of the character and the exact pixel co-ordinate in which it has been recognised are now used to separate the required character from its touching fragment (Fig.10).

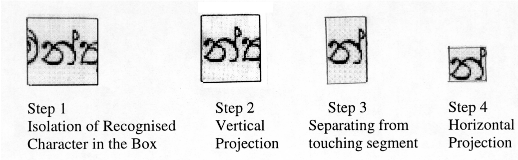


Figure 10. Segmentation of a touching character

This method has produced nearly 100% accuracy in segmenting vertically touching characters. In the final stage of segmentation, a horizontal projection is carried out in order to segment the character completely.

4.2 Grouping

The final stage is the identification of components in the segment. Majority of characters posses a unique association with specific modifier symbols and this characteristic could effectively be used in grouping.

The parameters used in the grouping process are the width and the height of the recognised character, the height and the mid point [2] of the line in which the recognised character is located. The segment is classified into four broad groups initially and narrowed down to 10 subgroups subsequently, by (virtually) positioning the segment in its original line.

Table 1. Four basic groups

Group	Layer/s Occupied by the Character	Example
A	Middle	ස, ග, ක, ප, ර
B	Upper/Middle	සි, ති, එ, ශ්
C	Middle/Lower	සු, කු, තු, ල
D	Upper/Middle/Lower	ශ්‍රී

The group A in Table 1 contains only the basic characters and modifier symbols. Therefore no further segmentation of group A will be required. The group B and C may contain basic characters, modified characters and modifier symbols. All the basic characters except two, in group B could be separated using the general rule that a basic character should touch neither the upper most nor

the lower most boundary. In certain texts, it is often noticed that the two characters mentioned above violate this rule.

Once the basic characters are identified and removed from the group B, the remaining characters are tested for five different conditions to identify different modified versions. The modified characters violating the basic modification rule (as explained in 1.1 above) need to be identified and treated accordingly within this group. It is highly effective that each such character is assigned the status of a basic character.

The group C is separated into three subgroups in a similar manner, at the next level. Each sub group is then processed to identify the basic character and the associated modifier symbol using appropriate conditions.

The group D is divided into three sub groups using two conditions. The first being the relative height of the character and the second being the relative mid point [2] of the segment and that of the line in the vertical direction.

The final result of grouping will produce one of the following.

i. a basic character or a modifier symbol (lying in the horizontal line)
ii. a basic character and its associated modifier symbols composing a unique modified version

5 Experimental Results

Results of the recognition of individual characters during the initial stage show that while the characters with unique distinct shapes consist of 60% of the alphabet are recognised between 93 to 100 percent, the confusing characters are recognised at an average rate of 76%. 96% of the confusing characters are distinctly recognised at the secondary level of filtering. The experiments conducted for around thirty images of the same fonts and size carrying 600 to 1200 characters per image yield approximately 92% accurate results. Nearly forty images carrying widely used fonts with varying sizes contributed to as high as 84% accurate results. It has been observed that the variation of size is tolerated within a 5-10% margin. The quality of images varied from more noise to less noise and some of the images were captured from two year old newspapers of low quality prints.

6. Conclusions and Future Work

The results of the investigations carried out so far provide a highly effective method for the recognition of Sinhala characters. Since the Sinhala language and the script have been evolved more independently in the island of Sri Lanka, without a close connection to other South Asian languages in the Indian sub-continent, some of its characteristic features need to be handled distinctly. It has been shown that the use of linear symmetry of a single character as the principal features in the recognition process is highly effective. The testing alphabet constructed from an LS tensor of the same font and the size performs fairly well for slightly different fonts. Construction of a more stable database of the alphabet, and the methods to generate dynamic relative threshold for separation will be investigated in the next stage. Application of features characteristic to the Sinhala script (e.g. a vowel is used only as the first character of a word) and the statistical methods will also be considered. As a small percentage of false rejections and false acceptance occur due to the distortion of characters in the source document, the possibility of word-level recognition using the on-line dictionary facility, will be explored. The use of conventional recognition techniques such as ANNs to supplement the improvement of accuracy, will also be considered. The research project will be continued to develop a comprehensive system covering a variety of fonts and different sizes to recognise the Sinhala script which will be useful in developing an OCR for the Sinhala script.

Acknowledgement: Support of SIDA (Swedish International Agency for Development Co-operation) is gratefully acknowledged.

References

[1] J.Bigun and G.H.Granlund, Optimal Orientation Detection of Linear Symmetry. ICCV'87, London 1987, pp 433-438, IEEE Computer Society Press, Los Alamitos, 1987.

[2] Nucharee Premchaiswadi, Wichian Premchaiswadi and Seinosuhe Narita, Segmentation of Horizontal and Vertical Touching Thai Characters. pp 987-995, ITC-CSCC'99.

[3] .S.M.S.Rajasekaran and B.L.Dekshatulu, Recognition of Printed Telugu Characters. Computer Graphics Image Processing 6, 1977

[4] G.Siromoney, R.Chandrasekaran and M. Chand-rasekaran. Machine Recognition of Printed Tamil Characters, Pattern recognition 10, 1978.

[5] A.K.Dutta, A Generalised Formal Approach for Description and Analysis for Major Indian Scripts. J.IETE 30, 1984

[6] P.J.Burt and E.H.Adelson, The Laplacian Pyramid as a Compact Image Code. IEE Trans. COMM, 31:532-540, 1983.

[7] G.S. Lehal and Chandan Singh, A Gurmuki Script Recognition System, pages 557-560, ICDAR'00.

[8] G.S. Lehal and Chandan Singh, A Gurmuki Script Recognition System, pp 557-560, ICDAR'00.

[9] Scott D. Connel , R.M.K. Sinha and Anil K Jain, Recognition of Unconstrained Devanagari Characters, pp 368-371, ICDAR'00.

[10] Yi-Kai Chen, Jhing-Fa Wang, Skew Detection and Reconstruction Based on Maximisation of Variance of Transition-Counts, pp 195-208,Pattern Recognition, 33 (2000).