Segmentation of Handwritten Text in Gurmukhi Script

Rajiv K. Sharma rajiv.patiala@gmail.com
Sr. Lecturer, SMCA
Thapar University,
Patiala, 147002
Punjab, India

Dr. Amardeep Singh amardeep_dhiman@yahoo.com
Reader, UCoE,
Punjabi University,
Patiala, 147004
Punjab, India

Abstract

Character segmentation is an important preprocessing step for text recognition. The size and shape of characters generally play an important role in the process of segmentation. But for any optical character recognition (OCR) system, the presence of touching characters in textual as well handwritten documents further decreases correct segmentation as well as recognition rate drastically. Because one can not control the size and shape of characters in handwritten documents so the segmentation process for the handwritten document is too difficult. We tried to segment handwritten text by proposing some algorithms, which were implemented and have shown encouraging results. Algorithms have been proposed to segment the touching characters. These algorithms have shown a reasonable improvement in segmenting the touching handwritten characters in Gurmukhi script.

Keywords: Character Segmentation, Middle Zone, Upper Zone, Lower Zone, Touching Characters. Handwritten, OCR

1. INTRODUCTION

In optical character recognition (OCR), a perfect segmentation of characters is required before individual characters are recognized. Therefore segmentation techniques are to apply to word images before actually putting those images to reorganization process. The simplest way to segment the characters is to use inter – character gap as a segmentation point[1]. However, this technique results in partial failure if the text to be segmented contains touching characters. The situation becomes grim if text consists of handwritten characters. The motivation behind this paper is that to find out a reasonable solution to segment handwritten touching characters in Gurmukhi script. Gurmukhi script is one of the popular scripts used to write Punjabi language which is one of popular spoken language of northern India. Because our work is related with segmentation of Gurumukhi script, so it is better to discuss some characteristics of the said script so that the reader can have a better idea of the work.

2. CHARACTERISTICS OF GURMUKHI SCRIPT

Gurmukhi script alphabet consists of 41 consonants and 12 vowels[2] as shown in FIGURE 2. Besides these, some characters in the form of half characters are present in the feet of characters. Writing style is from left to right. In Gurmukhi, There is no concept of upper or lowercase characters. A line of Gurmukhi script can be partitioned into three horizontal zones namely, upper zone, middle zone and lower zone. Consonants are generally present in the middle zone. These zones are shown in FIGURE 1. The upper and lower zones may contain parts of vowel modifiers and diacritical markers.

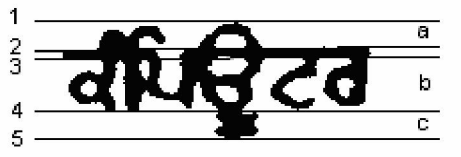

FIGURE 1 : a) Upper zone from line number 1 to 2, b) Middle Zone from line number 3 to 4, c) lower zone from line number 4 to 5

In Gurmukhi Script, most of the characters, as shown in FIGURE 2, contain a horizontal line at the upper of the middle zone. This line is called the headline. The characters in a word are connected through the headline along with some symbols as i, I, A etc. The headline helps in the recognition of script line positions and character segmentation. The segmentation problem for Gurmukhi script is entirely different from scripts of other common languages such as English, Chinese, and Urdu[3] etc. In Roman script, windows enclosing each character composing a word do not share the same pixel values in horizontal direction. But in Gurmukhi script, as shown in FIGURE 1, two or more characters/symbols of same word may share the same pixel values in horizontal direction.

This adds to the complication of segmentation problem in Gurmukhi script. Because of these differences in the physical structure of Gurmukhi characters from those of Roman, Chinese,

Consonants (Vianjans)

ੳ ਊੜਾ (ūṛā) u, ū, o	ਅ ਐੜਾ (aiṛā) a, ā, ai, au	ੲ ਈੜੀ (īṛī) i, ī, e	ਸ ਸੱਸਾ (sas'sā) sa [sə]	ਹ ਹਾਹਾ (hāhā) ha [hə]
ਕ ਕੱਕਾ (kakkā) ka [kə]	ਖ ਖੱਖਾ (khakhkhā) kha [kʰə]	ਗ ਗੱਗਾ (gaggā) ga [gə]	ਘ ਘੱਗਾ (ghaggā) gha [gə]	ਙ ਙੰਙਾ (ṅaṅṅā) ṅa [ŋə]
ਚ ਚੱਚਾ (caccā) ca [ʧə]	ਛ ਛੱਛਾ (chachchā) cha [ʧʰə]	ਜ ਜੱਜਾ (jajjā) ja [ʤə]	ਝ ਝੱਜਾ (jhajjā) jha [ʤə]	ਞ ਞੰਞਾ (ñaññā) ña [ɲə]
ਟ ਟੈਂਕਾ (ṭaiṅkā) ṭa [ʈə]	ਠ ਠੱਠਾ (ṭhaṭhṭhā) ṭha [ʈʰə]	ਡ ਡੱਡਾ (ḍaḍḍā) ḍa [ɖə]	ਢ ਢੱਡਾ (ḍhaḍḍā) ḍha [ɖə]	ਣ ਣਾਣਾ (ṇāṇā) ṇa [ɳə]
ਤ ਤੱਤਾ (tattā) ta [tə]	ਥ ਥੱਥਾ (thaththā) tha [tʰə]	ਦ ਦੱਦਾ (daddā) da [də]	ਧ ਧੱਦਾ (dhaddā) dha [də]	ਨ ਨੱਨਾ (nannā) na [nə]
ਪ ਪੱਪਾ (pappā) pa [pə]	ਫ ਫੱਫਾ (phaphphā) pha [pʰə]	ਬ ਬੱਬਾ (babbā) ba [bə]	ਭ ਭੱਬਾ (bhabbā) bha [bə]	ਮ ਮੱਮਾ (mam'mā) ma [mə]
ਯ ਯੱਯਾ (yayyā) ya [jə]	ਰ ਰਾਰਾ (rārā) ra [rə]	ਲ ਲੱਲਾ (lallā) la [lə]	ਵ ਵੱਵਾ (vavvā) va [wə]	ੜ ੜਾੜਾ (ṛāṛā) ṛa [ɽə]
ਸ਼ ਸ਼ੱਸ਼ਾ (śaśśā) śa [ʃə]	ਖ਼ ਖ਼ੱਖ਼ਾ (x̱ha x̱hakx̱hx̱hā) x̱ha [χə]	ਗ਼ ਗ਼ੱਗ਼ਾ (ġaġġā) ġa [ɣə]		
ਜ਼ ਜ਼ੱਜ਼ਾ (zazzā) za [zə]	ਫ਼ ਫ਼ੱਫ਼ਾ (faffā) fa [fə]	ਲ਼ ਲ਼ੱਲ਼ਾ (l̲all̲ā) l̲a [l̲ə]		

FIGURE 2 a) : Consonants (Vianjans)

Vowels and Vowel diacritics (Laga Matra)

FIGURE 2 b) : Vowels and Vowel diacritics (Laga Matra)

Other symbols

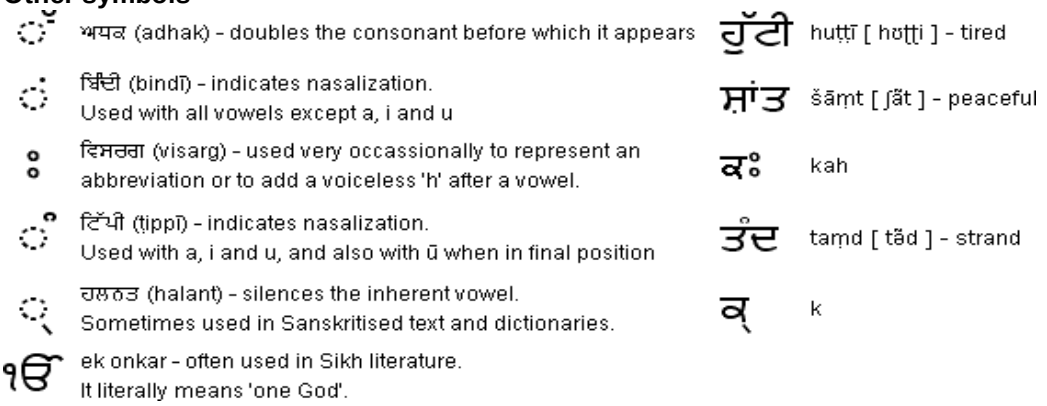

FIGURE 2 c) : Other symbols

Japanese and Arabic scripts, the existing algorithms for character segmentation of these scripts does not work efficiently for handwritten Gurmukhi script.

3. PREPROCESSING

Preprocessing is applied on the input binary document so that the effect of spurious noise can be minimized in the subsequent processing stages. In the present study, both salt and peeper noise have been removed using standard algorithm [4]. It is supposed that height and width of document can be known easily. The image is saved in the form of an array. For that purpose a 2-D array with number of rows equal to height of the document and number of columns equal to width of the document is created. Calculate the maximum intensity of pixels in the document using any standard function available in the tool used for the implementation, it is getRGB() method available in java. Scan every pixel of document and compare its intensity with the maximum intensity. If the intensity is equal to maximum intensity, store one in the array at that location, and if it is not equal store zero in the array.

4. PROPOSED PROCEDURES TO SEGMENT LINE, WORD and CHARACTER

Line Detection

The following procedure is implemented to find the location of lines in the document.

i. Create an array of size equal to height of the document and with two columns.

ii. Start from the first row and count the number of 1's in that row. If it is zero, move to next row. And if it is not zero, that is the starting location of that line. Store that location in the array.
iii. Check consecutive rows until we get 0.The before we get zero is the ending location of that line. Store that value in the array.
iv. Also calculate the location of maximum intensity in each line and store it in the second column before that line. It would be used as the starting position of characters.
v. Repeat step (ii) to (iv) for the whole document.

Word Detection

The following procedure is implemented to find location of words in each line.

i. Create a 2-D array.
ii. For each line move from 0^{th} pixel up to width.
iii. Calculate the number of one's in first column from the starting location of line to the ending position of line.
iv. If number of 1's are not zero, that is the starting location of word. Save that location in that array. Keep on moving to the right until we get no one in any column. The column with 0 1's is the ending location of the word. Store that location in array too.
v. Repeat this until we reach the width.
vi. And repeat step (ii) to (v) for each line.

Character Detection

The following procedure is implemented to find the location of character in each word.

i. Create a 3-d array. Its first index will represent line number. Second index will represent word number and third index will contain the location of character. This array will be created dynamically.
ii. Repeat the step (iii) to (iv) for each line and each word detected so far.
iii. Move from starting position of the word to the ending position of the word.
iv. Start from the starting position of line and move downwards to the ending position. Count the number of one's in that column leaving the location of line with maximum intensity. If it is not zero, that is the starting position of character. Move to right until we get column with no ones. that will be the ending location of character.

This process will generate the location of characters.

The above approach was put to number of documents; the image of one such scanned document is given below.

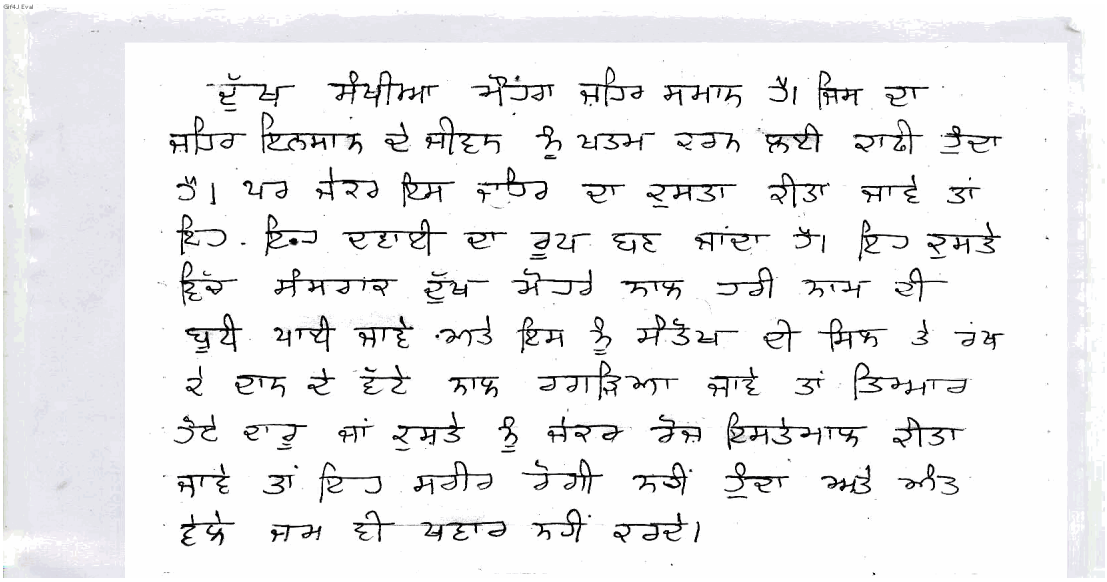

FIGURE 3: Scanned Image of a Document

The result of the scanned document after processing is given below.

FIGURE 4: Processed Document

The main objective of the work was to segment the lines, words and to segment the touching characters present in handwritten document in Gurmukhi script. We obtained the following table after putting handwritten Gurmukhi documents for segmentation. The results are summarized as in following tables:

Document	No of Lines	Correctly Detected	Inaccurate segmentation	Accuracy
Doc1	5	4	1	80%
Doc2	8	7	1	87.5%
Doc3	10	8	2	80%
Doc4	13	11	2	84.61

TABLE 1: ACCURACY for Line Segmentation

Document	No of Words	Correctly Detected	Inaccurate segmentation	Accuracy
Doc1	38	32	6	84.21%
Doc2	56	49	7	87.5%
Doc3	95	79	16	83.15%
Doc4	110	90	20	81.81

TABLE 2: ACCURACY for Word Segmentation

Document	No of Characters	Correctly Detected	Inaccurate segmentation	Accuracy
Doc1	79	71	8	89.8%
Doc2	168	145	23	86.30%
Doc3	224	175	49	78.12%
Doc4	289	232	57	80.27

TABLE 3: ACCURACY for Character Segmentation

5. CONCLUSION AND FUTURE WORK

This work was carried out to detect lines present in scanned document in handwritten Gurumukhi script. So firstly we are to find out the lines present in the document then to find words present in each line detected at the first step. Using the detected words it is to segment characters present in each word. Therefore using line detection algorithm (the first approach) lines were detected. Mostly we found the correct lines, but some were not detected correctly. The reason behind this may be the writing style of Gurumukhi script as shown in FIGURE 1. So the words presents in the

lower zone were considered as a different line. The correctly detected lines were further put to word detection algorithm. Here the results were good, but sometimes when the words were not joined properly then that was detected as a different word. The locations of the detected words were used to segment the characters. At few point segmentation was good but at few point it was not upto the expectations. This may be because of the similarity in the shapes of few characters. All these issues can be dealt in the future for handwritten documents written in 2-dimensional script like Gurumukhi by making few changes to proposed work.

6. REFERENCES

1. Y. Lu. "Machine Printed Character Segmentation – an Overview". *Pattern Recognition,* vol. 29(1): 67-80, 1995

2. M. K. Jindal, G. S. Lehal, and R. K. Sharma. "Segmentation Problems and Solutions in Printed Degraded Gurmukhi Script". IJSP, Vol 2(4),2005:ISSN 1304-4494.

3. G. S .Lehal and Chandan Singh. "Text segmentation of machine printed Gurmukhi script". Document Recognition and Retrieval VIII, Proceedings SPIE, USA, vol. 4307: 223-231, 2001.

4. Serban, Rajjan and Raymund. "Proposed Heuristic Procedures to Preprocesses Character Pattern using Line Adjacency Graphs". Pattern recognition, vol. 29(6): 951-975, 1996.

5. Veena Bansal and R.M.K. Sinha. "Segmentation of touching and Fused Devanagari characters, ". Pattern recognition, vol. 35: 875-893, 2002.

6. R. G. Casy and E. Lecolinet. "A survey of methods and strategies in character segmentation". IEEE PAMI, Vol. 18:690 – 706,1996.

7. U. Pal and Sagarika Datta. "Segmentation of Bangla Unconstrained Handwritten Text". Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR), 2003.

8. U. Pal, S. Sinha and B. B. Chaudhuri. "Multi-Script Line identification from Indian Documents", Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR) 2003.

9. Rajean Plamondon, Sargur N. Srihari. "On – Line and Off – Line Handwritting Recognition: A Comprehensive Survey", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol 22(1). Janurary, 2000.

10. Giovanni Seni and Edward Cohen. " External word segmentation of off – line handwritten text lines". Pattern Recognition, Vol. 27(1): 41-52, 1994.