Character Segmentation in Gurumukhi Handwritten Text using Hybrid Approach

Rajiv Kumar and Amardeep Singh

***Abstract*—The desire to edit scanned text document forces the researchers to think about the optical character recognition (OCR). OCR is the process of recognizing a segmented part of the scanned image as a character. OCR process consists of three major sub processes - pre processing, segmentation and then recognition. Out of these three, the segmentation process is the most important phase of the overall OCR process. It is the most significant process because if the output of segmentation phase is incorrect then we can not expect the correct results; it is just like garbage in and garbage out. But on the same time, segmentation is complex too. If the document is handwritten then the situation becomes more cumbersome, because in that case only few points are there which can be used to make segmentation. In this paper, we formulate an algorithm to segment the scanned document image as a character. As per our earlier published work, the information about the lines and words within each line is written in a data file. According to proposed algorithm, one part is extracted from the word present in the line. This extracted part is checked whether it has some meaningful symbol (as per Gurumukhi script). If it has then the extracted part is marked and written in the file, otherwise the extracted part is readjusted to find the symbol. For classification, we have used hybrid approach which consists of water reservoir and feature extraction approach. This concept was implemented and got good reasonable results.**

***Index Terms*—OCR, Segmentation, gurumukhi, handwritten, feature, water reservoir.**

I. Introduction

The subject of optical character recognition has received considerable attention in recent years. In case of character, computer aided character extraction and classification, is held back by both the scope of useful solution and the computational power of the time. For the past 35 years, the research community shows huge interest in language recognition problem. The subject has attracted immense research interest not only because of the challenging nature of the problem, but also because it provides a commercial angle to the final end product, that is, the final product can be used for automatic processing of large volumes of data such as postal codes, automatic cheque amount reading in banking environments and for office automation. The basic problem is to assign the digitized character into its symbolic class. Written language recognition is the task of transforming language represented in its spatial form of graphical marks into its symbolic representation. After capturing the text image, it is passed through various phases. One can name them as pre processing, segmentation, and then recognition to have a lexical meaning.

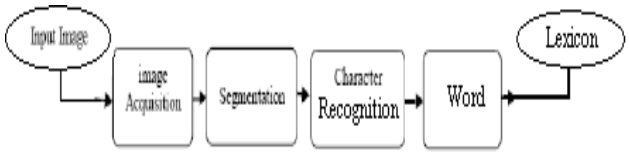

Fig. 1. Flow chart showing different phases.

In optical character recognition (OCR), a perfect segmentation of characters is required before the recognition of individual characters i.e. recognition is possible only if segmentation is correct. Character segmentation can be defined as a technique, which partitions images of lines or words into individual characters. It is an operation that seeks to decompose an image of a sequence of character into sub-images of individual symbols. Its decision, that a pattern isolated from the image is that of character (or other identifiable unit), can be right or wrong. It is a critical step because incorrectly segmented characters are not likely to be correctly recognised. According to a survey of vast literature done by Casey *et. al.* and Sridhar *et. al.*, *[1] & [2]* there are three pure strategies for segmentation, and plus numerous hybrid approaches that are weighted combination of these three pure ones. The elementary strategies are **The Classical Approach**, **Recognition Based Segmentation, and Holistic Methods.** In Classical approach, the segmentations points are identified based upon character-like properties. This method is also called as dissection method. In Recognition Based Segmentation, the system searches the image for components that match classes in alphabet. As per Holistic Methods, the system seeks to recognize words as a whole, thus avoiding the need to segment into characters.

These three strategies are shown to occupy orthogonal axes. Hybrid methods can be represented as weighted combinations of these lying at points in the intervening space.

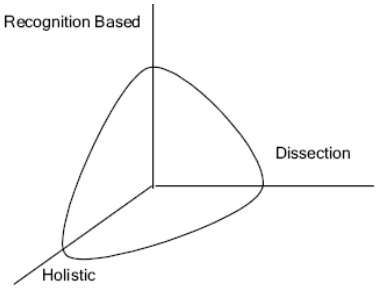

Fig. 2. A 3 D space representing the strategies of segmentation.

II. Gurumukhi Script and Its Characteristics

Our work is related with the segmentation of handwritten

Manuscript received June 26, 2010; revised May 20, 2011.
Rajiv Kumar, Thapar University (email: rajiv.patiala@gmail.com)
Amardeep Singh, Pbi University (email: amardeep_dhiman@yahoo.com)

text written in Gurumukhi script which is one of the popular scripts used to write Punjabi , a popular spoken language of northern India. Gurumukhi script alphabet consists of 41 consonants and 12 vowels [3]. 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 Gurumukhi, there is no concept of upper or lowercase characters. A line of Gurumukhi 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 the following figure. The upper and lower zones may contain parts of vowel modifiers and diacritical markers.

Fig. 3. 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

III. Pre Requisite Processing

The quality of scanned image depends upon the scanner type too and it plays an important role in segmentation. We are using higher end scanner for the scanning purposes. In the file, the image consists of the shapes / symbols of handwritten characters in Gurumukhi script. Between any two lines and any two words there is a definite gap of minimum width. A line is supposed to have different words and the words are made up of one or more characters. As per processing requirements, only two types of information is sufficient, which is either zero or one. But the image file is in grey scale. So first of all it is important to convert image file information to zero or one. The simplest and commonly accepted criterion is that, first the average of intensities of all the pixels present in the document image file is calculated. Then the intensity of each pixel is set as per the following rule:

If pixel intensity is less than Average intensity then
  set pixel intensity = 0
else
  set pixel intensity = 1

The lines consisting of words are generally straight in nature. If there is any skew then present work may not work properly.

IV. Hybrid Character Segmentation Algorithm

The whole image is considered as a large window. This can be achieved by the trapping the values while reading the BMP file. This information is always available itself in the BMP file and is independent of any language used to process the BMP file. From this large window, next is to find a window (smaller than the earlier found larger window) consisting of a line. This is done by finding minimum and maximum of X coordinates and minimum and maximum of Y coordinates for two no pixel zones. No pixel zone is the zone having no pixels. The Lines and words have been detected by flexible windowing, [4] & [5].

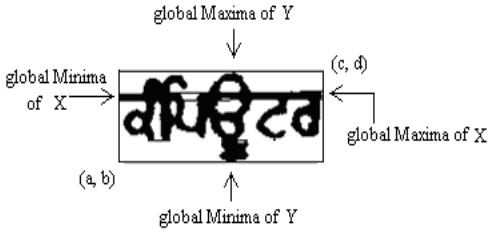

Fig. 4. Window with coordinates (a, b) and (c, d)

The coordinates of each line and words are written in a file. The data in the file is in the following format.

L1, W1, W2, W3; endl
L2, W1, W2, W3, W4; endl
……
Lr, W1, W2, W3; endl
……
Ln, W1, W2, W3; endl

Following is the proposed algorithm:

```
Hybrid_Character_Segmentation(Data File as Input File)
{
//Read Window coordinate to Line.window of first line from data file.
Get Line.window from Data File
While (Line.window < > NULL)
{ //Read window coordinates of word present in current line to Word.window
        Get Word.window
        While(Word.window < >NULL)
        {       Write Word . window to Data Out File
                Fpix . X = Word . window . a
                While (FPIX.X < Word.window. (a + c))
                {
                        Call Find_Cutting_Point ( WORD.window, FPIX )
                        Call Classifier (FPIX, SPIX, FLAG)
                        If(FLAG) then
                        {
                                FPIX.X = SPIX.X +1
                                Write FPIX . X to Data Out File
                                SPIX. X = 0
                        }
                        Else
                        {
                                SPIX . X = SPIX . X +1
                        }
                }
                // Read Next Word Window
                Get Word.window from current line.
        }
        //Change to next line
        Get Line.window coordinates from date file
}
}
        Find_Cutting_Point ( WORD.window, FPIX )
        {POINT TPOINT
                int Prev_Min, Min, FirstFlag=0
                //Along head line, move from left to right from position FPIX . X.
                //Find minimum number of pixel present in vertical column. Set it to SPIX.
                // d is the number of pixel present in the head line of the word.
                Prev_Min = TPOINT. X = FPIX . X
                If ( TPOINT. X > = Word . window. MAXX ) then return NULL
                Else
                {    While (TPOINT . X < Word.window. MAXX )
                        {
                                TPOINT. X = TPOINT. X + 1
                                Get number of Pixels present in the TPOINT. X and set to variable COUNT
                                If(FirstFlag == 0)
                                {       Prev_Min = COUNT
                                        FirstFlag = 1
                                }
                                Else
                                {
                                        Min = COUNT
                                }
                                If( Min – Prev. Min < = d )
                                {
                                        return TPOINT
                                }
                        }
                }
        }
```

The work related with The Classifier Module [6], called in the above algorithm, is already published. So it would not be appropriate to discuss the whole concept again. Results obtained are summarized in the following table.

TABLE 1: ACCURACY FOR CHARACTER SEGMENTATION

Document	No of Characters	Correctly Detected	Inaccurate segmentation	Accuracy
Doc1	89	84	5	94.38%
Doc2	148	138	10	93.24%
Doc3	223	209	14	93.72%
Doc4	278	255	23	91.72%

V. CONCLUSION

The approach developed consists of two parts - first is to get segmented area (say) SA, second is to check whether SA has meaningful symbol or not. The role of Classifier Module comes to picture in second part. Classifier Module returns True if SA has something meaningful (as per classification scheme) or otherwise returns False. On True, the area SA is marked and is written in the Data file. On False, the area in SA is readjusted. Here, to some extent, a recognition based approach is used but exactly recognition is not done. Further it is to be clarified that, to get correct recognition is not the motive of the present work but it is used to confirm segmentation as per the classes and indirectly correct segmentation. It is a reverse approach to ensure correct segmentation.

There was some wrong segmentation for characters too. Certain Gurumukhi characters are combined in nature as shown in the following figure.

Fig. 5. Gurumukhi Combined Character

The present work results it as a segmentation of one character, but ideally it should be segmented as two characters – one character as between the lines 2 and 4 and the other as between the lines 4 and 5. The shape of some of Gurumukhi characters (few in numbers) plays an important role. But overall the concept is working,. We can conclude that it was good for the character segmentation.

REFERENCES

[1] Casey, R.G. and Lecolinet, E., "A Survey of Methods and Strategies in Character Segmentation". IEEE Transactions on Pattern Analysis and Machine Intelligence, 1996, Vol.18, No.8, pp.690-706.

[2] Liang, S., Shridhar, M. and Ahmadi, M., "Segmentation of Touching Characters in Printed Document Recognition". Pattern Recognition, 1994, Vol.27, No.6, pp.825-840.

[3] Rajiv K. Sharma and Amardeep S. Dhiman, "Challenges in Segmentation of Text in Handwritten Gurmukhi Script". Proceedings of BAIP 2010, CCIS 70, Springer-Verlag Berlin Heidelberg, pp. 388–392.

[4] Rajiv K. Sharma and Amardeep Singh, "Segmentation of Handwritten Text in Gurmukhi Script". International Journal of Computer Science and Security, 2008, Vol.2. No.3, pp 12-17.

[5] Rajiv K. Sharma and Amardeep Singh, "Detection and Segmentation of Handwritten Text in Gurmukhi Script using Flexible Windowing". IJCTE, 2010, Vol 2, No. 3, pp. 329 – 332.

[6] Antarpreet Kaur, Rajiv K. Sharma, and Amardeep Singh, "A Hybrid Approach to Classify Gurmukhi Script Characters". International Journal of Recent Trends in Engineering, 2010, Vol 3, No. 3, pp 103-105.

[7] M. K. Jindal, G. S. Lehal, and R. K. Sharma. "Segmentation Problems and Solutions in Printed Degraded Gurmukhi Script". IJSP, 2005, Vol 2, No. 4.

[8] G. S .Lehal and Chandan Singh. "Text segmentation of machine printed Gurmukhi script". Document Recognition and Retrieval VIII, Proceedings SPIE, USA, , 2001,Vol. 4307, pp. 223-231.

[9] Veena Bansal and R.M.K. Sinha. "Segmentation of touching and Fused Devanagari characters". Pattern recognition, 2002, Vol. 35, pp. 875-893.

[10] Giovanni Seni and Edward Cohen. "External word segmentation of off – line handwritten text lines". Pattern Recognition, 1994, Vol. 27, No. 1, pp. 41-52.

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

[12] Devasar, N. M., Madan, S., and Singh, H., "A Hybrid Approach to Character Segmentation of Gurmukhi Script Characters". Proceedings of the 32nd Applied Imagery Pattern Recognition Workshop (AIPR'03), 2003.

[13] Perminder Singh, "A Technique for Preprocessing and Segmentation of Printed Text in Gurmukhi Script". M.Tech.thesis submitted to Debt. Of Comp. Sc. & Engg., Punjabi University, Patiala., 1997.

[14] Fujisawa, H.; Nakano, Y. and Kurino, K., "Segmentation Methods for Character Recognition: From Segmentation to Document Structure Analysis". Proceedings of the IEEE, 1992, vol.80, No.7, pp.1079-1091.