Handwritten Text Image Compression for Indic Script Document

Smita V. Khangar
Department of Computer Science and Engg.
G.H.Raisoni College of Engg.
Nagpur, India

Latesh G. Malik, PhD.
Prof. Department of Computer Science and Engg.
G.H.Raisoni College of Engg.
Nagpur, India

ABSTRACT
In this paper, compression scheme is presented for Indian Language handwritten text document images. Document image compression is an active area of research. Current OCR technology is not effective for handling the handwritten text images. The proposed compression scheme deals with the handwritten gray level document in Devnagri script. The method is based on the separation of foreground and background of an image and connected component labeling. Experiments are done with handwritten images in Devnagri (Hindi and Marathi). Compression schemes are available for the printed text in Indian language. But there is little work reported towards the compression standards for handwritten text image. The results of the modules are showing good compression ratio. Hence compression of handwritten text images in Indian language is important.

General Terms
Image Processing, Data Compression, Experimentation, Verification.

Keywords
Handwritten Text, Connected Component Labeling Compression, Indian Language, Devnagri Script, Gray Level Document.

1. INTRODUCTION
Today most of the digital libraries publish the document over the web. Due to increasing demand of the digital libraries document image compression is becoming more popular in recent years. Also the publication in the form of periodicals and manuscript is increasing over the web. These documents are in the form of Handwritten Text and printed text. Current digital libraries use the Optical Character Recognition to extract the text. In Indian Language (IL) context most of the important documents are in printed or handwritten form. Therefore converting these documents into electronic form is difficult task. For Indian Language scripts are having different variations, hence there is unavailability of the OCR system for most of the Indian language. Also OCR system works for the printed text. But for handwritten text this method may not work properly. Therefore storage of these documents in the form of scanned images is important. Also the compression methods achieving high compression ratio is required for retrieval and accessing of these document images over the web. In Indian Language, document with handwritten text have historical significance. Thus for speedy communication over the web this data must be in compressed form. Compression techniques for printed scanned image documents in Indian language are available in literature. But there is absence of handwritten document image compression in Indian language. Most of the work is done for the Chinese and Arabic handwritten document image compression.

For document image compression data is represented in the form of images, but it mostly contains printed text. Such images are known as textual images [1]. Different approaches for compression of such textual images are used followed by various compression standards. Soft pattern matching (SPM) and Pattern matching and substitution (PMS) with JBIG2 standards for bi-level images are used in literature [2]. These methods are working only for the printed text, but for handwritten text methods are not working well. In SPM and PMS the image is divided into the marks. These groups of marks contain letters, digits, and symbols. In PMS only the one symbol bitmap representative of each group is required. It is followed by position of each character. Coding of new symbol is done from the symbol dictionary with smallest mismatched. The method gives high compression ratio for repeated symbols, which is available in printed text. But for handwritten text similarity of each character cannot be predefined. This method leads to substitution error if matching marks are not found [3]. In SPM if matching mark is found then coding is done directly. Unlike PMS even for totally mismatched mark it is not produce any error [2]. These methods are suitable to the classification of pattern used in languages. It is much more suitable for English language then Indian language. Handwritten text shows the difference in lines, shapes, similarity and strokes varying from person to person. Thus compression standards like JPEG and JPEG2000 may not suitable because it may degrade the high frequency component and cause bluring of lines and strokes [4].

As mentioned earlier there is unavailability of the compression methods for Indian language handwritten text. Thus major focus is on the compression of handwritten text in Indian language context. In this paper compression of scan digitized handwritten gray level images for Devnagri script is proposed. Next section gives the brief overview of foreground-background separation techniques and features of Devnagri script. The rest of the paper describes new compression method based on separation of foreground and background of an image and connected component labeling (CCL) followed by experimental results and conclusion.

1.1 Survey of Foreground-Background Separation of an Image
Many of the documents such as magazine articles and color documents foreground and background are important. For text images foreground of an image shows mainly textual matter. In some cases the text can also be written on background image. Compare to printed text, handwritten text shows distinct features for such images. Also the separation of

foreground and background varies with the color and gray level documents. Many of the time background is of uniform color and foreground shows the text in terms of the characters, lines etc. Also when the text has been written with the different sized tip like broad tip, quill tip or pencil stroke marks are different. When such images are scan with different dpi (dot per inches) results in variation of background color. This is also applicable for gray level images. Hence unlike printed documents, handwritten text documents may not show very well contrast in foreground and background [5].

For gray level images many of the separation techniques are often based on thresholding. Thresholding can be local or global. B. Gatos uses the method of adaptive binarization for gray scale images for historical documents [6]. Kittler and Illingworth choose the threshold based on every region or pixel [7]. XU Danhua separated the foreground and background for scanned receipt and handwritten text with the median filter of large size window [4]. Technique proposed by U. Garain is based on the connected component detection followed by the dominant background components detection in terms of member pixels. After that segmentation of entire image into hierarchical rectangular blocks of different size and bicolor clustering of each block has been done. It then form binarized image [5]. A popular DiVu document image compressor effectively makes the separation of foreground and background. The separation algorithm segment image into two separately compressed layers [8]. Then foreground and background layer is compressed separately. This method gives results for binary images with uniform background. For printed text, method works very well but does not work for the handwritten text image having low contrast.

1.2 Properties of Devnagri Script

Since the compression techniques targeted for compression of the handwritten document images in Devnagri, some basic properties of this script is discussed here.

The Devnagri alphabet is used for writing the Marathi, Hindi, Nepali, and Sindhi. The Devnagri script does not have any uppercase and lowercase distinction. Also the writing style is left to right horizontal. The script has 5 basic vowels and 29 consonants. It is also having 12 modifiers. The characters of the words in this script are connected by a horizontal line known as "Shirorekha". The neighboring characters are touched through this headline and form the connected components [9].

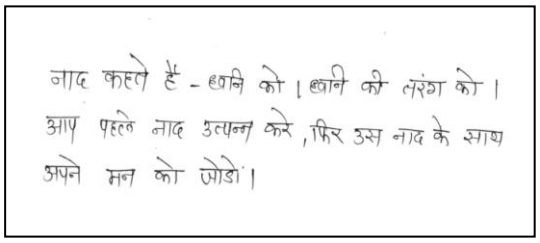

Fig 1. Handwritten Textual image in Devnagri script

Most of the Indian script is dividing into the three zones. Fig. 1 shows an example of handwritten Devnagri Text. Upper zone consist of the matra information above the headline. Middle zone showing the characters and bottom zone may show other consonants to form the complex characters

2. THE PROPOSED WORK FLOW

Depending upon the assumption that system uses only the gray scale images with uniform background the workflow is shown in figure 2.

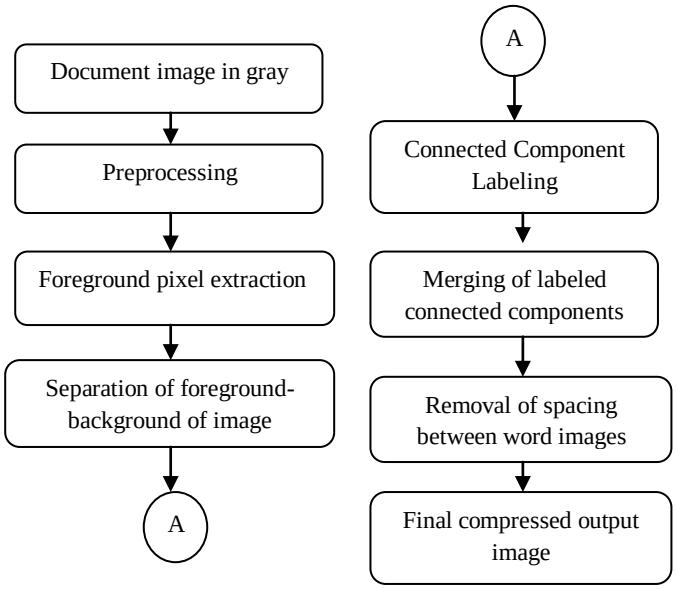

Fig. 2. Flow of proposed work flow

3. THE COMPRESSION STRATEGY

The preprocessing step from the working flow is optional here. It is used when image is depredated while scanning the image or any kind of noise is occurred. It then consists of following steps.

3.1 Foreground Pixel Extraction

The input handwritten document is scanned in gray level with different dpi for testing purpose. For this input, image width and image height is calculated. The values of the image pixels in terms of RGBs are read and stored into the buffer. From the various range of the values of the image pixels an average value is calculated and referred as threshold. Along with average value foreground and background pixel values for the input image is calculated. These pixel values are used in further steps.

3.2 Separation of Foreground and Background of Image

As the compression methodology focus on gray level images, there is a distinction between the binarization and foreground-background separation for gray and color documents. For gray level document it may be easy to separate the foreground-background than color documents. Ideally binarization separates the background (paper) from the foreground (characters and text) of an image. Binarization is conversion of gray image into the black and white image [5]. In this step the calculated foreground and background pixel values are separated from the buffer. These values are in vector form. To produce the foreground extracted text image, the values are written to the array elements of raster. The raster defines the value of pixel in a particular area to be written [10]. This raster is stored on to the disk producing the foreground extracted text image of the original image.

The foreground output image has only textual form of information. The background of the input image is now

replacing with the white value. There is a reduction in the stored new foreground image depending on with which pen it is writing and at which dpi the document is scanned. Image shown in figure 3 is scanned at 100 dpi and written with the broad tip angled pen on A4 size white background paper. The image is having dimension in pixels 803×1146.

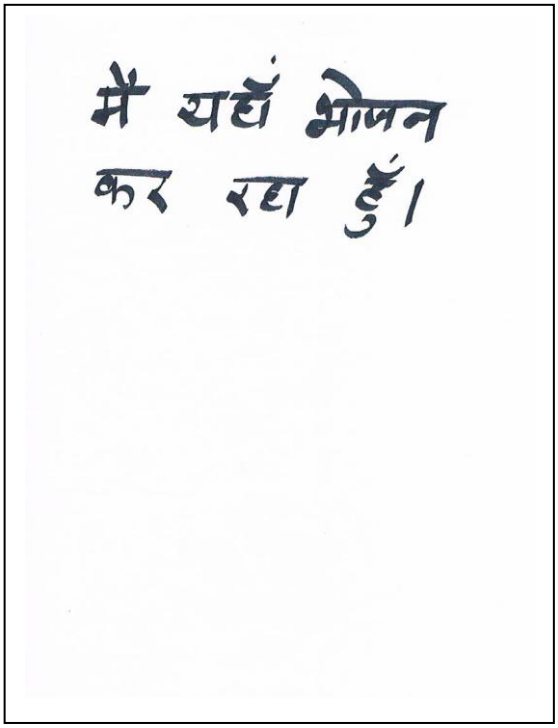

Fig. 3. Original image Input_Image1.jpg

The original image shown in fig. 3 is having jpg format and written in Hindi Devnagri script. The images with the .png and .bmp are also showing significant results. This image is having the size 44.7 KB. By extracting the foreground text from this image, size of the image reduced to 30.1 KB. The corresponding foreground extracted text image is shown in figure 4.

3.3 Connected Component Labeling

The connected component labeling (CCL) is executed on the foreground extracted image to detect all the connected components. CCL is done typically on output from other image processing steps. Connected components labeling is a process of assigning the unique label to each connected or touching component in binary image [11]. The connectedness between the pixels can be defined in two common ways: 4-connected and 8-connected. The algorithm used of CCL can be of multipass, one pass and two pass algorithm. Our approach is based on the 8–connected two pass algorithm. On 2D array of image, a forward scan assign labeling from left to right and top to bottom [12].

Two pass algorithm [12, 13] works in two phases, Scanning phase and Labeling phase. In scanning phase, image is scan row by row, records the equivalence information and assign the temporary labels. Labeling phase assign the final labels by replacing each temporary labels of its equivalent class. To represent this equivalence information data structure union-find is used [11, 14]. Conceptually it like a rooted tree, where nodes of a tree represent the temporary labels and edge represent the equivalence information between the labels [15]. This union-find data structure unites the tree and returns the root label.

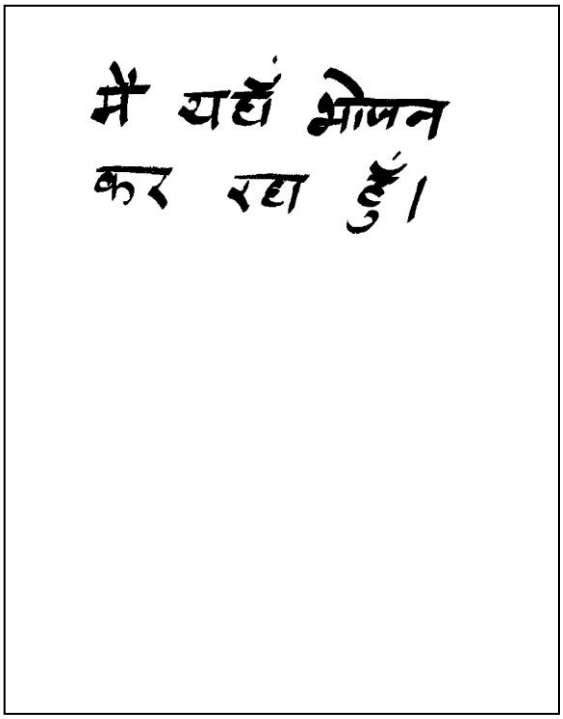

Fig. 4. Foreground extracted text image foreground_Input_Image1.jpg

3.3.1 Detection of Connected Components

At first by using CCL two pass algorithms, labeled connected components from the foreground image is extracted.

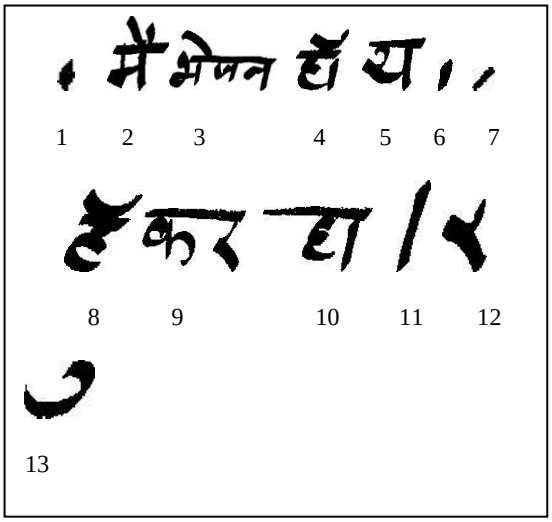

Fig. 5. Connected components labeling for Input_Image1.jpg

For the image shown in fig 4, 13 connected components are detected. Figure 5 shows the all 13 connected components. As the algorithm scan the image row by row, whichever the first component of the pixel to be scanned first, it assign the labels accordingly. Thus the labels information is important here not the number.

For word images connected component labeling gives two types of components. Some components represent the

complete word images or punctuation symbols and other represent the parts of word images [8]. Components showing the parts of word images involve the further processing for generation of word images.

For each connected component its bounding box is calculated. Bounding box is rectangle showing the component-extent positions from the topmost left corner to the rightmost bottom corner. It can be represented by four tuple like (xt, yl, xb, yr) where the (xt,yl) and (xb,yr) represent the left top and right bottom corner respectively. This can be done by calculating the height and width of each component and save it on the bounding box array. Component height can be calculated as (yr-yt +1) and component width as (xb-xt+1). The final array of component height and width gives the component image to be written on disk.

3.3.2 Component Merging

A labeled component image can be represented by two or more components images. It can be reconstructed by merging of its corresponding components. In following conditions the merging of the components is to be done.

Condition 1. Merge only those components which are either overlap or touch each other boundary.

Condition 2. Let A and B are the two overlapping components. The diagonal distance of bounding box, ccbDiagonal and connected component diagonal distance ccDiagonal is computed. If (ccDiagonal<ccbDiagonal) then X and Y are merged together. This is for merging of small components into the large components.

Condition 3. Merging of adjacent components into subsequent passes. It means finding the subcomponents which are the parts of labeled components; it should be merged into single components. Then removal of the subcomponents from their pixel positions since small components are merged into its adjacent component.

It is to be noted that in worst cases if the component is too away from the bounding box pixel positions, then simply discard that components. Although the overall performance is not affected much.

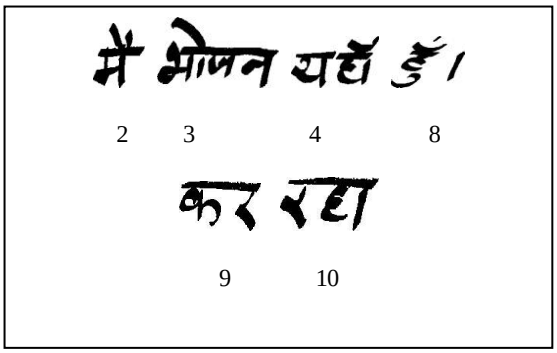

Fig. 6. Component merging

From the figure 6 total six components are merged and this image is written to the disk.

3.4 Removal of Spacing between Word Images

Removal of spacing is done between the merged components of word images. Also the spacing between the top and left boundaries of the merged image. For this purpose components has to be shifted in X-horizontal (row by row) and Y-vertical directions. Let Xshift and Yshift denotes the shifting in X and Y direction respectively. The starting row position pixel values and end row position pixel values are computed before the occurrence of first component. As the first component occurs, Yshift is calculated as (Yshift=StartY2-EndY1). By using this Yshift pixel position value the components are shifted in vertical direction. The iteration continues until the last row of the components. For shifting in X direction, HighestRight of the component is calculated. HighestRight =ccBoundingbox [arrayIndexof (rowComponent)-1] where ccBoundingbox denotes array of connected component bounding box. The distance between arrayIndexof left position and HighestRight is greater than two pixel positions then they are shifted in horizontal direction and Xshift is calculated.

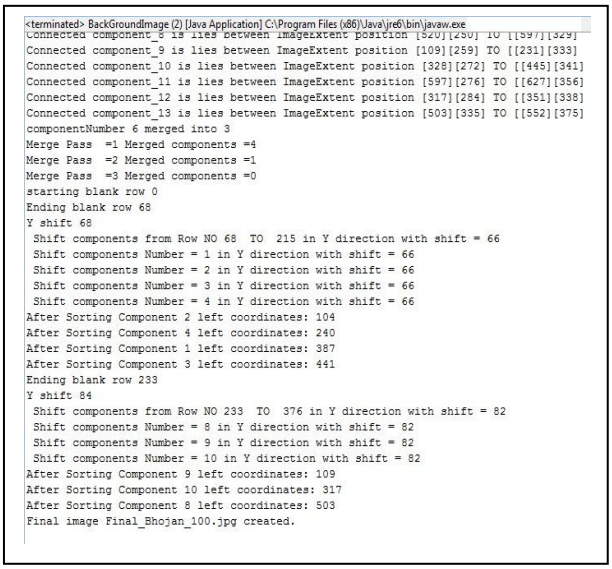

Fig. 7. Snapshot showing shifting pixel position in X &Y direction

With the removal of spacing between the word images, there is no difference in textual matter of an image. Also the image size is reduced from the foreground image size. The final image is in compressed form than original image.

Fig. 8. Final Compressed output image

Figure 8. shows the output image after removing spacing between the word images. This output image is of 21.9 KB in size.

4. EXPERIMENTAL RESULTS

The documents scanned at varying resolution starting from 100 to 300 dpi. The images are written with the different tip of pens. All the images are stored in jpg format. However it can compress the .png and .bmp image format also. Text documents are written with the Devnagri script (Hindi and Marathi). Depending upon style of writer handwritten text

show variety in lines and shapes. Thus there is difference in storage space of different images. Table 1 shows the compression size for various streams of images. There is variation in compression ratio in various sample files because the documents are written with verity of pens. Whenever the documents are written with the pin pointed pen or broad tip pen, stroke marks are different and there is a variation in the image size. The experimental results are only done for the gray level documents.

Compression algorithm based on SPM and PMS are not giving the significant results for handwritten document, as they are based on pattern matching. Due to the variation in writer's style, handwritten characters similarity matches found is difficult. Unlike printed text documents, handwritten text documents do not fitted into any compression standards. However CCITT Gr-3 and Gr-4 are emerging standards.

5. CONCLUSION

In this paper compression strategy of handwritten text for Indian language gray level document is presented. To the best of our knowledge, this is the first effort towards the compression of handwritten text for Indian language documents. As mentioned earlier, most of the work is done for foreign language handwritten document. The proposed methodology only focuses on the gray level handwritten images. The compression technique presented here is lossless in nature. The compression technique works well and gives results accordingly. It can be extended for the color document also. This aspect is left for future extension of current study.

Table 1. Size of the intermediate and final output files for some sample documents

Image Name Height ×Width	Written With	Scanned at dpi	Original Size	Foreground image size	Final Output image size	%Compression Ratio
Image1.jpg 803×1146	Broad tip angled pen	100dpi	44.7 KB	30.1 KB	21.9 KB	49
Image2.jpg 708×1017	Sketch Pen	100 dpi	382 KB	91.6 KB	85.5 KB	22.39
Image3.jpg 1275×1750	Broad tip pen	150 dpi	169 KB	158 KB	146 KB	86.39
Image4.jpg 1700×2296	Sketch pen (Pointed tip)	200 dpi	234 KB	181 KB	150 KB	64.10
Image5.jpg 2550×3507	Small tip Ball pen	300 dpi	203 KB	193 KB	107 KB	52.70
Image6.jpg 2409×3437	Broad tip Sketch pen	300 dpi	441 KB	235 KB	169 KB	38.33

6. REFERENCES

[1] I.Witeten, T.Bell, H.Emberson, A.Moffat, J. 1994. Textual Image Compression: Two Stage Lossy/ Lossless Encoding of Textual Images. In Proceedings of the IEEE. transaction.

[2] Y.Ye and P.Cosman, J.2001. "Dictionary design for text image compression with JBIG2". IEEE Transcation on Image Processing.

[3] P.G.Howard, "Text image compression using soft pattern matching" , The Computer Journal,1997.

[4] X.Danhua, B.Xudong, 2009. High efficient compression strategy for scanned receipts and handwritten documents. IEEE International Conference on Information and Engineering.

[5] U.Garain, T.Paquet, L.Heutte, "On foreground-background separation in low quality document images", International Journal of Document Analysis, 2006.

[6] B.Gatos, I.Pratikakis, 2004. An Adaptive technique for low quality historical documents. 6th International Workshop on Document Analysis Systems, vol.3163,pp.102-113,2004.

[7] J.Kittler, J.Illingworth, 1985. Threshold Selection based on Simple Image Stastics. Computer Vision Graphics and Image Processing.

[8] L.Bottou, P.Howard, "High quality document image compression with DjVu", International Journal of Electronic Imaging,1998.

[9] U.Garain, S.Debnath, A.Mandal, B.Chaudhari 2003. Compression of scan digitized printed Text: A soft pattern matching technique. ACM Symposium on Document Engineering.

[10] Java Sun Documentation. [Online]. Available: https://docs.oracle.com/javase/1.3/docs/api.

[11] Michel B. Dillencourt, Hannan Samet, Markku Tamminen. A General approach to connected component labeling for arbitrary image representations. J.ACM 1992.

[12] Kesheng Wu, Ekow Otoo, Kenji Suzuki J." Optimizing two passs connected component labeling algorithms", Journal of Pattern Analysis and Application, 2009.

[13] S. Naoi, 1995. High speed labeling method using adaptive variable window size for character shape feature. IEEE Asian Conference on Computer Vision.

[14] Chirstophe Fiorio and Jens Gusted. 1996 Two linear time union-find strategies for image processing. Theoretical computer sci.

[15] Benarard A. Galler and Michel Fisher. 1964 An improved equivalence algorithm. Communication on ACM.