International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume 3, Issue 1 (Jan-Feb 2015), PP. 128-131

RECOGNITION AND CONVERSION OF HANDWRITTEN MODI CHARACTERS

Prof. Mrs. Snehal R. Rathi[1], Rohini H. Jadhav[2], Rushikesh A. Ambildhok[3]
VIIT college,
University of Pune, India
Snehal_rathi@rediffmail.com[1], jadhavrohini63@gmail.com[2], rushiamby@gmail.com[3]

ABSTRACT- The technical study had been performed on many foreign languages like Japanese; Chinese etc. but the efforts on Indian ancient script is still immature. As the Modi script language is ancient and cursive type, the OCR of it is still not widely available. As per our knowledge, Prof. D.N.Besekar, Dept. of Computer Science, Shri. Shivaji College of Science, Akola had proposed a system for recognition of offline handwritten MODI script Vowels. The challenges of recognition of handwritten Modi characters are very high due to the varying writing style of each individual. Many vital documents with precious information have been written in Modi and currently, these documents have been stored and preserved in temples and museums. Over a period of time these documents will wither away if not given due attention. In this paper we propose a system for recognition of handwritten Modi script characters; the proposed method uses Image processing techniques and algorithms which are described below.

General Terms
Preprocessing techniques: Gray scaling, Thresholding, Boundary detection, Thinning, cropping, scaling, Template generation. Other algorithms used- Average method, otsu method, Stentiford method, Template-based matching method.

Keywords- MODI Script, handwritten character recognition (HCR), Image processing.

I. INTRODUCTION

Handwritten character recognition has been a popular field of research but it is still an open problem. The challenging nature of handwritten character recognition has attracted the attention of researchers from industry and academic people. The recognition task of Modi script is very difficult because the Modi handwritten characters are naturally of two types, cursive and unconstrained. There is high similarity between character and distorted and broken characters. Hence the extreme variation is observed between the collected character samples. The proposed work is an attempt for handwritten Modi characters recognition and conversion into corresponding English character.

Section II describes about Modi script. In section III Recognition model is discussed. Section IV covers Preprocessing steps taken. Section V, discusses about Feature extraction methods. Classification method is explained in section VI. Result and Discussion is covered in section VII. Future work is discussed in section VIII and conclusion explained in section IX.

II. MODI

Modi is one of the scripts used to write the Marathi language, which is the primary language spoken in the state of Maharashtra in western India. There are several theories about the origin of this script. One of them claims that in 12th Century MODI was developed by '*Hemandpant*' or '*Hemadri*', (a wellknown administrator in the kingdom of '*Mahadev Yadav*' and '*Ramdev Yadav*' ('*Raja Ramdevrai*', Last king of '*Yadav empire*' (1187-1318 at '*Devgiri*'.). Dr. Rajwade and Dr. Bhandarkar believes that *Hemandpant* brought MODI script from Sri Lanka, but according to *Chandorkar*, MODI script has evolved from *Mouryi (Bramhi)* script of *Ashoka* period.

The Modi alphabet was invented during the 17th century to write the Marathi language of Maharashtra. It is a variant of the Devanāgarī alphabet. The Modi alphabet was used until 1950 when it was replaced by the Devanāgarī alphabet. Modi alphabets are classified into vowels, consonants and numerals.

Notable features are that each letter has an inherent vowel (a). Other vowels are indicated using a variety of diacritics which appear above, below, in front of or after the main letter. Some vowels are indicated by modifying the consonant letter itself.

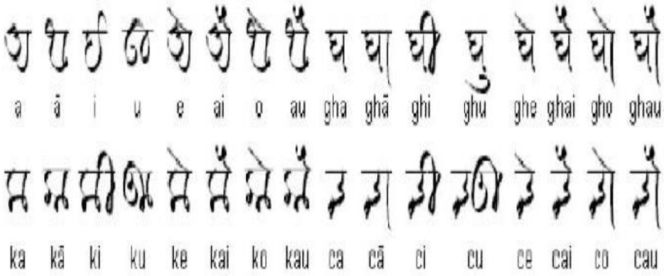

a ā i u e ai o au gha ghā ghi ghu ghe ghai gho ghau
ka kā ki ku ke kai ko kau ca cā ci cu ce cai co cau

Fig. 1: Modi Vowels and diacritics

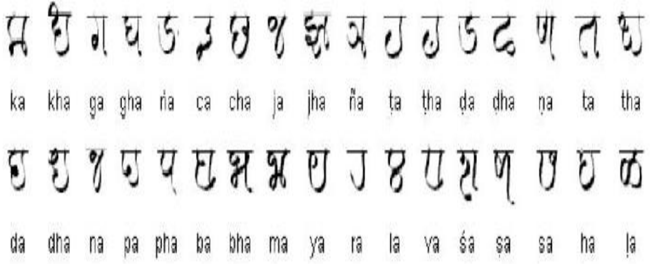

ka kha ga gha ṅa ca cha ja jha ña ṭa ṭha ḍa ḍha ṇa ta tha
da dha na pa pha ba bha ma ya ra la va śa ṣa sa ha ḷa

Fig. 2: Modi Consonants

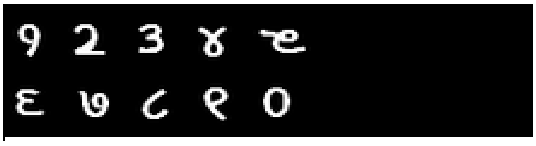


Fig. 3: Modi Numbers

III. RECOGNITION MODEL

The Character Recognition process includes some vital sub steps like: Preprocessing, Feature Extraction, and Post Processing. The block diagram of typical character recognition is shown in Fig. 4. The preprocessing steps are described in step IV.

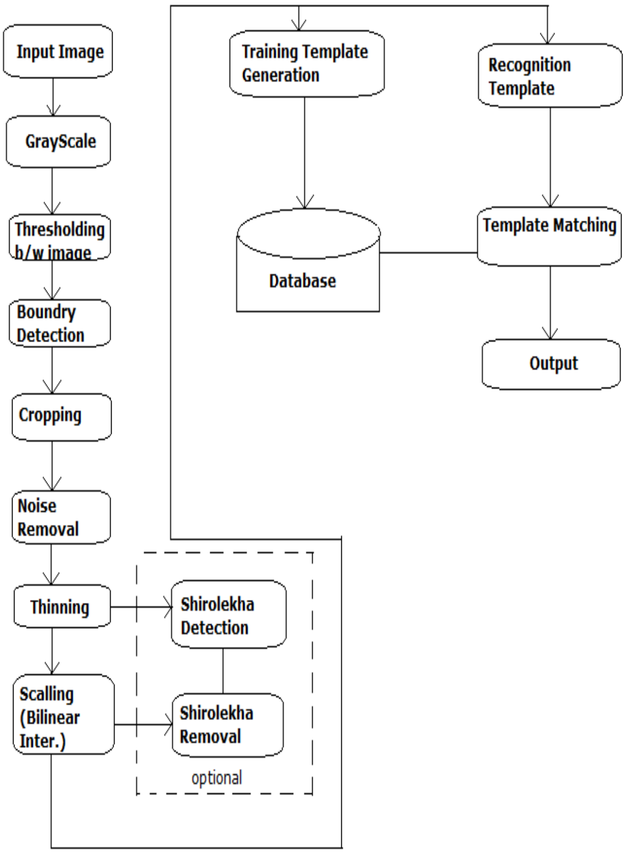


Fig.4: Block Diagram

IV. PREPROCESSING

Preprocessing is a very important step in any optical or any handwritten character recognition system. As a preliminary work we have collected papers containing Modi characters, written by different people without considering the variations in ink or pen. It contains weak, broken and distorted characters also. We have scanned these pages using 200,300 or 600 DPI and stored as JPG, BMP or TIF format. On these scanned input images we are going to perform some preprocessing methods as shown in Fig.4. Initially by applying Average method we are going to convert a scanned input image into a GrayScale image i.e. a monochrome image (Image made up of single color i.e. Gray). Subsequently, we are going to convert grayscale image into binary image by applying Thresholding algorithm to it. After Thresholding, we have to find out the boundaries of that character and we have to crop it if necessary. After cropping we have to remove the noise from that image using Median filter. For noise removal, we have to apply Stentiford thinning algorithm for thinning that input image. After the thinning process we can scale that image to bring it in a proper size template. Now we have to train this template to generate a trained template. All these steps come under preprocessing. The results of some preprocessing steps are shown in Fig 5 and 6.

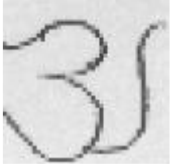


Fig. 5: Original Image

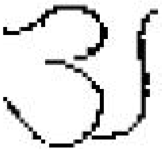


Fig. 6: Binarized Image

A. Gray-scale image

Image is a 2D array or a matrix of pixels. Pixel is the smallest element of an image on a display screen. They can be imagined as a continuous series of square boxes placed on the screen. Pixels are stored as integers of size varying from 8 bit, 24 bit or 32 bit. 24 bit pixels consisting of a combination of 3 colours, viz red, green and blue are most commonly used to depict any image. Many image processing operations work on a plane of image data (e.g. a single colour channel) at a time.

So if u have an RGB image you many need to apply the operation on each of the three image planes and then combine the results. Gray scale images only contain one image plane containing the gray scale intensity values. If u convert an RGB image to gray scale, you would only need to process 1/3 of the data compared to the coloured image. This data reduction saves a reasonable amount of time. Methods like lightness, luminosity and average are used for converting an image into a gray scale image. Average method is the most commonly used method where we compute the average by adding the three colour components of the pixels and divide it by 3. (Gs= r+g+b / 3).

B. Thresholding

Thresholding is one of the simplest methods of image segmentation. Thresholding is performed generally on a gray image to generate binary images. That is, an image with black and white colours only. Thresholding is commonly used to extract essential features. Feature extraction is basically a concept of differentiation between the foreground and the background. The required features of an image are converted to black and everything else to white or the other way around. In thresholding, we compute the binary pixel value of the output based on a formerly defined threshold value. Thresholding is mostly applied on gray scale images, though it can be directly applied on a coloured image as well. But, as mentioned ago, this increases the executing time of the algorithm and slows down the process considerably.

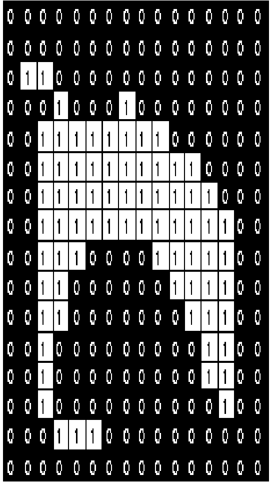
C. Boundary detection and Cropping

Boundary detection or commonly known as edge detection is a method of identifying points in an image at which the image brightness changes in a crisply manner. The points at which image brightness changes suddenly or abruptly are organized into a set, termed as edges. Horizontal and vertical scanners are incorporated which detect the edges from all the sides of the image. Carrying out the edge detection procedure on any image under consideration may substantially lessen the quantity of data to be processed and hence flush out information that perhaps be regarded as immaterial, while keeping the vital structural properties of an image preserved. Thus, the succeeding task of interpreting the information contents in the primary image may therefore be simplified significantly.

Cropping is the simplest of photo manipulation processes, and is performed with the aim to delete unnecessary data or immaterial details from a picture, transform its aspect ratio, or to superiorize the general constitution. Here, cropping is used to remove the non essential details from the photo eventually helping to reduce the processing time.

D. Thinning

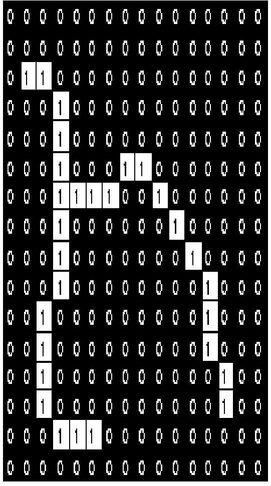
Thinning is basically a peripheral procedure that is used to delete the marked pixels from binary images. Thinning is particularly used for skeletonizing a particular image. It is commonly used to clean up and tidy up the output of the edge detectors by cutting down all the lines to single pixel thickness. Thinning is applied only to a binary image and the consequential image is also a binary image.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	1	1	0	0	0	0	0	0	1	1	1	1	0	0
0	0	1	1	0	0	0	0	0	0	0	1	1	1	0	0
0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Image before thinning

→



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Image after thinning

The Stentiford Algorithm for thinning is described below:

1) Search a pixel location (x, y), where the pixels in the image tally those in template T (A). Using the following template, all pixels placed at the top of the image are deleted shifting from left to right and top to bottom of the image.
2) If the pixel at the centre is not a terminating pixel, and has connectivity no. (CN) = 1, then this pixel is highlighted for removal.
3) Terminating pixel: A pixel is a terminating pixel if it is joined to only one pixel. That means, if a black pixel has only one black neighbour out of the 8 potential neighbours, it can be marked as a terminating pixel.
   Repeat steps I and II for all the pixel locations matching T (A).
4) Replicate steps I to III for the remaining templates: T (B), T (C), and T (D).
5) T (B) will matches pixels on the left side of the image, by shifting positions from bottom to top and from left to right. T (C) chooses pixels along the bottom of the object and shift from right to left and down to up. T (D) addresses pixels on the right side of the image, shifting from up to down direction and right to left.
6) Set to white or black as per mentioned at the beginning, the pixels highlighted to be erased.

E. Scaling

Scaling is the process of resampling or resizing an image into a size which is predefined which coincides with the template with which we are examining that particular image. In image processing, bilinear interpolation is one of the most basic resizing techniques.

It is also known as bilinear filtering in texture mapping, and it can be utilized to produce a sensibly non virtual image. A weighted average of the attributes viz colour, alpha, etc. of the 4 neighbouring pixels is calculated and practically applied to the pixel on the screen. This method is applied recursively for each and every pixel forming the image which is currently being textured.

Each and every pixel of the primary image needs to be shifted in a certain particular direction based on the scale constant, when an image is being scaled up. But sometimes, in up scaling of an image, there are pixels (or holes) that are not allotted proper pixel values, when up scaling is done by a non integral scale value. In such a case, those holes should be assigned appropriate RGB or gray scale values so that the output image does not produce non-valued pixels.

Bilinear method of interpolation can be effectively utilized where exact image transmutation with pixel matching is not feasible, so that one can compute and apport suitable intensity measures to pixels. Dissimilar to other interpolation methods such as bicubic interpolation and closest neighbour interpolation, this method make use of only the 4 closest pixel measures which are placed in diagonally from a given pixel with the goal to evaluate the values of that pixel with appropriate colour intensities.

The method of bilinear interpolation only takes notice of the nearest 2x2 surrounding pixels of the previously known pixel values neighbouring the unknown pixel's calculated position. It subsequently takes a weighted average of the marked 4 pixels to ultimately reach at its end, interpolated value. From each of the known pixel positions, the weight on each of the 4 pixel values is based on the calculated pixel's distance in 2 dimensional spaces.

F. Template Matching

Template matching is a digital image processing method developed for searching minute parts of a picture corresponding a template image. This technique of matching templates can be further divided into the following:
(1) feature-based matching
(2) template-based matching.

If the template image under consideration strongly featured, a feature-based approach may be preferred. Since this method doesn't consider the entire template image, it can be more feasible to compute when working with pictures of enhanced resolution. As an optional method, template-based, may require searching potentially large amounts of points in order to find the best matching location. This approach may further prove very handy if the match in the search image can be transformed in some fashion.

For templates having weak features, or for images where the majority of the template image constitutes the matching image, the template-based technique may be considered. As mentioned before, since template-based matching perhaps require sampling of a massive number of positions, it is possible to lessen the amount of sampling points by decreasing the resolution of the search images and template images by the same factor and carrying on the operation on the resultant images downsized, thus providing a window of points inside the search image, so that the template need not search every workable information or a combination of the two.

V. FUTURE WORK

Several important documents written in 'Modi' language still remain in vegetative state. These documents have priceless data and information. They can be of great help if they are successfully decoded. The problem of Modi OCR and handwriting recognition is a challenging job, and experts try hard to interpret these issues and fabricate potential answers to these issues. A large number of issues still remain to be solved and active research in this area is required to take this potential problem to useful levels, when product using the solution would become available to common man.

VI. CONCLUSION

Here, we have mentioned about the various steps in the image processing techniques in order to convert the Modi characters into English.

REFERENCES

[1] Sakal News Paper(9th July 2014)

[2] D. N. Besekar, R. J. Ramteke, *International Journal of Computer Applications*, vol. 64, no. 3, February 2013. "Study for Theoretical Analysis of Handwritten MODI Script – A Recognition Perspective".

[3] Lawrence Lo, 'ancientscripts.com A compendium of world-wide writing systems from prehistory to today', "MODI","www.ancientscripts.com/modi.html", Accessed 28 March 2014 Accessed 28 March 2014

[4] David Lalmalsawma, India Insights, Reuters, Edition US, 7 Sept 2013. "India speaks 780 languages, 220 lost in last 50 years–survey", "apresearch.org/india-speaks-780-languages-220-lost-in-last-50-years-survey-india-insight", Accessed 28 March 2014

[5] Rajesh Khillari, "History of MODI Script", 30 May 2008, "http://modi-script.blogspot.in/2008/05/history-of-modiscript.html" Accessed 28 May 2014.