A Novel Approach to Skeletonization for Multi-font OCR Applications

C. Vasantha Lakshmi, Sarika Singh, Ritu Jain, and C. Patvardhan

Dayalbagh Educational Institute
cvasantha@rediffmail.com, sarika_singh01@yahoo.com

Abstract. A novel approach to generate skeletons of binary patterns that has a wide variety of applications including multi-font OCR is proposed in this paper. The proposed algorithm ensures connectedness of the pattern and minimizes loss of information while capturing the essential shape characteristics. Computational tests on printed Telugu characters show that the algorithm is useful in getting a generalized form of the character symbols on the common multiple dissimilar fonts.

Keywords: Skeletonization, OCR, Multifonts, telugu.

1 Introduction

Skeletonization plays an important role for the analysis and recognition of binary images e.g. in single and multi-font Optical Character Recognition (OCR). Although, machine printed characters are uniform in size, position and pitch for any given font and are "OCR friendly", multi-font OCR has always been proven to be a difficult problem because of the wide variety in which the characters are written in different fonts [1-4].

But then how do human beings manage to recognize the different fonts and even different people's handwritings? This is because they somehow eliminate the redundant information that the different ornate fonts and handwritings have and recognize the basic structure of the character. Humans do not rely on pixel level features. They somehow capture the "overall" shape of the character. The same ability is necessary for a multi-font OCR engine to succeed in recognizing a variety of shapes. Figure 1 shows the skeleton of a character for two different fonts. It is evident from the figure that the skeleton looks the same in spite of the wide difference in fonts. Once this skeleton is captured it would be easier to build a recognizer that recognizes that the characters are same. An algorithm is proposed in this paper to compute the skeleton of a character.

A good skeletonization algorithm should possess the following properties: (1) preserving connectivity of skeleton, (2) converging to skeleton of unit width, (3) closely approximating the medial axis; (4) possessing insensitivity to boundary noise; (5)

Fig. 1. Skeletons of a telugu character

S. Chaudhury et al. (Eds.): PReMI 2009, LNCS 5909, pp. 393–399, 2009.

achieving high data reduction efficiency. Thinning is the most frequently used method to achieve the skeletonization goal. In the past several decades, many thinning algorithms have been developed and a comprehensive survey is presented in [5].

Some classical algorithms provide good results but still have some deficiencies. For example, some of them cannot preserve the connectedness of an image. In addition, some of these algorithms result in loss of information of the pattern. Huang et al. [6] overcome this information loss by integrating contour and skeleton of pattern. Another method based on connected component approach is proposed by Vijaya Kumar et al. [7]. A simple sequential thinning algorithm for peeling off pixels along contours is described by Govindan and Shivaprasad [8]. One more problem with thinning algorithms is deformation at crossing points. To solve this problem, a knowledge-based thinning algorithm (KBTA), was proposed by Li and Suen [9]. Block Adjacency Graph(BAG) structure is proposed by Suryaprakash[12] .

In this paper, a new skeletonization algorithm using modified Block Adjacency Graph (BAG) structure is proposed which can be used for getting a generalized form of character symbol for multiple fonts. The algorithm presented is not an iterative deletion of pixels. It ensures the connectedness of the image and also minimizes information loss. Although the approach can be used for any character image in general, computational performance on several popular fonts of an Indian script, Telugu, is presented.

The rest of the paper is organized as follows. The proposed approach for skeletonization is described in section 2. Quantitative evaluation is done is section 3. Results on various fonts and sizes are in section 4. Conclusions are derived in section 5.

2 The Proposed Approach

It is very difficult to balance the twin requirements of removal of as many pixels as possible and maintaining connectivity. Thus, the proposed approach relies on finding shape by approximating it as a sequence of carefully constructed segments. BAG can be created by classifying runs as merging, splitting, or continuing runs. The relevant concepts and definitions are as follows.

Horizontal and Vertical Runs-Run Length encoding keeps track of horizontal runs (Hruns) and vertical runs (Vruns) of black pixels, storing the coordinate of the first black pixel and the run length in the corresponding row or column.

Splitting, Merging and Continuous Runs-Hruns are classified by the number of Hruns above and number of Hruns below. If the number of runs above=0 and the number of runs below >1 then the runs are split. If the number of runs below=0 and above >1 then the runs are merged. The remaining Hruns are considered continuous.

Area of block: Number of black pixels in the block.

The proposed method with the help of pseudo-code is given below.

Step1: Isolate the character image say I from the document.

Step 2: Convert I to binary image I' using Otsu's method [10].

Step 3: Create horizontal and vertical runs of the image I' as shown in Figure 2(c).

Step 4: Mark runs as splitting, merging and continuous as defined above (Figure 2 (d)).

Step 5: Construct blocks as shown in Figure 2 (e):

(a) Merge all adjacent continuous runs to form a block till a split or merge run is encountered.

(b) Merge this split/merge run into the block if it is directly above or below the block so formed else start a new block.

Step 6: Compute centroids of each of the blocks as shown in Figure 2(f).

Step 7: To remove the spurious blocks/pixels which may result in skeletal legs, a block is removed if both the following conditions are true.

(a)Area of block is less than average size where average size is computed as

$$Avg\ .Size = \frac{\sum_{i=1}^{n} Area\ (b_i)}{n}$$

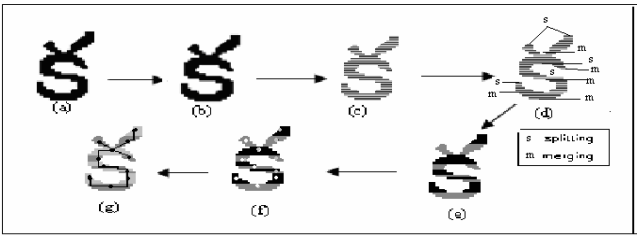
,where b_i denotes block i and n is the total number of blocks.

(b) The block is adjacent to only one block and number of black pixels on the path connecting the two centroids is less than k. Value of k depends on average size of character image.

Step 8: Connect the centroid points preserving the path as observed from the original character image as follows.

(c) If two adjacent centroid points, if they lie in same column or row, and path between them corresponds with original image, all pixels connecting them in that row or column are made black.

(d) If two adjacent centroid points do not lie in same row or column the connections as shown in Figure 3 can occur. So the path is computed using these two centroid points and the actual path present in the original image between them.



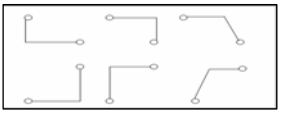


Fig. 3. Connecting paths

Fig. 2. Steps in proposed algorithm (a) Isolated character (b) After Otsu's thresholding (c) Horizontal runs and vertical runs images (d) runs marked as splitting, merging and continuing (e) Creation of blocks (f) computation of centroids (g) connecting centroid points

The identification of splitting, merging and continuous runs (step 4), conditions for noise removal (step 7) and connecting the centroids (step 8) are different.

3 Quantitative Evaluation and Experimental Results

In this section, a set of measures is defined to evaluate the performance of the proposed algorithm. Connectivity was first examined. Then, measures are computed to evaluate i) how closely the resulting skeleton approximates the true medial axis of the object, ii) how well the pattern converges to the desired one pixel-wide skeleton, and iii) how sensitive the resulting skeleton is to boundary noise. Finally, the data reduction efficiency is measured to evaluate the efficiency of the algorithm.

3.1 Connectivity

A connectivity algorithm is used to check that skeleton contains only one connected component [11]. In the proposed algorithm, since the connection of centroids of adjacent blocks is established following the possible paths; the skeleton is always connected.

3.2 Measure of Convergence to Unit Width

If the converged skeleton S_M does not contain any one of the patterns $\boldsymbol{Q^k}$ as shown in Figure 4, then S_M is one pixel wide. To measure the width of the resulting skeleton, m_t is defined as $m_t = 1 - \frac{Area\left[\bigcup_{1 \le k \le 4} S_M Q^k\right]}{Area[S_M]}$ where Area is the operation that counts the number of black pixels. This measure has a nonnegative value less than or equal to 1, with $m_t = 1$ if S_M *is* a perfect unit-width skeleton.

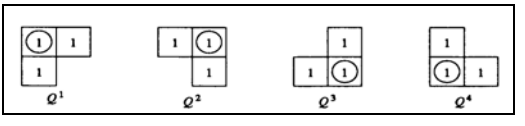


Fig. 4. Templates Q^k used to examine the width of the convergent skeleton

3.3 Measure of Medial Axis Representation

To evaluate the amount of variations between the skeleton and its ideal medial axis, a measure is defined as a function of the input image S and its resulting skeleton S_M. At each point $p \in S_M$, the maximum digital disk included in S and centered at p is denoted as $DD(p,r_p)$, where r_p represents the radius of a continuous disk whose digital image of the disk is DD centered at p. Note that DD(p, 0) = {p}by convention. Then, we have $S' = \bigcup_{p \in S_M} DD(p, r_p)$, where $S' \subseteq S$. Hence, the measure m_m, can be defined as $m_m = \frac{Area[S']}{Area[S]}$.

This measure has a nonnegative value less than or equal to 1 with equality if and only if the resulting skeleton contains the discrete medial axis.

3.4 Measure of Boundary Noise Sensitivity

Let S be the given noise-free binary image and S'' be its noisy version generated by randomly adding and subtracting k unit-size points along the boundary of S. We define the signal-to-boundary noise ratio as

$$SBNR_k = \frac{Area[\partial S]}{Area[S''/S] + Area[S/S'']}$$,where ∂S denotes the boundary of S. The terms S''/S and S/S'' are the set differences between the ideal image and noisy image, respectively. Thus, the error introduced by boundary noise at a particular $SBNR_k$ can be measured by the normalized quantity

$m_e(SBNR_k) = \min(1, \frac{Area[S_M / S''_M] + Area[S''_M / S_M]}{Area[S_M]})$,where S_M and S''_M are the resulting skeletons of S and S'', respectively. The measure is normalized between 0 and 1; a highly noise-sensitive algorithm will yield an m_e close to 1.

3.5 Measure of Computational Cost

A measure to evaluate both the data reduction efficiency and the computational cost is defined as $m_d = \min\left[1, \frac{Area[S] - Area[S_M]}{Area[S]}\right]$. This measure has a value between 0 and 1; a large value indicates high efficiency. The quantitative measures described above were used to evaluate the proposed algorithm and compared with parallel thining algorithm[13] on sample skeleton shown in figure 5.

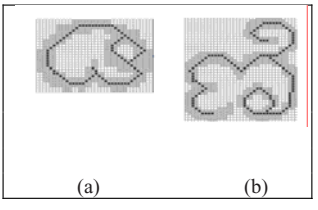


Fig. 5. Skeletons obtained by proposed algorithm

Table 1. Comparison of parallel thinging algorithm and proposed algorithm

Quantitative evaluation	Parallel thining Algorithm of fig. 5(a)	Proposed algorithm of fig. 5(a)	Parallel thining Algorithm of fig.5 (b)	Proposed algorithm of fig. 5(b)
m_t	0.99	0.99	0.99	0.99
m_m	1.00	0.99	0.99	1.00
m_d	0.77	0.80	0.82	0.84

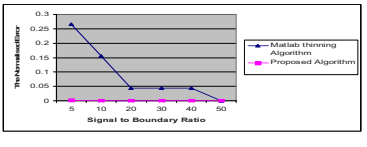


Fig. 6. Effect of boundary noise

Fig. 7. (a) Telugu character Ta (b) Skeleton of Ta (c) Telugu character lu (d) Skeleton of lu

In another experiment, a random boundary noise was generated at various (signal boundary ratio) $SBNR_k$ levels. The experiment is summarized by plotting $SBNR_k$ against m_e in Figure 6.

4 Results on Different Telugu Fonts and Sizes

Telugu character samples are prepared using 3 fonts- Godavari, HarshaPriya and Hemlata and three font sizes: 25, 30, and 35. Figure 7 shows two confusingly similar characters 'Ta' and 'lu' in Godavari 25 and their skeletons. An experiment is performed using density features on 3x3 grids with Euclidean distance measure for computing the distance. Figure 7(a) is the base character for comparison in columns 3 and 4 while Figure 7(b) is for columns 6 and 7 of Table2. Comparing columns 5 and 8 depicting ratio's, we see that the distinction between confusingly similar characters is more prominent with skeletonized characters and as a result is less prone to errors in recognition.

Table 2. Distance of skeletonized and unskeletonized character 'Ta' (font Godavari and size 25) with character 'lu' of different fonts and sizes and with same character of different fonts and sizes

Fonts	Sizes	Distance of 7(a) with unskeletonized		Ratio (2)/(1)	Distance of fig.7 (b) with skeletons of		Ratio (4)/(3)
		'Ta' (1)	'lu'(2)		'Ta'(3)	'lu'(4)	
Godavari	25	0	0.504	-	0	0.116	-
	30	0.208	0.39	1.87	0.033	0.1396	4.23
	35	0.1519	0.429	2.82	0.0484	0.1269	2.62
HarshaPriya	25	0.2394	0.456	1.90	0.056	0.1023	1.82
	30	0.2912	0.4857	1.66	0.047	0.095	2.02
	35	0.3433	0.542	1.57	0.032	0.1104	3.45
Hemlata	25	0.2617	0.4616	1.76	0.0273	0.1212	4.43
	30	0.3042	0.4151	1.36	0.0509	0.1318	2.58
	35	0.27	0.454	1.68	0.043	0.1296	3.01

5 Conclusion

Multi-font OCR has proven to be a tough problem, especially for Indian scripts because of the complicated scripts and the large variety of ways in which the characters are written in different fonts. One of the approaches to tackle this is to reduce the characters to their basic structural forms. In this paper, a method is proposed to do this utilizing a modified BAG data structure. Examples are provided to show that the method indeed amplifies the dissimilarity between different characters and the similarity between same characters in different fonts. This is conducive to better downstream operations for OCR like feature detection and recognition. Efforts are on to develop a complete OCR engine for multi-font Telugu OCR based on this simplification approach.

References

[1] Chan, C., Wong, P.: A branch and bound decision tree Bayes classifier for robust multi-font printed Chinese character recognition. In: IEEE Region 10th International Conference, November 11-13, vol. 1, pp. 267–271 (1992)
[2] Ben, A.N.: Multifont Arabic Characters Recognition Using Hough Transform and HMM/ANN Classification. Journal of Multimedia 1(2) (May 2006)
[3] Ho, T.K., et al.: A Computational Model for Recognition of Mutifont Word Images. Machine Vision and Applications 5, 157–168 (1992)
[4] Wang, J., Jean, J.: Resolving multifont character confusion with neural networks. Pattern Recognition 26(1), 175–187 (1993)
[5] Lam, L.S.W., Suen, C.Y.: Thinning Methodologies-A Comprehensive Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(9), 869–885 (1992)
[6] Huang, L., Genxun, W., Liu, C.: An Improved Parallel Thinning Algorithm., Proceedings of the Seventh International Conference on Document Analysis and Recognition, ICDAR (2003)

[7] Kumar, V.V., Srikrishna, A., Shaik, S.A., Trinath, S.: A New Skeletonization Method Based on Connected Component Approach. IJCSNS International Journal of Computer Science and Network Security 8(2) (February 2008)
[8] Govindan, V.K., Shivaprasad, A.P.: A pattern adaptive thinning algorithm. Pattern Recognition 20(6), 623–637 (1987)
[9] Li, B., Suen, C.Y.: A knowledge-based thinning algorithm. Pattern Recognition 24(12), 1211–1221 (1991)
[10] Otsu, N.: A Threshold selection method from Grey-level histograms. IEEE Transactions on Systems, Man and Cybernetics SMC-9(1) (January 1979)
[11] Efford, N.: Digital Image Processing. A practical introduction using java
[12] Kompalli, S.: A stochastic framework for font-independent Devnagari OCR. P.hd. thesis, SUNY Buffalo (January 2007)
[13] Guo, Hall, R.W.: Parallel thinning with two-subiteration algorithms. Comm. ACM 32(3), 359–373 (1989)