AN EFFICIENT THINNING ALGORITHM FOR ARABIC OCR SYSTEMS

Mohamed A. Ali

Department of Computer Science, Sabha University, Sabha, Libya
fadeel1@sebhau.edu.ly

ABSTRACT

This paper address an efficient iterative thinning algorithm based on boundary pixels deletion using colour coding for different pixel types. A black pixel is tested by observing neighbouring pixels, and it gives us an efficient way to decide whether the pixel is deleted or not. In the propose algorithm number of 3x3 templates were used to make good deleting decision, then we delete the pixels which satisfy the deletion templates until there is no pixel that can be deleted. Other templates were used for discontinuity recovery. This algorithm allows us to deal with typical troublesome handwritten text efficiently. Without smoothing before thinning, the algorithm produces robust skeleton even in the presence of noises. The algorithm produces skeletons that are more representative of the shape of the original patterns and with less noise spurs. The algorithm is considered fast enough to be used in Arabic OCR systems.

KEYWORDS

Arabic handwriting, optical characters recognition (OCR) systems, , thinning algorithms.

1. INTRODUCTION

Thinning plays a major role in OCR system, and since recognition is dependent in part on the effectiveness of the thinning algorithm, attention is given in this paper to the development of effective thinning algorithm for the purpose of developing an Arabic OCR system. Character recognition is a field of pattern recognition that has been subjected to considerable work during the past four decades [1]. Although the designing of thinning algorithm has been an important research area, merely number of researchers have considered designing of reliable thinning algorithm for Arabic writing [1,3].

In general, an effective skeletonization algorithm should ideally remove all redundant pixels and retain the significant aspects of the pattern under process [2]. The resulting set of lines and curves is called the skeleton of the object. Good algorithm should fulfill some requirements namely;

1. Skeleton connectivity should be preserved.
2. Thinning to the approximated medial axis of the original image.
3. Excessive erosion should be prevented, i.e. end points of a skeleton should be detected as soon as possible so that the length of a line or curve that represents a true feature of the object is not shortened excessively.
4. The skeleton should be immune to noises. Noise, or small convexities, which do not belong to a skeleton, will very often result in a tail after thinning.
5. The algorithm output should be a skeleton of unity pixel width.

DOI : 10.5121/sipij.2012.3303

One of two approaches has, commonly, been followed in most of thinning algorithms, the iterative approach and noniterative approach [2-7]. In the iterative approach, pixels on the boundary are examined (either in sequential or parallel) and successively deleted until a skeleton of one pixel width is obtained. On the other hand noniterative approach produces a medial line of the original image (in one pass) without the need of examining all pixels individually.

In the proposed algorithm we follow the iterative approach, and a color coding is used in bitmap file of sixteen colors to mark, examine, preserve, delete and recovering pixels to achieve thinning and solve the problem of discontinuity yielding a very fine skeleton of the original image of Arabic handwritten text.

2. THE ALGORITHM PROCEDURE

Our algorithm utilizes a windows color bitmap file format. Six codes were chosen to represent on-pixel (black), off-pixel (white), noise pixel, start or end point pixel, deletable pixel and recovered pixel. The algorithm needs to follow five main steps to achieve the task of skeletonization and they are as follows:

2.1. Start and End points marking

This is done by scanning the whole image from top-left to bottom-right corner allocating all pixels in inner and outer boarder of the image and distinguish those deletable from undeletable pixels.

For undeletable pixels, the algorithm consider all on-pixels which surrounded by six or seven off-pixels (in directions according to the Freeman's code diagram shown in Figure 1) are undeletable. These pixels are expected to be a start or end points on the image and, hence, must be preserved for sake of image shape preservation and they should not be examined in all iterations come afterward as shown in Figure 2.

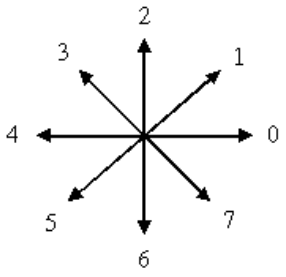


Figure 1 Freeman's chain Code

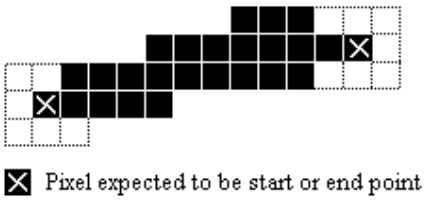


Figure 2 Start and end points detection

In the same manner, algorithm consider all black pixels which surrounded by five or eight white pixels are noise and then delete them as shown in the Figure 3-a and 3-b.

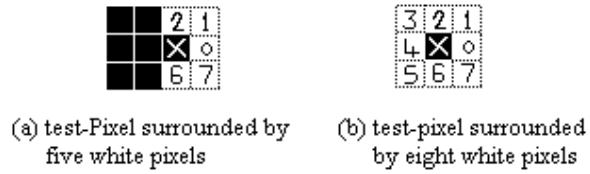


Figure 3 Pixels that considered as noise

2.2. Allocation of Deletable Pixels

In this step we need to allocate all pixels on the boundary of the image that can be deleted for the sake of thinning. Allocation of these pixels should follow the rules (template) shown in Figure 4.

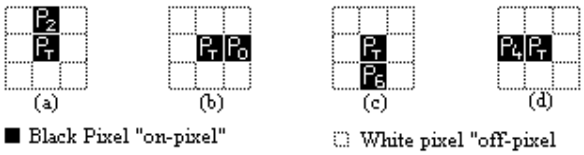


Figure 4 Templates for allocation of deletable pixels

Where P_T is a pixel under test and P_0, P_2 , P_4 and P_6 are the four neighbor pixels of P_T in four directions according to Freeman's Code. The conditions that make P_T deletable are as follows:

If {(P_2=on) & (P_6 =off) or

(P_0 =on) & (P_4 =off) or

(P_2 =off) & (P_6 =on) or

(P_0 =off) & (P_4=on)}

So P_T in all four, above mentioned, cases is deletable pixel provided that it should be connected to at least two other black pixels. Subsequently they will be mark first as deletable pixels, and later the algorithm will decide whether to delete them or not according to the conditions fulfillment.

Now to avoid discontinuity there are three more rules to apply before start deleting all pixels marked as deletable pixels:

a) The first rule is set to avoid discontinuity by making sure that all deletable pixels are not following any of patterns shown in the Figure 5.

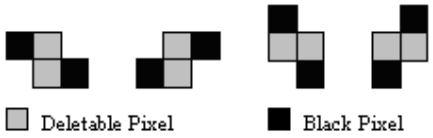


Figure 5 first rule for discontinuity prevention

If any of deletable pixels do fall under any of patterns shown in Figure 5, one of deletable pixels should be retained. The priority of retaining a pixel goes to the deletable pixel which has more other deletable pixels connected to it than the other. However, if both of deletable pixel have the same number of other deletable pixel the priority goes to the one which leads the other according to the direction of image scanning from top-left to bottom-right. As a result, that pixel is marked as undeletable pixel.

b) The second rule states that if a deletable pixel connected to another three deletable pixels in a manner shown in Figure 6-a, the algorithm marks the medial pixel as a black pixel as shown in Figure 6-b.

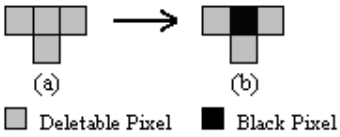


Figure 6 Second rule for discontinuity prevention

c) The third rule states that any pixel which has been marked as deletable and has two white pixels at direction of (P_2 & P_6) or (P_0 & P_4) as shown in Figure 7 should be reverted to black pixel.

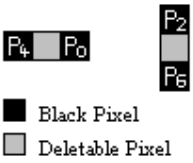


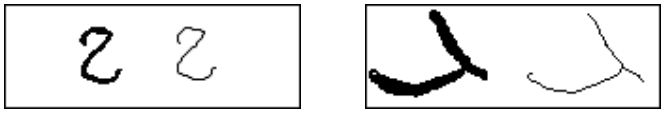
Figure 7 Third rule for discontinuity prevention

2.3. Deletion Process

We shall now delete all pixels that still marked as deletable pixels. Deletion follows the scanning of the image from top-left corner to bottom-right corner. As a result of this deletion we have noticed that some discontinuities have occurred and hence we make the algorithm finish this process without any interruption and make it iterate as described in the next section till there are no more pixels to be deleted (in other word the number of deleted pixels after each iteration is same). Only then the algorithm starts checking for discontinuities and suggests proper connections.

2.4. Iteration

The algorithm now will iterate repeating step-2 and step-3 till there are no more deletable pixels to delete. In other word the templates in Figure 4 are no longer applicable. The number of iterations depends mainly on the thickness of the handwriting in the input image. For instance the handwritten character (ha), shown in Figure 8-a, took five iterations to reach its final skeleton whereas character (dal), shown in Figure 8-b, took six iterations.



(a) (b)

Figure 8 two Arabic handwritten characters of different thickness and their skeletons

2.5. Discontinuity Deletion and Recovery

In case of any discontinuities in one place or another in the output skeleton, we propose a technique involves recovering of those deleted pixels which cause this type of discontinuity as following:

We move a window of 3x3 on the whole thinned image and if one of the templates shown in Figure 9 was found, we check the missed pixel so that if it is proved that this pixel was there and, because of thinning algorithm, has been deleted we just recover that pixel back (make it black pixel), hence the problem of discontinuity is solved, otherwise we shall consider that as a deliberate discontinuity (i.e. is one of the character feature) and keep it as it is.

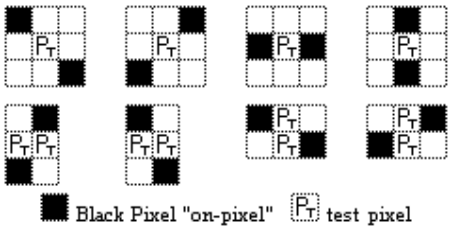


Figure 9 Templates for recovery of deleted pixel and preserve connectivity

Referring to Figure 9, P_T is a pixel to be checked whether it was there before applying the algorithm or not, so if it was there we just convert this pixel back to black pixel otherwise we leave it as it is.

Solving this type of discontinuity does not prevent other type of discontinuity from occurring like the one shown in the Figure 10 where none of those templates is applicable and the length of discontinuity is more than two pixels and that is notably happened in the line or stroke which inclined diagonally in the direction of P_3 or P_7 (i.e. lines goes to North-West or South-East)

Figure 10 Type of discontinuity with more than one pixel long

In the Figure 10 we can clearly notice (from left to right) original image of Arabic character (LamAleef), skeleton with discontinuity and skeleton with discontinuity being recovered.

The measures taken to recover this type of discontinuity is as follows: the algorithm sweep the whole image skeleton looking for those black pixels which are connected to one black pixel only

(excluding those pixels marked as start and end point pixels) and check its neighbor at P_3 or P_7, so if the tested pixel connected to either P_3 or P_2 and that P_7 is white and it was black before deletion then P_7 is converted back to black, likewise if the tested pixel connected to either P_6 or P_7 and that P_3 is white and it used to be black before deletion then P_3 is converted back to black. Figure 11 illustrates this mechanism. This mechanism is repeated till there are no more pixels (excluding start and end point pixels) connected to one black pixel only. In this way it is verified that our algorithm is effectively capable of solving this type of discontinuity

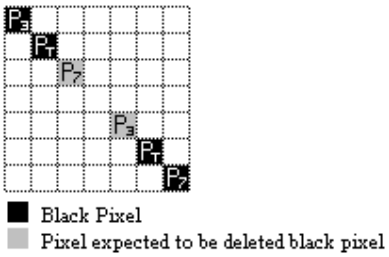


Figure 11 Mechanism applied for discontinuity of more than one pixel long

3. EXPERIMENTS AND RESULTS

The algorithm was tested on different Arabic handwritten text in both cases discrete and cursive using hp-scanner (with 1200 bpi resolution) for image capturing. A preserved smooth skeleton was obtained. Figure 8 and Figure 12 show examples of tests carried out on different Arabic handwriting images along with their output skeletons. Figure 12 clearly shows how a skeleton of an image has a shape reserved, smooth, intermediate and one pixel width line of the original image when we superimpose them.

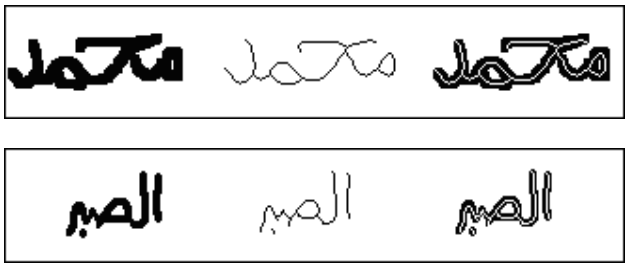


Figure 12 samples of original Arabic handwritten images and their skeletons

4. OPTIMIZATION

To confine the algorithm to a minimum number of pixels for testing in each iteration so that we reduce the run-time and make it faster, we made the algorithm (in the first scan) assign the location of first and last black pixels found as pixels of origin so that for the next iterations the algorithm starts and ends at these pixels rather than scanning the whole image area as defined by BitMap file format.

On the other hand , to avoid inefficient iteration the algorithm is designed so that the process of deletion (thinning) is stopped and final output image (skeleton) is saved when either there are no more pixels to delete or the number of deleted pixels in two successive iteration are same, subsequently the excessive iterations are avoided and program run-time is minimized. .

5. CONCLUSIONS

The main objective of this brief is to develop an accurate thinning algorithm for Arabic characters to be used in Arabic character recognition system. A sequential iterative thinning algorithm is presented in this paper. The algorithm has used Six codes to represent on-pixel (black), off-pixel (white), noise pixel, start or end point pixel, deletable pixel and recovered pixel. In the propose algorithm number of 3x3 templates were used to make good deleting decision, the algorithm deletes the pixels which satisfy the deletion templates until there is no pixel that can be deleted. Other templates were also used for discontinuity recovery. The algorithm was tested on different Arabic handwritten in both cases discrete and cursive. The algorithm allows us to deal with typical troublesome handwritten text efficiently, and produces robust skeleton even in the presence of noises. The algorithm produces skeletons that are more representative of the shape of the original patterns and with less noise spurs. The algorithm is considered fast enough and very applicable to be used in Arabic OCR systems.

ACKNOWLEDGEMENTS

The authors would like to thank Sabha University administration for its fully support in form of moral and financial support, without which I couldn't have finish this research and publish it.

REFERENCES

[1] Supriana, I.; Aryan, P.R., (2011), "Direct skeleton extraction using river-lake algorithm ," *International Conference on Electrical Engineering and Informatics (ICEEI),* pp. 1–3

[2] Rafael C. Gonzalez & Richard E. Woods, (2007), *Digital Image Processing (3rd Edition)* Prentice Hall

[3] Al-nuzaili, Q.; Mohamad, D.; Ismail, N.A.; Khalil, M.S., (2012) "Feature extraction in holistic approach for Arabic handwriting recognition system: A preliminary study", *IEEE 8th International Colloquium on Signal Processing and its Applications (CSPA),* pp 335 - 340.

[4] Lei Haijun; Zhang Panpan; Li Xianyi, (2010) "The Application of an Improved Thinning Algorithm in Numeral Recognition System", *International Conference on Multimedia Technology (ICMT)*, pp. 1 – 3

[5] Le Zhang; Qing He; Ito, S.-I.; Kita, K., (2010) "Euclidean distance-ordered thinning for skeleton extraction", *2nd International Conference on Education Technology and Computer (ICETC),* Vol. 1, pp 311-315

[6] Bag, S.; Harit, G., (2010) "A medial axis based thinning strategy and structural feature extraction of character images", *17th IEEE International Conference on Image Processing (ICIP),* pp. 2173 – 2176.

[7] Azeem, S.A.; El Meseery, M., (2011), "Arabic Handwriting Recognition Using Concavity Features and Classifier Fusion", *10th International Conference on Machine Learning and Applications and Workshops (ICMLA),* Vol. 1, pp. 200 – 203.

Author

Mohamed Ali received the BSc in Electronic & communication in 1984 from Tripoli University. In 1993 he received his MSc. From Nottingham University in Computer Engineering; in 2005 he got his PhD. degree in computer science from UKM - Malaysia. Since then he hold the head of computer department in Sabha University. He is Member of ICT committee in Sebha University, IEEE member, Member of the Centre for Quality Assurance & Accreditation of Educational Institutes and Member of general assembly of Libyan Olympiad of information. His research activities have been in the areas of optical character recognition and related problems of document processing and Neural Networks