JKAU: Sci., vol. 7, pp. 119-130 (1415 A.H. / 1995 A.D.)

An Experimental Approach for Recognizing Handwritten Arabic Words*

KAMAL M. JAMBI
Department of Computer Science, Faculty of Science, King Abdulaziz University, Jeddah, Saudi Arabia

ABSTRACT. This paper discusses the process of implementing an off-line system for recognizing handwritten Arabic words. In order to recognize a word, its character decomposition should be known. This is done through segmentation. In our model, Arabic character recognition goes through a preprocessing stage followed by a recognition stage. Each character of the word is investigated in order to determine its features associated with the window number in which they are located. The steps taken for obtaining the window frame and windows as well as those features used are elaborated. A table lookup is used to determine the name of the character under consideration. This is followed by a discussion of the results and their interpretations. A comparison of the results obtained with other related work is given.

1. Introduction

Arabic character recognition is an active field of current research. Researchers these days strive to achieve better speed and higher accuracy for character recognition leading to a communication interface between the computer and its users. This implies direct storage of user handwritten documents into computer memory without going through a keyboard.

The Arabic language is the main language in more than 20 countries in the world and spoken by more than 200 million people. It is not spoken by Arabs only, it is also taught to all Muslim people. There are several properties of Arabic characters that

*A summary of this work was presented in Arabic, The Fourth Saudi Engineering Conference, Jeddah. Nov., 1995, Vol. 3, pp. 69-75.

give handwriting style uniqueness and cause more difficulty in recognition. The writing direction goes from right to left. Since the characters are cursive (i.e., characters of a single word within any text are connected) it implies that boundaries of these characters overlap. It is interesting to note also that some groups of characters have the same main body with slight changes. These changes are presented by the number of dots and their relative position with respect to the main body. For example, Table 1 shows that characters Baa, Taa, and Thaa have the same main stroke but differ in the number of dots as well as their position. Therefore, Baa has a single dot below the main body while Taa and Thaa have two and three dots above the main body respectively. Moreover, Arabic characters have different shapes depending on their location within a word (i.e. isolated, or occuring at the beginning, middle or end of the word) as shown in Table 1. Therefore, rather than dealing with the 28 characters of the Arabic alphabet, researchers must deal with more than 60 characters.

TABLE 1. The different shapes of Arabic characters depending on their Position Within the Word.

Name	Isolated	First	Middle	Last	Name	Isolated	First	Middle	Last
Alif	ا	ا	ا	ا	Dhad	ض	ضـ	ـضـ	ـض
Baa	ب	بـ	ـبـ	ـب	Tta	ط	طـ	ـطـ	ـط
Taa	ت	تـ	ـتـ	ـت	Zha	ظ	ظـ	ـظـ	ـظ
Thaa	ث	ثـ	ـثـ	ـث	Ain	ع	عـ	ـعـ	ـع
Geem	ج	جـ	ـجـ	ـج	Ghain	غ	غـ	ـغـ	ـغ
Hha	ح	حـ	ـحـ	ـح	Faa	ف	فـ	ـفـ	ـف
Kha	خ	خـ	ـخـ	ـخ	Gaf	ق	قـ	ـقـ	ـق
Dal	د	د	ـد	ـد	Kaf	ك	كـ	ـكـ	ـك
Thal	ذ	ذ	ـذ	ـذ	Lam	ل	لـ	ـلـ	ـل
Raa	ر	ر	ـر	ـر	Meem	م	مـ	ـمـ	ـم
Zain	ز	ز	ـز	ـز	Noon	ن	نـ	ـنـ	ـن
Seen	س	سـ	ـسـ	ـس	Haa	ه	هـ	ـهـ	ـه
Sheen	ش	شـ	ـشـ	ـش	Waw	و	و	ـو	ـو
Saad	ص	صـ	ـصـ	ـص	Yaa	ي	يـ	ـيـ	ـي

In order to get familiar with the approaches used to recognize Arabic characters, the reader may refer to Jambi[1,2]. The scope of the surveyed work varies, with respect to complexity, from printed isolated characters to handwritten words. This implies that evaluation of the work based only on the recognition rate is not fair and the reader may be misled. For instance, it is not fair to compare a 100% recognition rate for isolated printed characters and 90% recognition rate for handwritten cursive words. In Jambi[2] there are also some details discussing different techniques used in other languages. This might suggest concepts for researchers to implement with respect to Arabic characters.

Although several papers deal with printed Arabic characters and texts, very little research has been done regarding handwritten Arabic words. The approach adopted in the work of Jambi[3], which was also implemented by the author of this work, used a rectangular frame with six windows. Moreover, in the work of Nouh *et al.*[4] the authors stated that a circular frame is the best shape to contain Arabic characters. Therefore, the frame in this work was changed to a circular one and many tests were performed to figure out the best arrangement of the window to be accepted.

After the image was obtained with a scanner, the operations of the preprocessing stage are applied. These operations include finding the boundaries of the word, thinning, and separating the overlapped subwords. The main section of the paper discusses the process of recognition which begins by decomposing the word into its characters. Then each of these characters is investigated to identify its feature points and the location of each within the seven windows of the frame that contains the character under consideration.

2. Preprocessing Operations

The image of the isolated word is obtained by means of a scanner, where noise elimination is not needed and the binary image can be obtained automatically. The extreme boundaries of a word are used to simplify the operation by eliminating the process of scanning the white area around the word. Therefore, having the absoltue correct values of the extreme boundaries is not a critical issue, which makes the situation flexible and not sensitive to the noise. These boundaries are used also to identify the constraints of upper and lower limits for the image of each character.

The One Pass Thinning Algorithm[5] is used for thinning. It starts by scanning the image frame in a raster fashion using a sliding template. If the scanned pixel is black, this pixel and its neighbors are treated as a template that should be compared to other predetermined templates. Whether the scanned pixel is removed (i.e., a 1 becomes a 0) or kept depends on the results of the comparison with any one of the templates for testing pixel removal. Another test takes place by comparison with templates used for connectivity testing. In the case of a match, the removed pixel has to be restored. This step should be repeated until no change is produced.

Separation of overlapped subwords has generally been ignored in previous work, although the overlapping of adjacent characters exists naturally. Overlapping means that the beginning of a character starts on a column located before the one where the end of the previous character has been detected. Although, Almuallim and Yamaguchi[6], however, took care of this problem implicitly by tracing continuous strokes, erasing them from the original image, then considering the other strokes. In this work, it is essential that overlapping should be resolved. This is because the histogram for the segmentation process depends heavily on this operation. Figure 1 shows how this work deals with this problem explicitly by shifting some strokes and making sure that there is at least one blank column between overlapped strokes. In fact, separation of overlapped subwords is not as easy as it may initially appear. It is time consuming, since all strokes should be traced and assigned different labels. Dots should be assigned to their main strokes by giving those dots the same label as the main stroke. Left and right boundaries should be determined in order to test for overlapping. If that is the case, shifting should be done for the left stroke(s) as well as all dots and secondary strokes associated with those strokes. Shifting is done by having at least one blank column (i.e., no black pixel) between two adjacent strokes of overlapped subwords. It should be mentioned that shifting will take place provided that enough space is available to accommodate the process of shifting[2].

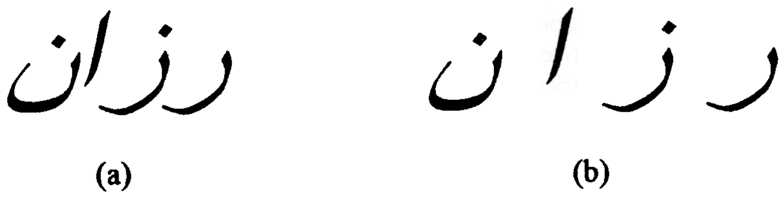

FIG. 1. An Arabic word with overlapping (a), and the same word after overlapping is resolved by a shifting process (b)[2].

3. The Process of Word Recognition

The process of word recognition starts by the decomposition if the isolated word into its character components where each of these characters is investigated separately.

3.1 Segmenting the Word into Separate Characters

The characters used to construct the handwritten word should be identified. The work of Jambi[7] gives a complete description of our approach to implementing this operation. The operation starts by constructing a histogram done by counting the number of black pixels in each column. The beginning of each character is identified by sensing a sudden change in the histogram, however some difficulties should be resolved regarding different width sizes of Arabic characters. Therefore, a threshold value is determined to overcome this problem.

This histogram, which is known also as an image projection, produces an array that contains the number of black pixels counted column wise. The last array is then processed in order to identify some interesting points. These points are either actual start or end points of a character (represented by 's' and 'e' accordingly). Other points are represented by 'b' or 'c' indicating candidate begin or end points of a character. The 'c' is replaced by 'f' if it is recognized to be a permanent end point. These point can also be displaced in order to eliminate the effect of the long connecting stroke that can be used for connecting those characters. Also, these points may be removed if it is discovered that they are no longer needed[7]. There are some tests that should be done regarding the different forms of the last character of each word. At the end of this procedure, the above points are used to identify the beginning and the end of each character of the word.

3.2 Defining the Features to be Considered

Although there are many features that can be considered, the following features are the most suitable ones for the structural approach adopted in this work

- ***Branch point***

Represented by a black pixel surrounded by at least three black pixels.

- ***Corner***

This point represents a change in the direction of the pen's movement (decided if the measurement of the angle, between both lines of each direction, exceeded a pre-determined threshold value based on the work of Han *et al.*[8].

- ***End point***

This point represents the start or stop point (i.e., a point where a pen starts or stops writing the current stroke). This can be found easily within the image, since this is the only point (black pixel) with just one black pixel neighbor.

- ***Position***

An integer for the position of the character. It takes one of the following values

1 for isolated characters,
2 for those connected from left,
3 for those connected from right, and
4 for those connected from both sides.

- ***W-H***

This variable indicates the relationship between the width and the height of the character. Therefore, the set of characters are classified into three classes. The first class includes those characters whose length is twice the width or more such as (Alif and Lam). The second class contains those characters whose width is twice the length or more such as (Geem-first and Taa-isolated). The third class indicates those characters where the above relations can not be identified.

- ***Loop***

This integer variable gives the window number where the loop is located. It gets the value '0' if no loop is detected. This is because the handwriting characters contain at most one loop.

- ***Dots***

This variable indicates the number of dots associated with the character. It should be mentioned at this point that although the result of thinning shows dots as short strokes, they are removed in the segmentation process[7]. Those dots are then re-placed by single pixels at the center of the previous short strokes.

Therefore, after finding all features of the character under consideration, a vector (or a database entry) will be constructed[2]. For each window a byte is allocated where two bits are assigned to store the number of end points, branch points, corners and dots that are identified in that window. Since we have seven windows, (see discussion below) seven bytes are needed. An extra byte is also required to store the other features such as position, W-H and loop.

3.3 Identifying the Window Frame and Windows

The process of finding the coordinates of a rectangular frame is a simple one. This is done by scanning the image of the character from top, down, left, and right to find out the locations of the first pixel from each direction. Notice that the extreme boundaries, which have been calculated previously for the whole word, can be used as constraints. Therefore, determining those coordinates gives us the ability to identify the window frame as well as the windows where feature points are located as shown in Fig. 2. The point (a, b) indicates the center of the circular frame and r presents the radius.

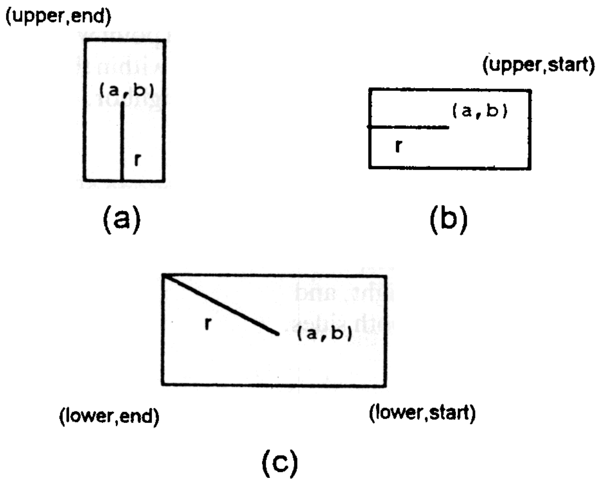

FIG. 2. The rectangular frame is used to identify the center (a, b) and the radius (r).

As shown in Fig. (2a), this frame suits those characters whose height is greater than their width while the frame in Fig. (2b) suits characters whose width is greater than their height. However, the frame in Fig. (2c) is selected to be able to capture features that might be located at the corners of the frame, which might be missed if the other frames are selected.

The number of windows plays a significant role here. A small number of windows gives an overlapping of features of different characters. On the other hand, a large number may cause difficulties in identifying small variations of the same character. For instance, if the window is of the size of a single pixel then the situation becomes an exact pixel matching. A single window complicates the process of classification, since different characters with the same feature points are put in the same group. After a careful study of Arabic characters, it seems that the best results are obtained by diving the frame into seven windows as shown in Fig. (3).

The window's frame has a circular shape where the center is (a, b) identified as the following :

$$a = (\text{upper} + \text{lower}) / 2,$$

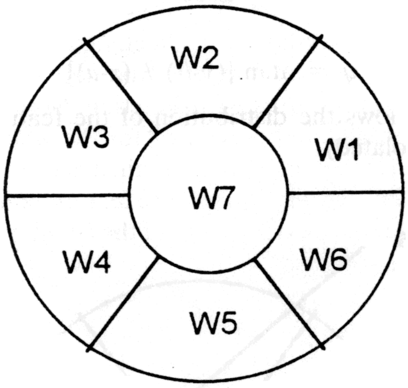

FIG. 3. Dividing the frame into seven windows.

It should be mentioned at this point that the origin of the image (i.e. the pixel (0,0)) is located at the upper left corner. The radius of the frame is defined as shown in Fig. 2 where :

$$r = \text{start} - b$$

for the cases where the length of the character is greater than the width as the case of Fig. (2a), and

$$r = \text{lower} - a$$

for the cases where the width is greater than the length as the case of Fig. (2b). However, these situations are discovered not to be very accurate. Especially, for those characters where some features are located on the corners of the rectangular frame such as (Lam) and (Khaa-isolated), etc. Therefore, the radius should be calculated with a different approach. So, the length of the radius is computed as the distance from the center to the corner of the rectangular frame as shown in Fig. (2c).

$$r = \text{sqrt}\ [(a - \text{lower})^2 + (b - \text{start})^2\ ]$$

After identifying the frame, it is divided into seven windows (Fig. 3). Six of these windows are identified by determining the angles associated with each of these windows. The seventh one is obtained by having an inner circle which has the same center as the frame. The length of the radius of the inner circle is presented as r7 which is computed to have a value as a percentage of the length of frame's radius.

Therefore, identifying the window number associated with a given feature located at (x,y) position is done as follows :

- Calculate the distance between (x, y) and the center (a, b) and the feature will be located in window 7 if the distance is less than or equal to r7.
- If the above is not the case, a comparison between y and b is done to determine if the feature is located in the upper or lower half. Then the exact window number is

identified by determining the angle (q) associated with this feature, where q is calculated as follows :

$$q = \operatorname{atan} [(y-b) / (x-a)]$$

For instance, Fig. 4 shows the distribution of the features associated with the Arabic character (Taa-isolated).

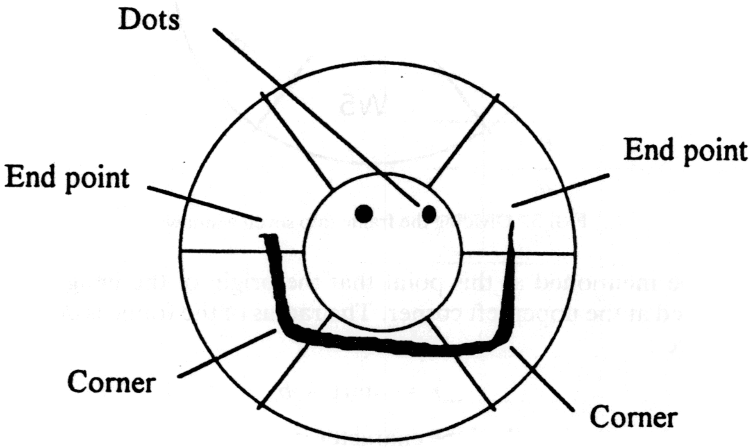

FIG. 4. The layout of the Arabic character 'Taa' with its features.

3.4 The Recognition Process

Identifying of all features of a character produces an entry in the database. The database itself is built by filling it with entries constructed from characters being studied at learning stage. (This process is described in more detail in the next section). After the learning stage, the recognition of unknown characters is done by finding all features associated with the unknown character, then constructing its vector. This is followed by a table look-up to compare the vector of the unknown character and entries in the database.

4. Implementation and Experimental Results

More than 300 images were created for testing the recognition rate of this system. All these images are generated by the same person. Each of these images contains a single handwritten Arabic word. With the average rate of four characters per word, this implies dealing with more than 1200 Arabic characters. A group of 130 words was chosen as a training set, which is used to create the initial database. The remaining 170 words were used as test data for calculating the recognition rate, which is discussed in the next section.

Several tests were implemented depending on the angles to be selected and the radius of the inner circle (i.e., window 7). The results of these tests are shown in

(Table 2). Distributing the features associated with the character under consideration among these windows eventually produces the database entry associated with this character. This entry will be used to search for a matching entry within the database.

TABLE 2. The results obtained after implementing several tests on the value of the angle and percentage associated with r7.

Test	Angle associated with w1, w4	Angle associated with w2, w5	Angle associated with w3, w6	Percentage associated with r_7	Absolute recognition	Recognition by learning	Mis-recognition
1	45	90	45	10	76.21	20.69	2.76
2	60	60	60	10	80.21	18.37	1.41
3	60	60	60	40	81.11	17.51	1.38
4	60	60	60	50	82.61	15.84	1.55
5	60	60	60	60	83.40	14.98	1.62
6	60	60	60	70	82.33	15.36	2.31
7	55	70	55	60	80.92	16.26	3.82

In order to appreciate the results obtained from this work, the reader should have an idea about the results of similar projects. As a matter of fact, most of the work in Arabic character recognition deal with typed text or handwritten characters. Very few research projects deal with handwritten words. Table (3) gives a quick review of

TABLE 3. A quick review for the results of this work and other relative researches.

	This work	The work of (3)	The work of (6)	The work of (9)
Absolute recognition	83.40	79	81.25	83 *
Mis-recognition	1.62	3	2	7.3
Recognition by learning	14.98	13	10	0
Rejection due to wrong segmentation	0	5	6.75	9.7

the results obtained in this work and the work of other relative researches. The first two columns are the results obtained after testing the same set of data. This is true for learning and testing phases. It is clear that a considerable improvement is achieved. Although, the third and fourth columns are the results of other work on different sets of data, yet they are brought to insure that the results obtained from this work are as good as other relevant work. In other words, to appreciate this work the reader should compare the results of this work with the work of the same scope, which is handwritten words in this case. In fact, it is unfair for the reader to compare the results of this work with the results obtained from recognizing printed text which are relatively high. The term "recognition by learning" is used instead of "rejection" since the system deals with the unknown characters in a different manner. Therefore, rather than just rejecting the unknown characters, the system with its interactive ability asks the user if he/she wishes to insert that character in the database for future use. This means that these unknown characters can be stored in the database to be used later on.

5. Conclusion

The process of Arabic character recognition is still an open field; more research is needed. As stated before, new research should be directed at achieving greater accuracy in less time. In order to reach this goal research in character recognition for other alphabets introduce a kind of parallelism, or use a kind of associative matching to speed up the recognition time. Moreover, the characteristic of Arabic characters should be considered when a new approach is derived. Sometime "quick and dirty" algorithms can also be used as a preliminary step to identify what might be called a superclass or cluster to narrow the scope of a search.

Regarding this work, simplicity is the key to the process of design and implementation. There are no complex equations to be evaluated nor complex features to deal with. This approach derives its power from associating simple features with windows. However, the work deals with features in the absolute sense (*i.e.*, they either exist or do not exist). This implies the exact matching of the features of the character under consideration for recognition. As a result, many entries might be associated to one character which means a larger database to deal with. For the future research, the introduction of fuzzy concepts might be helpfull to eliminate different entries associated with the same character.

It should be mentioned at this point that thinning has become less popular in recent research related to Arabic Character Recognition[9]. This is because thinning takes many iterations in order to obtain the skeleton of the Arabic word. Therefore, thinning is considered a time consuming process. Also, some noise may be obtained and should be considered for further processing. Therefore, the new trends move towards obtaining and analyzing the contours of the word. However, dealing with skeletons is more natural than dealing with contours.

References

[1] **Jambi, K.**, Arabic Character Recognition: Many Approaches and One Decade, *The Arabian Journal for Science and Engineering, KFUPM,* **16**(4): 499-510 (1991).

[2] **Jambi, K.**, *Design and Implementation of a System for Recognizing Arabic Handwritten Words with Learning Ability,* Ph.D. Thesis, Illinois Institute of Technology, Chicago (1991).

[3] **Jambi, K.**, A System for Recognizing Arabic Handwritten Words, *Proceedings of The 13th National Computer Conference, Riyadh,* **1**, 416-426 (1992).

[4] **Nouh, A., Sultan, A.** and **Tulba, R.**, An Approach for Arabic Character Recognition, *J. Eng. Sci., Univ., Riyadh,* **6**(2): 185-191 (1980).

[5] **Chin, R.**, A One-Pass Thinning Algorithm and Its Parallel Implementation, *IEEE IECON,* 113-118 (1986).

[6] **Almuallim, H.** and **Yamguchi, S.**, A Method of Recognition of Arabic Cursive Handwriting, *IEEE Trans. on PAMI,* **9**(5): 715-722 (1987).

[7] **Jambi, K.**, An Approach for Segmenting Handwritten Arabic Words, *Proceedings of The Second Conference on Arabic Language, Casablanca, Morocco,* 233-243 (1993).

[8] **Han, G., Jang, T.** and **Foster, S.**, Identification of Corner Points of Two-Dimensional Images Using a Line Search Method, *Pattern Recognition,* **22**(1): 13-20 (1989).

[9] **Amuer, A., Romeo-Pakker, K.** and **Lecourtier, Y.**, L'Arabe Manuscrit et sa Reconnaisance Informatique, *Proceedings of The Second Conference on Arabic Language, Casablanca, Morroco,* 244-254 (1993).

طريقـــة تجريبيـــة للتعرف عـــلى الكلمـــات العربيـــة المكتوبـــة بخـــط اليـــد

كمال منصور جمبي
قسم علوم الحاسب الآلي ، كلية العلوم ، جامعة الملك عبد العزيز
جـــدة ، المملكة العربية السعودية

المستخلص . ينـاقش هذا البحث عملية تنفيذ منـظومـة للتعرف على الكلمات العربية المكتوبة بخط اليد . فحتى يتم التعرف على الكلمة لابد من التعرف على الحروف المكونة لتلك الكلمة ، حيث يتم ذلك عن طريق عملية التقسيم . وفي هذا التطبيق تمر الحروف العربية من خلال مرحلة ما قبل المعالجة والتي تتبعها مرحلة المعالجة نفسها . فكل حرف يتم دراسته بغـرض تحديد الصفـات المميزة والمصاحبة لرقم النافذة التي تقع فيها تلك الصفـات . كما يتم في البحث مناقشة الخطوات التي تم اتخاذها لتحديد تلك النوافذ والصفـات المميزة . يلي ذلـك ، البحث في جدول لإيجاد اسم الحرف المطلوب التعرف عليه . وينتهي البحث بمناقشة النتائج وما تعنيه ، بالإضافة إلى القيام بمقارنة النتائج التي تم الحصول عليها في هذا البحث مع نتائج البحوث المشابهة له .