2012 International Conference on Advanced Computer Science Applications and Technologies

Preprocessing Algorithms for Arabic Handwriting Recognition Systems

Hanene Boukerma[1,2], Nadir Farah[2]
[1]Ecole Normale Supérieur de l'Enseignement Technologique (ENSET), Skikda, Algeria
[2]University Badji Mokhtat BP 2, 23200, Annaba, Algeria
LABoratoire de Gestion Eléctronique du Documents: LABGED
{boukerma, farah} @labged.net

***Abstract*—Preprocessing is one of basic phases of handwritten text recognition and it is crucial to reach high recognition rate. In this paper, we present several algorithms for Arabic handwritten text which are based on inherent properties of Arabic writing. These algorithms include noise removal and smoothing, diacritics detection, contour tracing/correction, baseline estimation, slope correction and detecting/correcting touching descenders. In first stage of propositions validation, each presented method is individually tested on the commonly used IFN/ENIT database. Then, the influence of the presented algorithms on the recognition rate is studied based on K-NN classifier and hybrid features. The obtained results show the efficiency of the proposed algorithms and their positive impact on features discrimination and recognition performance.**

Keywords-Arabic handwriting; preprocessing; subword; diacritics; baseline; slope correction; segmentation of touching descenders.

I. INTRODUCTION

The performance of any text recognizer depends on the quality of the input text and the lack of noise even more so than with humans. Document image of poor quality and high intra-class variation poses greater difficulty for recognizers. In [1], [2] and [3], the authors experimentally analyze the effects of distortion, noise and preprocessing stages on the recognition rate of their systems

In this regard, the input text image should be preprocessed to simplify recognition by removing all distortion and uninteresting variations in writing styles. Preprocessing operations are usually specialized image processing operations [4] that transform the image into another with reduced noise and variation. Those operations include binarization, noise removal, smoothing, thinning, contour analysis, baseline estimation, and text normalization such as skew correction and character normalization. The application of all these operations is not imperative in each recognition system. However, preprocessing is especially important when recognizing handwriting of some difficult cursive scripts such as Arabic, where the shape of the characters is context sensitive.

Arabic handwritten has its particularity in comparison with other scripts such as Latin which may pose significant challenges in employing conventional pretreatments, for example:

- Many of Arabic characters have dots: one, two or three dots, which are small connected components positioned above or below the character. Dots can be viewed as a noise (salt and pepper noise) and vice versa, which increases the complexity of the noise removal operation of Arabic text.
- The conventional thinning algorithms proposed for Latin produce a short line for one dot and two dots when these latter are written as a short line. Therefore, the thinning of single dot and two dots will produce nearly the same skeleton which makes characters identity ambiguous, as different number of dots indicates different characters.
- Writers frequently elongate characters for aesthetic reasons or to justify text. These elongations appear as extensions of the baseline between characters, or lengthening of hooks, often creating vertical overlaps with neighboring characters [5] which complicates more the processing of Arabic written.
- For Arabic handwritten, the conventional methods which extract baseline as straight line are ill-suited because some Arabic words may be contracted from two or more subwords (or PAW: Piece of Arabic Word), and the distribution of these subwords can produce different slant angles within the same word.
- Succession of ascender letters can produce the touching characters problem, which in its turn perturbs a set of an important preprocessing stages e.g. connected components extraction, projection profile analysis and baseline localization.

These features impose choices different from Latin and we oblige to develop processing techniques more suitable for Arabic handwritten words/text.

In this paper, a set of preprocessing algorithms for Arabic cursive handwriting are proposed. First, using the binary images of IFN/ENIT [6] database, we remove noises to avoid their consideration as single diacritics. Then, we perform smoothing, thinning [7], contour tracing/correction, diacritics extraction, and baseline estimation. Compared to existing approaches of baseline estimation, the proposed method brings novelty as it used the subword level as the real basic block to be processed rather than word level; subsequently the estimated curved baseline reflects more the variation of slant angles within the same treated word. The next very important steps during handwriting preprocessing

978-0-7695-4959-0/13 $25.00 © 2013 IEEE
DOI 10.1109/ACSAT.2012.59

are –slope correction based baseline extraction, which is performed in order to reduce writer variability, and –segmentation of touching descenders. While a segmentation of touching characters has been applied to the lines touching/overlapping problem, as far as we know this is the first time that the touching characters problem has been treated within the same cursive word.

The rest of the paper is organized as follows: in Section 2, we present the proposed preprocessing in a series of separately analyzed steps and evaluate the contributions of each one. Section 3 introduces the subsequent feature extraction and recognition stages in order to show how the proposed preprocessing methods influence the system performance. In Section 4, we evaluate and compare the proposed system. Section 5 concludes the paper.

II. Algorithms for Preprocessing Arabic Handwriting

All preprocessing methods presented in this section were tested on handwritten Arabic words of the benchmark IFN/ENIT database [6]. This database contains 26,459 images of 947 Tunisian towns/villages names, written by 411 different writers. It comes with baseline ground-truth (GT) and with ground-truth on character level; the baseline ground-truth is used to evaluate our baseline estimation method.

A. Noise Removal, Hole Filling and Smoothing

The noise removal operation consists of identifying pixels of background which do not belong to the word shape and removing them. Whereas smoothing operation is used to reduce the noise or to straighten the edges of the characters.

For noise suppression, a surface analysis of each connected component is applied; the connected components which have a lower surface than the 5 pixels are removed. By visual inspection of these marks, a value of 5 pixels (generally, equals to the pen thickness of word) was found to be suitable. In addition, the hole filling is realized by an analysis of the 8-neighbors of the background pixels (white pixels `0'). If the sum of the 8-neighbors of the pixel Pi is equal or higher than 5 pixels, Pi is made black (`1'). This operation can also smooth the word contour.

Nevertheless, the real smoothing step is carried out by the opening operation of the mathematical morphology. This method is simple to implement and proves its effectiveness on the masks method proposed in [8], particularly, in the cases of the small loops (see Fig. 1).

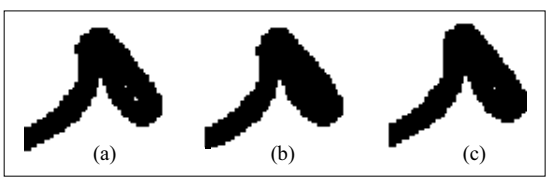

Figure 1. Smoothing oparation and loops fill problem. (a) original image of character 'Mim'. (b) smoothing by masks method of Cheriet et al [8]. (c) smoothing by the opening operation of mathematical morphology.

For the same reason, the closing operation was skipped from the smoothing stage.

B. Diacritics Elimination

In our work, the diacritics such as dots and zigzag should be eliminated before baseline detection to avoid the disturbance of local minima points selection and subwords localization. For diacritics detection, we use a modified version of diacritics extraction algorithm proposed in [9], this algorithm (Algo. 1) is based on the area, the height and the relative position of the connected components, the used thresholds were determined empirically:

Algorithm 1 (Diacritics detection). Let *Thiks* be the approximate pen size (thickness) of a word which is estimated by counting the run-lengths of black pixels in each column and line of the word image, then the most frequent run-length is adopted as the pen size.

For each connected component CC_i do:

1. Determine the area and the height of CC_i.
2. If $CC_i.Area > 15 \times Thiks^2$ then CC_i is not diacritic mark.
3. Else
   a. If $CC_i.Height \leq 3 \times Thiks$ then CC_i is diacritic mark.
   b. Else
      i. If $CC_i.Height > 5 \times Thiks$ then CC_i is not diacritic.
      ii. Else
         1. If CC_i is located at the top of another connected component CC_j which is vertically overlaps CC_i with more than 75% then CC_i is diacritic.
         2. Else CC_i is not diacritic mark.

Next

Table 1 shows the "false positives" and "false negatives" of diacritical marks detection obtained by our algorithm compared with Menasri's algorithm [9]. The false negatives detection problem is identified when the number of diacritics is less than the right number; whereas, the false positives is detected when the connected component is misclassified as diacritics instead of base forms (subword or part of subword). The experiments were conducted on 751 first images of set a of IFN/ENIT database. Because the diacritics ground-truth information is not included in IFN/ENIT GT, a manual evaluation was performed.

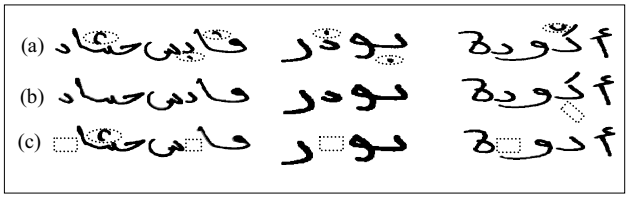

Figure 2. (a) Original images of three handwritten Arabic words with their diacritics rounded by dot circle. (b) results of the proposed algorithm. (c) results of [9] algorithm show their drawbacks (dotted box: erroneous consideration of small isolated letters as diacritics).

TABLE I. THE ERROR RATES OF DIACRITICS EXTRACTION ALGORITHMS

	False positives detection	False negatives detection
Menasri's algorithm	14.24%	5.32%
Our algorithm	5.72%	7.05%

The results in Table 1 and Fig. 2 show that the proposed algorithm performs well in case of small isolated letters thus it obtains 5,72% of false positives detection which presents considerable improvement from the method in [9]. However, it fails in some cases (751/53=7.05%) where the diacritics are larger when compared to the actual size of the word such as large zigzag (ﯼ) and hamza (ﺉ). A proposed solution is the application of second filter of the image after finding the baseline, so a small connected component which is totally located above or below the baseline will be marked as diacritic. The nature of our baseline which is not straight line permits the validation of this assumption.

C. Contour Tracing/Correction

Contour tracing is a common stage in the majority of OCR systems. In our work, we use the mathematical morphology operations to extract both outer and inner contours of the word. The contour is then traced clockwise by Freeman chain code (Fig. 7) starting by the top right point of contour.

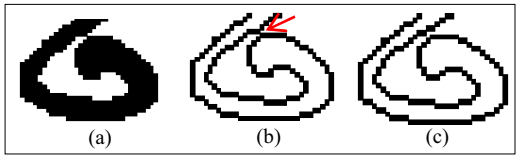

Figure 3. (a) Special form of character Ta' Marbouta. (b) problem of follow-up of contour by Freeman chain code. (c) application result of our fourth mask (see Fig. 4).

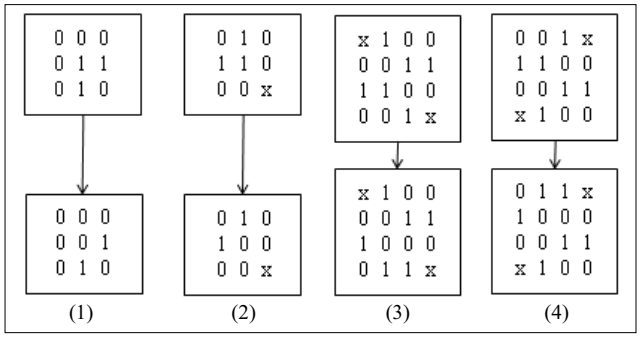

Figure 4. Some examples for our filter mask couples that are used for chain contour correction (x: pixel ignored).

Even if smoothing stage preceded contour tracing operation, the follow-up of contour by the Freeman chain code would fail in some special character forms (we note here that in some cases, this problem is created by smoothing operation itself). The region demonstrated by the red arrow in Fig.3.b illustrates an example.

To correct this situation, we develop more than 15 filter masks couples (3 by 3 or 4 by 4) for the detection/correction of the distortion in contour (Fig. 4). These masks are passed over the entire contour image to remove or to change the position of the pixel which causes the problem without affecting the connected contour. This process can be repeated until there is no change in the contour image.

D. Baseline Detection

As it has already been noted, the irregular distribution of subwords in the same handwritten word loading on different slant angles makes the extraction of perfect baseline as straight line one of the major challenges of Arabic baseline detection algorithms [10], [11].

However, finding this line based on each subword of the word seems to be a promising solution to robust baseline estimation. Based on this assumption, we propose an algorithm of baseline estimation (Algo. 2) in which the basic entity is the subword.

Algorithm 2 (Baseline detection)
Input: image, contour, skeleton, and thickness of word

1. Remove all diacritics detected by Algo. 1 and consider the remaining connected components as subwords.
2. For each subword image, estimate its appropriate horizontal band (*HB-subword*) by applying the following steps:
   a. Divide the image of the word on three equal horizontal partitions and use the second partition as the first horizontal band of whole word (*HB-word*).
   b. In *HB-word*, find the lowest points of closed loops and the branch points and cross points of word skeleton.
   c. Determine *HB-subword* as a horizontal range centered by the feature points selected in (b) and that has a total height equal to five times the approximate thickness of the word.
3. If there is subword *SWi* which do not contains any feature points of (b), so make *SWi* inherit the horizontal band of its nearest neighbor subword in the image.
4. If no feature points of (b) are extracted on all subwords of the word, then use of the horizontal projection method for baseline estimation.
5. Else, inside the *HB-subword* of each subword, detect the baseline relevant support points which are: - the local minima points of the lowest outer contour and - the lowest points of closed loops located next to lower *HB-subword*.
6. Trace the baseline of the entire word by applying the linear interpolation on each two consecutive support points selected in (5).

Output: not a straight baseline.

The proposed baseline detected algorithm was tested on 2240 first images of set-a of IFN/ENIT database. The obtained results are very encouraging (Table 2) and outperform those obtained by Farooq et al. [10]: 78.5% for [10] compared to 87.19% for our algorithm. For further detail on the proposed algorithm and the discussion of their results, the reader is referred to [12].

TABLE II. THE RESULTS OF THE PROPOSED BASELINE ESTIMATION ALGORITHM WITH DIFFERENT BASELINE ERROR.

Baseline error in pixel	10	15	20	25
	69.11%	87.19%	94.06%	97.68%

E. Slope Correction

The slope or skew is the angle between the horizontal direction and the direction of line in which the writer aligned the word (baseline). In our work, the extracted baseline is based on each selected subword and then reflects the different skews in the word. Using this line, we can achieve good slope normalization. The simple slope correction algorithm is as follows:

For each subword *SWi* of the inputted word, we determine *Dth* the distance between *y-mean* of *SWi* baseline and *y-mean* of word baseline. If *Dth* exceeds a threshold, we shift all pixels of *SWi* vertically by *Dth* pixels, so that *SWi* baseline becomes horizontal. Fig. 5 shows examples of slope correction; the image of word before slope correction is presented by watermark image, however, the slope correction result is showed by contour representation (blue color). The black line displays the re-estimated baseline of the normalized word.

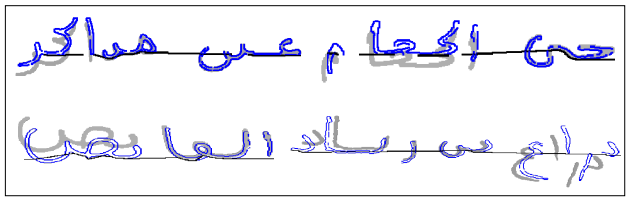

Figure 5. Slope correction results.

F. Detecting/Correcting Touching Descenders

The case of words (or subwords) with touching characters is a probable problem in free handwriting Arabic (Fig. 6.a.i and 6.a.ii). In this section, we propose a solution of touching characters problem when the characters are the descenders.

The touching descenders problem often appears in case of succession of descenders with a low extension which can be touched (Fig. 6.a.i), that engenders the appearance of the branch points in the skeleton word below the lower baseline. In order to solve this problem, we propose a simple algorithm of five stages:

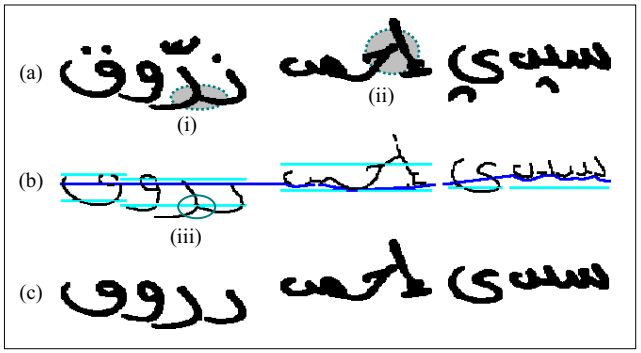

Figure 6. Detecting/Separating touching descenders. (a) original word image. (b) skeleton, (iii) branch point under the lower baseline. (c) segmentation of touching descenders.

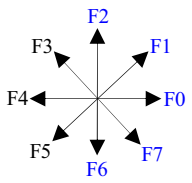

Figure 7. Freeman chain-coded contour

Algorithm 3 (Segmentation of touching descenders)
Input: baseline, word skeleton, and thickness

1. If there is branch points (*Bpt*) that are under the lower baseline, it means that the touching descenders problem exists in the examined word, then:
2. Determine the cut point (*Cpt*) by the follow of contour starting by the found *Bpt* and according to the five directions of Freeman: F6, F7, F0, F1 and F2 in this order (Fig. 7).
3. Cut the shape of the word in the determined *Cpt* and with width equal to word thickness.
4. If step (3) creates wastes in the word, then delete them.
5. Re-smooth the word contour.

Output: segmentation of touching descenders.

In Algo. 3, the follow of contour according to adopted order ensure that the cut operation is done on the first right character shape in the word. To validate this algorithm, we tested it on several words of IFN/ENIT dataset; Fig. 6 shows an example of satisfactory result.

III. FEATURE EXTRACTION AND RECOGNITION

Since the goal of this section is an overall evaluation of the proposed preprocessing steps combined together, it makes sense to first try a simple recognition method (K-NN) to see what performance can be achieved.

Face to the segmentation difficulties of Arabic word image into letters, experiments with K-NN classifier has been conducted on PAW level of Arabic handwritten words.

Using IFN/ENIT database, we have developed a novel database of Arabic handwritten subwords namely: "PAW-IFN/ENIT" [13], this database contains a total of 74104 images of handwritten subwords labeled according to 759 PAW classes and divided into four subsets a, b, c, and d in the same way as IFN/ENIT database; a complete description of the proposed scheme is in [13].

In feature level process, a set of hybrid features were chosen in order to capture the most essential and pertinent characteristics of deferent classes of PAW. These features are extracted on the subword image; some of them are statistical such as pixel density in various homogeneous zones of the subword skeleton (horizontal, vertical, diagonal 45°, and diagonal 135°), upper, lower, left and right profiles of subword, etc. Other features are structural such as aspect ratio, chain code, and number of loops, etc.

For each type of feature, a subset is baseline dependent such as histogram profiles and pixel density distributions related to baseline, the position of the center of gravity (above or below the baseline), distance between the y-baseline and the y-start point in skeleton subword without diacritics, etc. In a total 54 features were extracted, 20 of them are baseline dependent (Fig. 8).

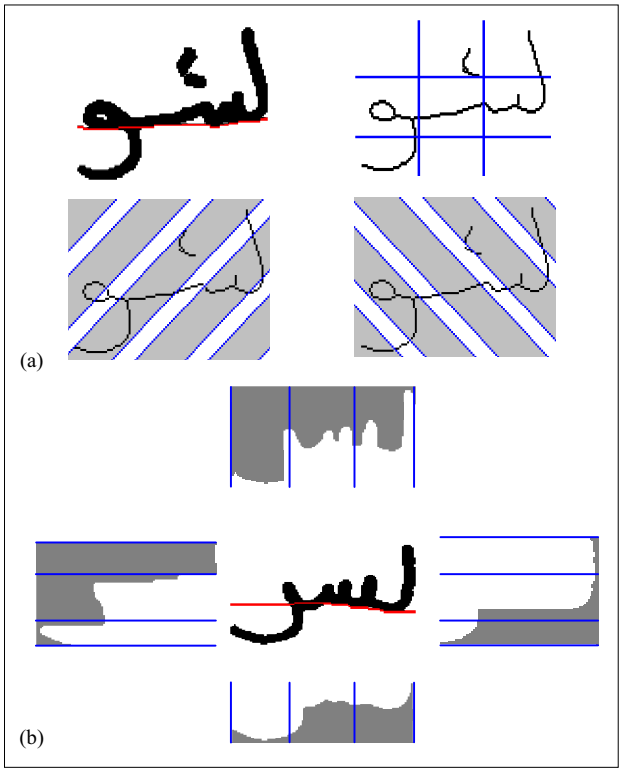

Figure 8. Example of extracted feature. (a) pixel density distributions of subword skeleton. (b) upper, lower, left and right profiles (red line : baseline).

IV. TESTS AND RESULTS

We notice that the occurrence of examples in IFN/ENIT database is different from one word class to another which is also true for our PAW-IFN/ENIT database.

However, as it was proved in the literature, the recognition rate increases dramatically when a larger number of training examples are available for each class. For this reason, we chose the PAW classes which occur more than 13 times in the whole PAW-IFN/ENIT database and which are the most frequents in word lexicon. So we use 150 PAW classes which correspond to 272 classes of words. Fig. 9.a presents some examples of the selected PAWs.

In our test, we have used 24437 PAW images for training and 6739 PAW images for testing. The experiments have been carried out as follows. First, we have tested the set of 34 baseline independent features, the obtained accuracy was 83.39% (Top1). Then, adding the 20 baseline dependent features (so we have 54 features) and normalizing the feature vectors the recognition rate increases to 90.1% (Top1). This shows that the use of both baseline dependent and independent features and of the proposed preprocessing operations increase recognition performance. Fig. 9.b illustrates some cases where the baseline is essential to distinguish between some PAW classes, especially, between isolated letters.

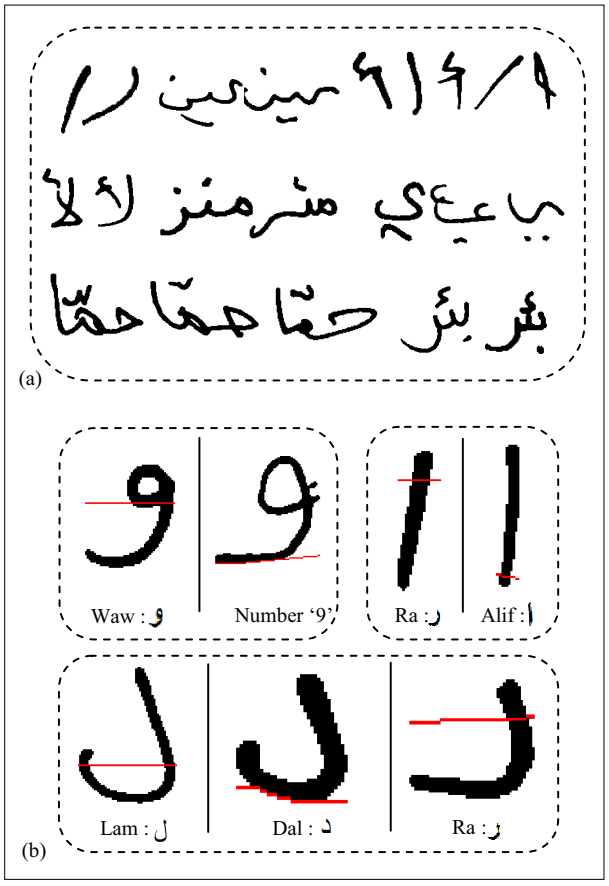

Figure 9. (a) Examples from our PAW-IFN/ENIT database: subword images used for system testing. (b) Importance of the baseline (red line) to distinguish between subwords.

Additionally, our system provided better recognition rate in comparison with the work of Burrow [14], even using similar classifier. Like in our work, Burrow used K-NN based classifier to recognize the subwords of IFN/ENIT database. He used different types of features (pixel representation, Zernike moments, tracing approach, etc) and achieved a recognition rate of 80% on small lexicon of 30 PAW classes. This difference in recognition rate may be caused by differences in the applied preprocessing techniques or the features extraction method.

Our result can also be compared to the result by Haboubi et al. [15] who have obtained a 87.1% of recognition rate using pixel representation based HMM classifier on reduced lexicon of 122 Tunisian town names of IFN/ENIT database.

V. CONCLUSION

A set of preprocessing algorithms for Arabic handwriting recognition systems were presented. We believe that correct preprocessing stage leads to high recognition accuracy and we have demonstrated that with commonly used K-NN based classifier.

First, the proposed algorithms for each step was evaluated separately using a standard database and sometimes against a competing algorithm. A key challenge detected here is that some algorithms rely on heuristics. Our proposed solution consisted in using pen thickness as parameter in threshold estimation which makes the approaches less sensitive to parameters setting. Further experiments will be carried out to check the generalization ability of the proposed methods for other Arabic database.

In next stage of proposition evaluation, it was shown that the use of the preprocessing steps and the baseline dependent features improved recognition accuracy. The obtained result was quite satisfactory and comparable to those reported in literature on similar lexicon size. Future work will involve the use of other features set and other recognition engines to deal with large lexicon size.

REFERENCES

[1] H. El Abed and V. Märgner, "Comparison of different preprocessing and feature extraction methods for offline recognition of handwritten Arabic words," in International Conference on Document Analysis and Recognition (ICDAR'07), Vol. 2, pp. 974–978, 2007.

[2] L. Likforman-Sulem, J. Darbon and E. H. Barney Smith, "Pre-processing of degraded printed documents by non-local means and total variation," in International Conference on Document Analysis and Recognition (ICDAR'09), pp. 758–762, 2009.

[3] R. Al-Hajj, L. Likforman-Sulem and C. Mokbel, "Arabic handwriting recognition using baseline dependant features and hidden markov modeling," in 8th International Conference on Document Analysis and Recognition, ICDAR'05, pp.893-897, 2005.

[4] B. Al-Badr, S. A. Mahmoud, "Survey and bibliography of Arabic optical text recognition," Signal Processing 41, pp. 49–77, 1995.

[5] J. Chan, C. Ziftci, and D. Forsyth, "Searching Off-line Arabic Documents," Proc. of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006.

[6] M. Pechwitz, S. Snoussi Maddouri, V. Märgner, N. Ellouze and H. Amiri, "IFN/ENIT database of handwritten Arabic words," Proc. of Colloque International Francophone sur l'Écrit et le Document (CIFED'02), pp. 129–136, 2002.

[7] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," Image Processing and Computer Vision, vol. 27, no. 3, 1984.

[8] M. Cheriet, N. Kharma, L. C. Liu. and C. Y. Suen, Character Recognition Systems, A Guide for Students and Practioners, published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2007.

[9] F. Menasri, N. Vincent, E. Augustin and M. Cheriet, "Un système de reconnaissance de mots arabes manuscrits hors-ligne sans signes diacritiques," Actes du dixième Colloque International Francophone sur l'Écrit et le Document, CIFED'08, 2008.

[10] F. Farooq, V. Govindaraju and M. Perrone, "Pre-processing methods for handwritten arabic documents," in 8th International Conference on Document Analysis and Recognition (ICDAR'05), 2005.

[11] A. M. Al-Shatnawi and Kh. Omar, "Methods of arabic language baseline detection: the state of art," IJCSNS International Journal of Computer Science and Network Security, vol.8, no.10; 2008.

[12] H. Boukerma and N. Farah, "A Novel arabic baseline estimation algorithm based on sub-words treatment," Proc. of International Conference on Frontiers in Handwriting Recognition (ICFHR'10), IEEE Computer Society, pp. 335–338, Kolkata, India, November 16-18, 2010.

[13] H. Boukerma and N. Farah, "PAW-IFN/ENIT: une nouvelle base de pseudo-mots arabes pour une approche de reconnaissance pseudo analytique," Proc. of the eleventh African Conference on Research in Computer Science and Applied Mathematics (CARI'12), pp. 491–499, Algeria, October 13-16, 2012.

[14] P. Burrow, Arabic Handwriting Recognition. PhD thesis, Master of Science, School of Informatics University of Edinburgh, 2004.

[15] S. Haboubi, S. Maddouri, N. Ellouzi and H. El-Abed, "Invariant Primitives for Handwitten Arabic Script: A contrastive study for four feature sets," in International Conference on Document Analysis and Recognition (ICDAR'09), pp. 691–697, 2009