Segmentation of Handwritten Textlines in Presence of Touching Components

Jayant Kumar Le Kang David Doermann Wael Abd-Almageed
Institute of Advanced Computer Studies
University of Maryland College Park, USA
{*jayant, lekang, doermann, wamageed*}*@umiacs.umd.edu*

***Abstract*—This paper presents an approach to textline extraction in handwritten document images which combines local and global techniques. We propose a graph-based technique to detect touching and proximity errors that are common with handwritten text lines. In a refinement step, we use Expectation-Maximization (EM) to iteratively split the error segments to obtain correct text-lines. We show improvement in accuracies using our correction method on datasets of Arabic document images. Results on a set of artificially generated proximity images show that the method is effective for handling touching errors in handwritten document images.**

***Keywords*-Text-lines, Handwritten Documents, Arabic**

I. Introduction

Segmentation of text-lines can be a crucial step in many document processing tasks including skew detection, layout analysis and word/character recognition. For handwritten text, the problem is even more difficult due to its free style nature, character size variations and non-uniform spacings between components. Moreover, the touching of characters across lines and overlapping spatial envelopes of text-lines make the problem more challenging. Methods previously developed for segmentation of printed text-lines do not adapt well to these scenarios.

Methods based on connected-components are fast but they suffer from touching or close proximity of components. In Figure 1, we show the result of a connected-component based text-line extraction method [1] on a document image with many touching components (shown in dotted boxes). When the touching components are separated manually by cutting the characters using an image editing tool, the same method gives correct segmentation without any change of parameters. One obvious solution is to detect and correct such touching errors before applying text-line segmentation. But this may be computationally expensive as the number of components in a document image may be large. We take another approach in which the detection and correction of such errors are delayed until an initial estimate of text-line segmentation is obtained. Each line is then checked for touching and proximity errors. This is computationally more efficient as the number of lines detected (20-30) are far less than the number of components (600-800) in a typical document image.

In this work we present a graph based text-line segmentation method which combines both local and global approaches to first obtain an initial estimate of text-lines. In the next step, we use the distances along the nodes in the local-orientation graph to automatically detect touching and proximity errors. Expectation-maximization (EM) is then iteratively applied to split the touching lines. Finally, the error components are localized and cut to obtain a accurate estimate of text-lines. The main contribution of this work are a graph based error detection technique and an EM based correction technique which have shown promising improvements on our handwritten Arabic data set.

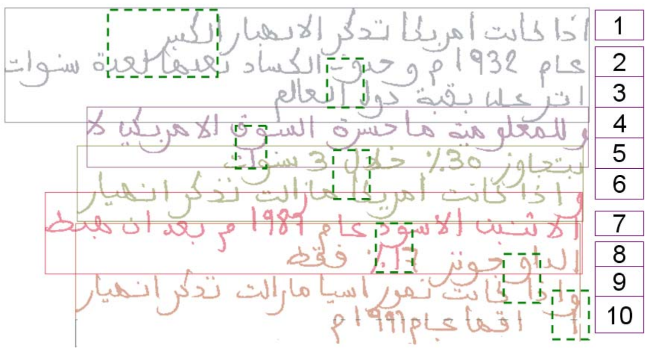

Figure 1. Text-line segmentation errors due to touching components across different lines. Line 1,2,3 are grouped as one segment due to touching components. Text-lines are color coded by the algorithm. Dotted boxes show the various touching component.

The remainder of this paper is organized as follows. Section II overviews the related work on text-line extraction. In Section III we present the proposed text-line extraction approach. We give the details of a pixel-based evaluation mechanism and discuss our experimental results in Section IV and conclude our paper in Section V.

II. Related Work

Existing text-line extraction techniques can be broadly categorized as *projection based*, *component-grouping based* or *hybrid methods*. *Projection based* methods typically divide the document image into vertical strips, compute horizontal projection profiles to extract components and group them based on few heuristics to extract text-lines. In [3], components are grouped by modeling the text-lines as bivariate Gaussian densities. A recent method initially over-segments the zones into text and gap regions, and uses a Hidden Markov Model to find the optimal assignment of text and gap areas in each zone [4]. The width of the zone is selected to maximize the amount of text and minimize the effect of skew in each zone. Due to large variations in width of Arabic characters this criterion may not always be satisfied and the method may give suboptimal performance. In [2], Pal et al. presents a text-line extraction method for handwritten Bangla document images. They use horizontal histograms of the vertical strips and the relationship of minimal values to obtain handwritten text-lines. The piece-wise projection based line computation used in their method may not work well if the lines are closely spaced and the orientation variation within each line is high. *Connected-component based* methods merge neighboring connected components using rules based on the geometric relationship between neighboring blocks, such as distance and size compatibility. In [5], a block-based Hough transform approach is used to detect handwritten text-lines. In a post-processing step false alarms are rectified using a merging method. A very effective method based on curve evolution and level-sets was proposed in [6], but the method is very slow for high resolution document images. Zahour et al. [7] used a partial contour following based method to find the separating lines in handwritten Arabic documents. They proposed a new segmentation method suited for Arabic historical manuscripts to segment the document images into three classes: text, graphics and background. But these approaches seem to be effective only when the components are not touching and the text-lines do not have overlapping envelopes.

III. Text-Line Extraction

Our line detection method consists of four steps: *Coarse text-line estimation*, *error detection and correction*, *touching component localization and separation*, and *diacritic/accent component assignment*. In the following subsections we explain each of these steps in detail.

A. Coarse Text Line Estimation

We filter the probable accent and diacritic components based on the mass and size of components to obtain a set of coarse components(CC). Removing such small components gives us two advantages. First, the reduction in number of components makes the graph-search and the Affinity-propagation method used in next step fast and second, the coarse text-lines can be assumed to have smoothly varying orientation which can be estimated locally. At each coarse component we estimate the direction of text-line by defining a local rectangular coordinate system with the origin at the centroid of the component. The dimensions of the region are adaptive based on the size of current CC and the median height and width of all the coarse components. We divide each of the four quadrants into two sectors giving

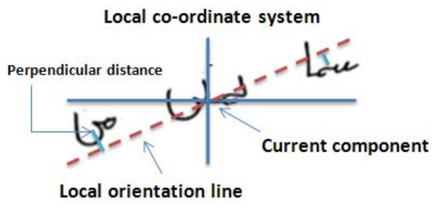

Figure 2. A local cartesian co-ordinate system with the origin at the centroid of component.

eight sectors. Diagonally opposite pairs of sectors represent possible quantized orientation of text-lines. A focused region at $\pm$ 10 degrees with the horizontal axis is also defined to represent approximately horizontal orientation. We count the neighboring components in each of the five regions and find the region with the maximum components(Rmax). Finally the least square estimate line passing through the centroid of components in Rmax is computed to obtain the local orientation (Figure 2).

A graph with nodes corresponding to the coarse components is constructed. The weights on edges between the current CC and the neighboring CCs are given by the distance to the estimated orientation line (Figure 2). Once all of the local orientations and distances are computed for all the CCs, a *shortest-path algorithm* [10] is used to compute the distances between non-neighboring components. We obtain two estimates of text-lines based on this graph. First estimate uses Breadth-First-Search (BFS) [10] to find the different connected-components of the graph which represent potential text-lines if the local estimates are correct. But due to overlapping envelopes, touching components and character size variations, local estimates may not always be correct. Hence we also compute soft similarities with all the other components and use Affinity-propagation clustering method [8] to obtain a global estimate of text-lines. Finally the two estimates are combined based on the definition of a valid text-line and a final set of text-lines are obtained. For more details on this approach please refer [1].

B. Error Detection and Correction

Graph Based Error Detection: Errors in text-line results obtained in previous step are of mainly two types: *merge* errors and *split* errors. When two or more neighboring text-lines in the result correspond to a single line in ground-truth then we refer to it as a *split error*. This usually happens when the local orientation detection fails to identify adjacent neighboring component. This error is detected and corrected by checking against Affinity propagation based results and analyzing inter-component distances of each lines.

Merge errors occur when multiple text-lines are grouped as one segment in our results. For each such segment obtained we find the *All-pair shortest path* distances [10] between the components. We then compare this graph-based

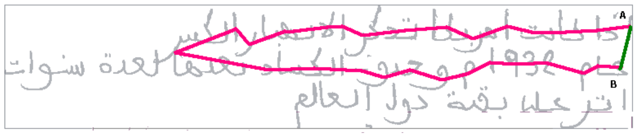

Figure 3. Two components labeled A and B have graph based Euclidian distance (red) along the shortest path much greater than the direct distance (green).

Euclidian distance and the direct Euclidian distance between each pair of components to detect the merge errors. If the difference in the distances along the shortest path in the graph and the Euclidian distances of components is greater than some pre-defined threshold, then the detected line is declared an error segment. As shown in Figure 3, shortest path in the local orientation graph between components A and B is much greater than the direct Euclidian distance on the image. For a valid text-line the proposed scheme works because as we move from one end to another, direct Euclidian distance grows in proportion to the graph-based distance. For a segment with multiple text-lines, coarse components have much longer graph distances than Euclidian distances. We find the pair of components having the maximum difference in both the distances and use it to detect merge errors.

EM Based Error Correction: We iteratively apply EM [9] to split the error detected segment into two segments in each iteration. This is done until the segments obtained in each iteration have no errors. We initialize two lines with slope m_k and y-intercept c_k based on majority of local orientations in each segment (as shown in Figure 4(a)) and update the parameters in each EM step. In the E-Step, we compute the likelihood of each component to be assigned to each line. For this, we find the residual (Res_{ik}) and the weights (w_{ik}) for each component as follows:

$$Res_{ik} = |m_k \cdot x_i + c_k - y_i| \tag{1}$$

$$w_{ik} = \frac{e^{-\frac{{Res_{ik}}^2}{\sigma^2}}}{\sum_k e^{-\frac{{Res_{ik}}^2}{\sigma^2}}} \tag{2}$$

where Res_{ik} represents the residual of the centroid of the i^{th} component with respect to the k^{th} line. (x_i, y_i) represent the centroid of the component. The free parameter σ corresponds to the amount of residual expected in the data.

In the M-step, we find the parameters that maximizes the likelihood of data points. We find the weighted least square estimate for each line. Equation 3 is solved twice (k = 1,2) one for each model using the weights obtained from the E-step.

$$\begin{pmatrix} \sum_i w_i {x_i}^2 & \sum_i w_i x_i \\ \sum_i w_i x_i & \sum_i w_i 1 \end{pmatrix} \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} \sum_i w_i x_i y_i \\ \sum_i w_i y_i \end{bmatrix} \tag{3}$$

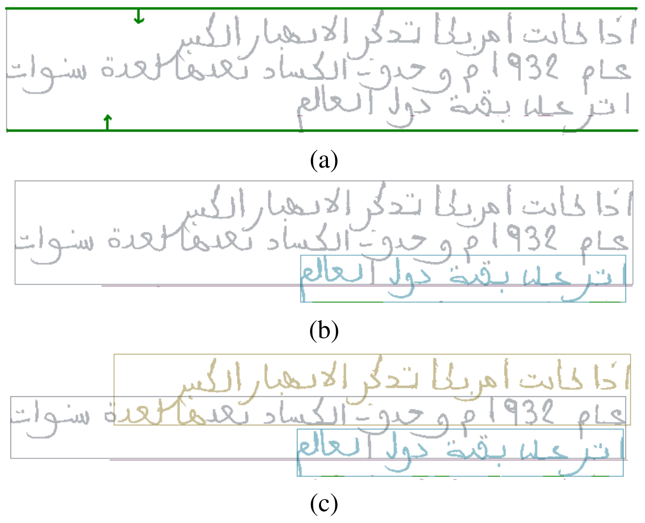

Figure 4. (a) Solid horizontal lines (in green) show the initial lines for EM (b) Two segments obtained using EM. Lower segment has no error but the upper segment has error (c) Results after applying EM on upper segment obtained in (b)

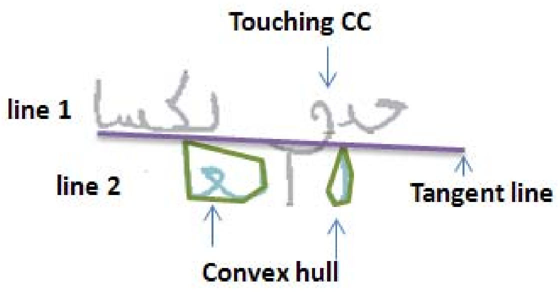

Figure 5. Common tangent to the Convex hulls of consecutive components is used for the error component localization.

C. Touching Component Localization and Separation

Once all the correct segments of an error detected segment are obtained, we localize the touching component in each segment. For this we find the common upper and lower tangent of the Convex-hull of consecutive components in neighboring lines (as demonstrated in Figure 5). The upper and lower tangent to Convex hull of the components give a good approximation of the extent of text-line locally in that region. If the ratio of the length of component below or above this tangent to the total height of the component is more than a certain threshold then the component is considered to be a touching component. The accurate separation of such error detected component requires the knowledge of how character's shape changes when they touch and interact with each other during the writing. In this work we take a very simple approach and cut the component at the junction nearest to the centroid of component.

Table I
COMPARISON OF THE PROPOSED METHOD WITH THE TOP EIGHT METHODS IN ICDAR 2009 SEGMENTATION COMPETITION.

Methods	CUBS	ILSP-LWSeg-09	PAIS	CMM	CASIA-MSTSeg	PortUniv	PPSL	LRDE	Proposed Method
F-Measure	99.5	99	98.5	98.4	95.6	94.5	93.4	92	97.8

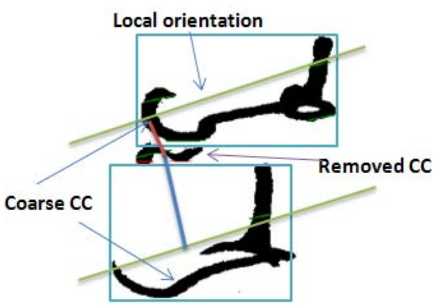

Figure 6. Ambiguity in assignment of diacritic and accent component is resolved based on distance to orientation line. CC refers to connected-component.

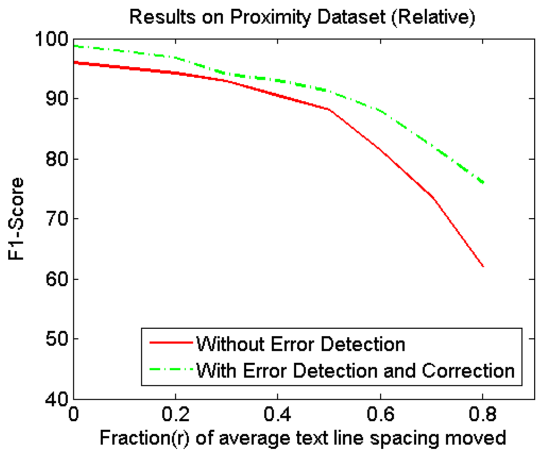

Figure 7. Plots of F-1 scores obtained using our method and a previous method which do not apply any touching error detection and correction method.

D. Assignment of Diacritic and Accent Components

In this final step we assign all the components which were removed as probable diacritic and accent components to the text-lines. Each such component is given the label of best matched coarse component closest to it. We give first preference to the coarse components whose bounding-box fully encloses the bounding-box of the removed component. Second preference is given to those components whose bounding-box overlaps with the removed component. In ambiguous cases, we resolve the ambiguity by computing the distances from the centroid of removed component to the orientation line detected at the coarse components and the component with the minimum distance is chosen (Figure 6).

IV. EXPERIMENTS AND EVALUATION

Our dataset consists of a set of 125 Arabic document images with 1974 handwritten text-lines. We generated a set of proximity datasets using these images to test the robustness of our approach. We moved each line closer to the line above it, in steps of some fixed fraction of average distance between the lines, to generate a series of datasets. We call this the *Relative* proximity dataset [11] which has 19740 text-lines.

We evaluated our results using a pixel-based matching-score(MS) criterion which is computed as follows:

$$MS(r_i, g_j) = \frac{T(P(r_i) \cap P(g_j))}{T(P(r_i) \cup P(g_j))} \qquad (4)$$

where $MS(r_i, g_j)$ is a real number between 0 and 1 and represents the matching score between the result zone r_i and the ground truth zone g_j. P represents the foreground and T is an operator that counts the number of pixels in the zone. We obtain the matching-scores between all the result zones and the ground-truth zones. If the score is found above a pre-defined threshold then the result zone is counted as a True positive (TP). Result zones which are not matched to any ground truth zones are False positives (FP) and the ground-truth zones which are left unmatched are False-negatives (FN). We compute precision, recall and the F1-score as follows:

$$Precision = \frac{TP}{TP + FP} \qquad (5)$$

$$Recall = \frac{TP}{TP + FN} \qquad (6)$$

$$F1_{Score} = \frac{2 * Precision * Recall}{Precision + Recall} \qquad (7)$$

Figure 7 shows the F1 scores obtained using our method on the relative-proximity data at MS threshold of 90. As shown, we obtained a F1 score of 98.76% on the original data (Fraction r = 0). We compare our results with the results in [1] which did not apply touching error detection and correction. As text-lines are moved closer, the performance of the proposed approach does not degrade as rapidly as the other method. We observe an improvement of 2.76% in accuracy on the original data and 14% on the proximity data

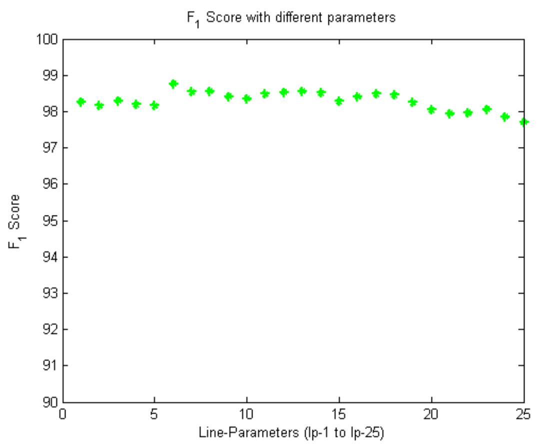

Figure 8. Plot of F-1 scores with different values of parameters.

at r = 0.8. Total number of touching components in the the proximity data at r = 0.8 is 604. If the two lines overlap with each other completely then our method breaks down and the accuracy falls substantially. But in real documents we rarely see all the components of two text-lines touching.

Figure 8, shows the F_1 scores for different sets of parameters. We varied the parameters $HBand_{thres}$ and $VBand_{thres}$ defining the dimension of rectangular region for local orientation computation. Values from 5 to 3 in steps of 0.5 for $HBand_{thres}$ and 0.8 to 0.4 in steps of 0.1 for $VBand_{thres}$ are used to create 25 sets of parameters (lp-1 to lp-25). As shown, our method is robust to these two parameters. We also evaluated our method on ICDAR 2009 segmentation competition dataset (200 images) [12] and F-Measures(FM) of top eight methods along with the proposed method is given in the Table I. Although the proposed method was developed for Arabic, it adapts well to other scripts like English, French, German and Greek used in the dataset. The average time taken for the processing of a single image is 2.2 seconds for the proximity data and 3.2 seconds for the ICDAR competition data on a P4 machine with 3BG RAM.

V. Conclusion

We have presented a graph-based approach for the extraction of handwritten text-lines in monochromatic document images. Using the local orientation graph we detect touching and proximity errors in our results. We then apply EM algorithm to correct errors. The proposed detection and correction scheme relies on the same graph used for clustering and does not add any computational overhead. Proposed method is fast due to the removal of small components in the first step. We demonstrated the effectiveness of our method on different datasets. We also showed the improvement in accuracies on a previous method [1] which did not use any touching detection and correction strategy. In general, our method can be used as a post-processing step in any CC-based method which gives an initial estimate of text-lines.

Acknowledgment

The partial support of this research by DARPA through BBN/DARPA Award HR0011-08-C-0004 under subcontract 9500009235, the US Government through NSF Award IIS-0812111 is gratefully acknowledged.

References

[1] J. Kumar, W. Abd-Almageed, L. Kang and D. Doermann, *Handwritten Arabic Text Line Segmentation using Affinity Propagation*, Document Analysis Systems, pp. 135–142, 2010.

[2] U. Pal and S. Datta, *Segmentation of Bangla Unconstrained Handwritten Text,* Intl. Conf. on Document Analysis and Recognition, vol.2, pp. 1128-1132, 2003.

[3] M. Arivazhagan, H. Srinivasan, and S. Srihari, *A Statistical Approach to Line Segmentation in Handwritten Documents,* Volume 6500. SPIE, 2007.

[4] V. Papavassiliou, T. Stafylakis, V. Katsouros, G. Carayannis, *Handwritten Document Image Segmentation into Textlines and Words*, Pattern Recognition, Vol. 43, Issue 1, pp. 369–377, 2010.

[5] G. Louloudis, B. Gatos, C. Halatsis, *Text Line Detection in Unconstrained Handwritten Documents Using a Block-Based Hough Transform Approach,* Intl. Conf. on Document Analysis and Recognition, vol. 2, pp.599–603, 2007.

[6] Y. Li, Y. Zheng and D. Doermann, *Detecting Text Line in Handwritten Documents,* Intl. Conf. on Pattern Recognition, pp. 1030-1033, 2006.

[7] A. Zahour, B. Taconet, P. Mercy, S. Ramdane, *Arabic Handwritten Text-line Extraction,* Intl. Conf. on Document Analysis and Recognition, pp.281–285, 2001.

[8] B. J. Frey and D. Dueck, *Clustering by Passing Messages Between Data Points,* Science 315, pp. 972–976, 2007.

[9] A. P. Dempster, N. M. Laird and D. B. Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm.*, J. R. Statist. Soc. B, 39:1–38, 1977

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. *Introduction to Algorithms*, MIT Press and McGraw-Hill.

[11] DATASET: Handwritten Arabic Textline Segmentation and Proximity Datasets, Language and Media Processing Laboratory, Univeristy of Maryland College park, http://lampsrv02.umiacs.umd.edu/projdb/project.php?id=65

[12] B. Gatos, N. Stamatopoulos and G. Louloudis, *ICDAR2009 Handwriting Segmentation Contest*, Intl. Conf. on Document Analysis and Recognition, pp. 1393–1397, Barcelona, Spain, 2009