A REAL-TIME LINE SEGMENTATION ALGORITHM FOR AN OFFLINE OVERLAPPED HANDWRITTEN JAWI CHARACTER RECOGNITION CHIP

***Zaidi Razak*[1], *Khansa Zulkiflee*[2], *Rosli Salleh*[3], *Mashkuri Yaacob*[4], *Emran Mohd Tamil*[5]**
Faculty of Computer Science and Information Technology,
University of Malaya, 50603 Kuala Lumpur, Malaysia.
Email: zaidi@um.edu.my[1], kulchazul@yahoo.com[2],
rosli_salleh@um.edu.my[3], mashkuri@um.edu.my[4], emran@um.edu.my[5]

ABSTRACT

Overlapped characters between upper and lower lines can cause a major problem in line segmentation of cursive characters such as Jawi (similar to Arabic characters). In this paper we will discuss our fast approach using a tangent value to find a separation point (SP) for accurate line segmentation without data loss. This approach will be used to design a dedicated Jawi Character Recognition Chip with the objective of obtaining 90% recognition accuracy.

Keywords: Jawi, Segmentation, Overlapped

1.0 INTRODUCTION

The arrival of Islam to the Nusantara archipelago along with Arabic script has further enriched Indonesian literature [1]. During that time, parts of the Nusantara society have began to express their thoughts through a new writing system modified from Arabic script according to the pronunciations used in their language. One of the Arabic script form which was modified based on local requirements is Jawi script. This font contains 28 characters originating from Arabic characters and six characters which are unique for Jawi script. Malay manuscripts are defined as handwritten documents in Malay which were written beginning from the 14th century and stopped in the early 20th century with the arrival of the West and introduction to printing machines. The oldest evidence of Jawi script usage was found on the Terengganu inscription in 1303 A.D. [2].

Although in the modern age the usage of Jawi script is generally replaced by Latin or Roman script; nevertheless Jawi documents are still needed to study South East Asian history and culture [3]. Especially in Malaysia, Jawi script has been and is still widely used in various fields including inscriptions, chronicles, genealogies, tales, religious texts on Islam, diplomatic treaties, legal documents, contracts, petitions and other administrative documents, and newspapers and magazines. Therefore, Jawi character recognition is necessary in order to preserve this valuable heritage.

Segmentation is the most important process in almost all character recognition systems especially in cursive character recognition. Inaccuracy in segmentation which involves line segmentation, word/sub-word segmentation and character segmentation will result in recognition errors. In this research we consider old manuscripts (with many overlapped characters over the lines) as the domain of our problem. This will require planning and designing of a very accurate method in the first stage of segmentation, i.e. line segmentation. In this research, we do not use artificial intelligence in our analysis since artificial intelligence can hardly be implemented in hardware. Our approach does not analyze the image using morphological tracing as has been used by Ymin et. al [4], neither it uses the calculation of pixel averages as has been done by other researchers [5,6]. Instead, our technique uses histogram projection without taking into consideration of the character orientation and line skew. Normalization of the histogram is performed to omit false local minimum SP. This algorithm is introduced in order to improve accuracy and speed, while maintaining the quality of the segmented text lines. In the next section, we review some previous works on text line segmentation in handwritten documents.

2.0 RELATED WORK

Text line segmentation in handwritten documents can be generally categorized into bottom-up and top-down. In the bottom-up approach, the connected components based methods merge neighboring connected components [7-11] using simple rules on the geometric relationship between neighboring blocks. On the other hand, projection based

methods [12-17] may be one of the most successful top-down algorithms for machine printed documents since the gap between two neighboring text lines in machine printed documents is typically significant, thus the text lines are easily separable. However, these projection based methods cannot be directly used in handwritten documents, unless gaps between lines are significant or handwritten lines are straight.

Likforman-Sulem and Faure [7] proposed an approach based on perceptual grouping of connected components of black pixels. Text lines are iteratively constructed by grouping neighboring connected components based on certain perceptual criteria such as similarity, continuity, and proximity. Therefore local constraints on the neighboring components are combined with global quality measures. To handle conflicts, the technique merges a refinement procedure combining a global and a local analysis. According to the authors, the proposed technique cannot be used on degraded or poorly structured documents, such as modern authorial manuscripts.

In [8] the text line extraction problem is seen in the view of artificial intelligence. The aim is to cluster connected components in the document into homogeneous sets, corresponding to the text lines of the document. To resolve this problem, a search is applied over the graph that is defined by the connected components as vertices and the distances among them as edges.

Feldbach and Tönnies [9] proposed a method for line detection and segmentation in historical church registers. This method is based on local minima detection of connected components and is applied on a chain code representation of the connected components. The idea is to gradually construct line segments until a unique text line is formed. This algorithm is able to segment text lines closed to each other, touching text lines, and fluctuating text lines.

In [10] an iterative hypothesis validation strategy based on Hough transform was proposed. The skew orientation of handwritten text lines is acquired by applying the Hough transform to the center of gravity of each connected component in the document image. If most nearest neighbors of the components in the alignment string belong to the group of components forming the alignment, the alignment has both properties of direction continuity and proximity, and it will be accepted as a text line. This enables the generation of several text line hypotheses. Then a validation is performed to eliminate incorrect alignments between connected components using contextual information such as proximity and direction continuity criteria. Based on the authors, this technique is able to detect text line in handwritten documents which may contain lines oriented in different directions, erasures and annotations between main lines.

Louloudis et al. [11] presented a text line detection method for unconstrained handwritten documents based on a strategy that consists of three distinct steps. The first step comprises preprocessing for image enhancement, connected component extraction, and average character height estimation. In the second step, a block-based Hough transform is applied for the detection of potential text lines while a third step is used to correct possible false alarms. A grouping method of the remaining connected components which uses the gravity centers of the corresponding blocks is applied. The performance of the proposed method is determined based on a consistent and concrete evaluation technique that relies on the comparison between the text line detection result and the corresponding ground truth annotation.

One technique divides the text image into columns. In [12], a partial projection is performed on each column. Then a partial contour following method is used to detect the separating lines, in the direction and opposite direction of the writing. Tripathy and Pal [13] used a projection-based method in each column, and combined the results of adjacent columns into a longer text line. The document is divided into vertical stripes. Analyzing the heights of the water reservoirs obtained from different components of the document, the width of a stripe is calculated. Stripe-wise horizontal histograms are then computed and the relationship of the peak-valley points of the histograms is used for line segmentation.

The algorithm proposed by Arivazhagan et al. [14] first obtains an initial set of candidate lines from the piece-wise projection profile of the document. The lines traverse around any obstructing handwritten connected component by associating it to the line above or below. A decision of associating such a component is made by (i) modeling the lines as bivariate Gaussian densities and evaluating the probability of the component under each Gaussian, or (ii) the probability obtained from a distance metric. The proposed method is robust to handle skewed documents and touching lines.

Sesh Kumar et al. [15] presented a graph cut-based framework using a swap algorithm to segment document images containing complex scripts such as those in Indian languages. The text block is first segmented into lines using the

projection profile approach. The framework enables learning of the spatial distribution of the components of a specific script, and is able to adapt to a specific document collection such as a book. Moreover, they can use both corrections made by the user as well as any segmentation quality metric to improve the segmentation quality.

Yanikoglu and Sandon [16] first searched for the handwriting text line boundaries and then processed each text line in turn. The boundary between two text lines is not a straight line if the text lines are touching or overlapping. To find the exact boundary between two text lines, they searched for the rough boundary location by analyzing the horizontal pixel density histogram of the line. They then apply a contour following algorithm within that zone to find the exact boundary. The contour following algorithm is modified to operate within a rectangular zone. It is also changed to force a cut at the half-line, when necessary, in order to separate text lines that are touching and cannot be separated otherwise.

The algorithm in [17] localized lines by computing the horizontal histograms for the entire image at a couple of relevant skew angles; then the angle and position where the histograms have local minima were chosen as the location between lines. Calculation of the horizontal histograms was done using the traditional histogram calculation executable on DSPs. They refined the line finding algorithm by using a method to blur the words without affecting their location. They computed the pseudo convex hull of each word using the HOLLOW template. The horizontal histogram computed on the pseudo convex hulls is smoothed further via sliding window. Then, the local maximum of the histogram was located since these correspond to the location of the lines. Thresholds were specified to associate all maxima with one line.

In [18], a Cut Text Minimization (CTM) method of text lines segmentation is described. The CTM method finds a path or cut line in between the text lines to be separated which minimizes the text line pixels cut by the segmentation line, especially descenders from the upper line and ascenders from the lower line. The method attempts to track around ascenders or descenders to avoid cutting them. If the deviation is too great, the segmenter aborts and continues its forward path. A rough estimate of text line separations were first obtained using vertical projection histograms.

Li [19] proposed a new approach for text line detection by adopting a state-of-the-art image segmentation technique. They first convert a binary image to gray scale using a Gaussian window, which enhances text line structures. Text lines are extracted by evolving an initial estimate using the level set method. Preliminary experiments show that their method is more robust compared to a bottom-up connected component based approach. Examples show that the method is script independent. This has been qualitatively confirmed by testing it on handwritten documents in different languages, such as Arabic, English, Chinese, Hindi, and Korean. Statistical results show that the algorithm produces consistent results under reasonable variation of skew angles, character sizes, and noise.

A method based on a shortest spanning tree search is presented in [20]. The principle of the method consists of building a graph of main strokes of the document image and searching for the shortest spanning tree of this graph. This method assumes that the distance between the words in a text line is less than the distance between two adjacent text lines.

Almost all text line segmentation approaches assume that text lines are straight. However, projection-based methods can be extended to deal with curved text lines. Generally, a better result is achieved than with simple projection methods, but the merging of detected line segments of adjacent columns may be ambiguous. As such, it is still difficult to generate a reasonable result.

3.0 LINE SEGMENTATION ALGORITHM

As it is a significant stage, the line segmentation process has to be performed correctly to eliminate propagation errors to the next process. In our approach, tangent from the graph will be calculated to find other representations (i.e. tangent representation of either 1 or -1). Our approach is to eliminate false local minima that exist in our graph by collecting the number of 0s (black pixels) of corresponding rows as shown in Equation 1 and 2. These values will be stored in one array as our temporary storage.

$$h_k(r) = \sum_{c=0}^{N-1} I_k(r,c) \qquad (1)$$

$$h'_k(r) = N - h_k(r) \quad (2)$$

where N = number of columns, r = row, c = column, $h_k(r)$ = horizontal projection of 1 values, and $h'_k(r)$ = horizontal projection of 0s. Values in the array (histogram) will be used to calculate the new value (1 or -1) according to Equation 3; these values will be used to form another graph.

$$f(x) = \begin{cases} 1 & m>0 \\ -1 & m<0 \end{cases} \quad (3)$$

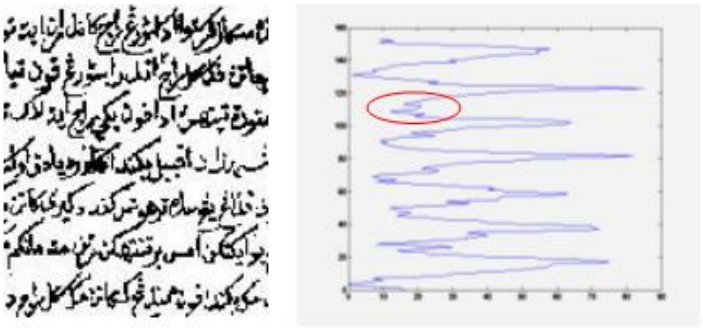
where m is $h'_{k+1} - h'_k$ and k is the corresponding row. The value of f(x) will be tested and will be given a new value if it fulfills the following rule.

Rule 1: if the number of adjacent -1 value does not exceed the constant value ***k*** *then it will be replaced by a new value i.e. 1*

The purpose for the Rule 1 above is to eliminate false local minima that disrupts the line segmentation process (with **k** = 3, that is, false local minima will not exceed value 3 in tangent tracing based on our experiment). The final step is to trace the changes of tangent in our new value and this will point to where the Valid Separation Point is. Figure 1 shows the raw image of an old manuscript.

Fig. 1: Old Jawi manuscript with size 1277 x 774 pixels

The first step in our experiment is to crop out our Region of Interest (ROI) from the image in Fig. 1 and convert it into a binary image which is shown in Fig. 2(a). This ROI clearly shows that there are overlapped characters between adjacent lines due to the way it is written. As it can be seen in Fig. 2(b) between lines 100 to 120, there are two false local minimum which is circled. In the next figure i.e. Fig. 3, it illustrates that when the graph in Fig. 2(b) is converted into tangent representation, it shows an active change of value from 1 to -1 (vice versa). These active changes might give inaccurate Vertical Segmentation Point if it is done without false local minimum elimination. This can be seen in Fig. 4 where the graph is more stable. The results of full line segmentations are shown in Fig. 5.



(a) (b)

Fig. 2: (a) Binary image of ROI of Old manuscript (b) Graph row versus no of 0's

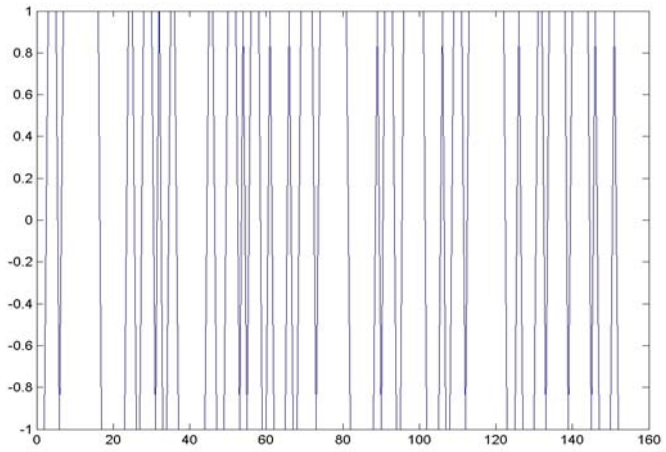


Fig. 3: New representation of tangent versus row (before elimination of false local minimum)

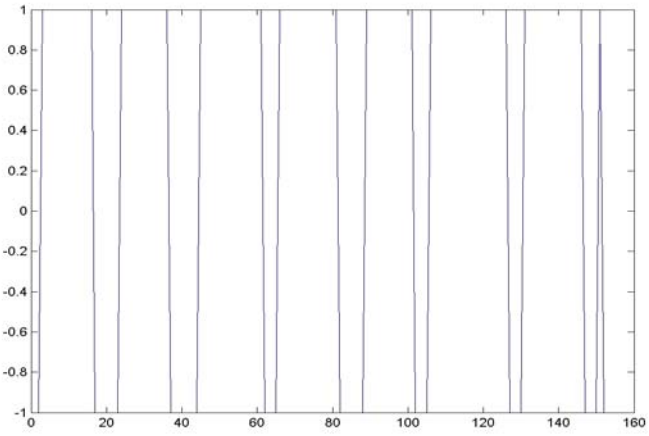


Fig. 4: Tangent versus row (after elimination of false local minimum)

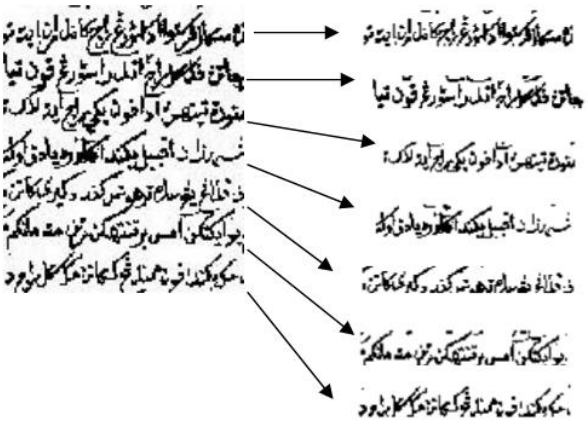


Fig. 5: Results of line segmentation

4.0 HARDWARE DESIGN

The hardware design will be implemented in a Field-Programmable Gate Array (FPGA). FPGA is an array of logic cells placed in an infrastructure of interconnections, which can be programmed at three different levels: (i) the function of the logic cells, (ii) the interconnection between cells, and (iii) the inputs and outputs. All three levels are configured via a string of bits that is loaded from an external source. FPGAs are highly versatile devices that offer the user a wide range of design choices. All modules of the line segmentation will be designed or modeled with VHDL. This language allows the simulation and adjustment of the circuit.

The system will be implemented on an ML401 evaluation platform which enables designers to investigate and experiment with features of the Virtex-4 family of FPGAs. The ML401 is a feature-rich and low-cost general purpose evaluation/development platform which provides easy access to resources available on the on-board Virtex-4 LX25 FPGA device. Supported by industry standard interfaces, connectors, and peripherals, the ML401 is a versatile development platform for multiple applications. It has a fairly powerful FPGA, type XC4VLX25-FF668-10 of the Virtex family made by Xilinx (www.xilinx.com).

This Virtex-4 FPGA contains 16 specialized global clock input locations for use as regular user I/Os if not used as clock inputs. This Virtex-4 device improves the clocking distribution by the use of 12 clock regions. The 8 Digital Clock Managers (DCMs) provide a wide range of powerful clock management features. The 4 Phase-Matched Clock Dividers (PMCDs) are one of the clock resources available in the Virtex-4 architecture. The Configurable Logic Blocks (CLBs) are the main logic resource for implementing sequential as well as combinatorial circuits. Logic resources available in all CLBs are 96 x 28 CLB array, 10,752 21 slices, 21,504 LUTs, a maximum of 168 Kb distributed RAM or shift registers, and 21,504 flip-flops. The XC4VLX25 has 10 usable I/O banks and one configuration bank.

Virtex-4 devices are configured by loading application-specific configuration data—the bitstream—into internal memory. Because Xilinx FPGA configuration memory is volatile, it must be configured each time it is powered up. The bitstream is loaded into the device through special configuration pins. The bitstream length for the Virtex-4 device is 7,819,520 configuration bits.

After studying several possible architectures, the line segmentation block diagram was designed. This line segmentation architecture can be divided in four entities: temporary input memory, histogram calculation, false minimum elimination, and temporary output memory. First, the system will load data into the temporary input memory. When the first 32 bit input is detected it will enable the histogram calculation process. If the histogram calculation detects drastic changes in the histogram, it will enable the false minimum elimination entity. Elimination process is done until it reaches the point of separation or segmentation point. Following its completion, it will activate the temporary input to accept the segmentation address. Then, it will start to transfer data into the temporary output for line segmentation. The interconnection between entities is shown in Fig. 6. Each entity's functionalities are illustrated in Table 1, and input output details are shown in Table 2.

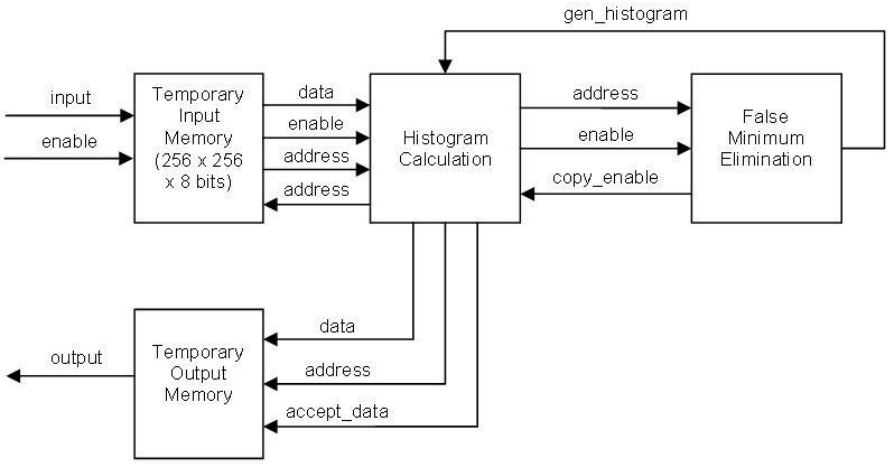


Fig. 6: Line Segmentation Block Diagram

Table 1: Entity functionalities

Entity	Function
Temporary input memory	Receive data loaded from the system. Enable the histogram calculation process when the first 32 bit input is detected.
Histogram calculation	Enable the false minimum elimination entity if the histogram calculation detects drastic changes in the histogram. Activate the temporary input to accept the segmentation address. Start to transfer data into the temporary output for line segmentation.
False minimum elimination	Perform elimination process until it reaches the point of separation or segmentation point.
Temporary output memory	Receive data from the histogram calculation entity for line segmentation.

Table 2: Entity Input and Output details

Input/ouput	Details
input	Jawi text image
enable	Enable the next entity
data	Data is sent to the next entity
address	Text line segment address
gen_histogram	Signal to start histogram generation
copy_enable	Signal to enable copy of data
accept_data	Signal for the entity to accept data
output	Text image segmented into lines

5.0 BENCHMARKING RESULTS

The proposed algorithm described in [12] was tested on 100 samples of text containing 1000 lines. The experiment results in 97% accuracy in correct line segmentation. Errors were caused by baseline-skew variability, overlaps between characters, and diacritical marks in the first column. The association of the diacritic symbols to a text line caused significant errors when the symbol was distant from the separating border line.

Tripathy and Pal [13] used 1627 text lines in single column document pages with different writing styles. Text lines segmentation accuracy is calculated by drawing boundary lines between two consecutive text lines. Then, from the computer's display, line segmentation accuracy was calculated manually. If all text lines are extracted correctly, segmentation accuracy is considered 100%. From 1627 lines tested, 984 lines were segmented correctly.

Experimental results in [14] show that on 720 documents including English, Arabic, and children's handwriting which consist of 11,581 lines, 97.31% of the lines were segmented correctly. On an experiment of over 200 handwritten images with 78,902 connected components, 98.81% of them were associated to the correct lines. Experiments were also conducted on 300 exam essays written on ruled line paper to test the robustness of the algorithm. The algorithm segmented with an accuracy of 96.3%. Furthermore, the segmentation algorithm proved language independence, with accuracy of 98.62% on 120 Arabic handwritten images. Most of the dots above or below a word were associated to the correct lines by the proposed algorithm.

Documents scanned from a Telugu book titled “Aadarsam" containing 256 pages printed in 1973 were used in [15]. The performance of segmentation is calculated using a segmentation quality metric. Segmentation corrections required on a new page are close to zero after adapting to the first 44 pages. The remaining 212 pages were segmented correctly.

The method in [18] was tested on 30 images from NIST special database comprising data in 34 text boxes from 2100 forms scanned at a resolution of 300 pixels / inch and saved in binary format. Text boxes in the form corresponding to a text paragraph of 52 words were abstracted for text line segmentation. The segmentation resulted in 183 text segmenting lines and 213 text lines in the images were correctly segmented into 176 lines with 96% accuracy.

Likforman-Sulem and Faure [7] tested their method on handwritten documents. Fluctuating text lines, sloped annotations, or annotations added between main lines were detected. Fluctuation combined with proximity of text lines may cause merged lines. The algorithm in [17] was tested on 10 pages of a handwriting database containing 7000 words from the LOB corpus (Lancaster-Oslo / Bergen) written by a single writer. The experiments were conducted using the MatCNN simulator in the Matlab software. The line segmentation algorithm correctly segmented each line in every page.

The algorithm in [19] was tested on more than 10,000 diverse handwritten documents in different scripts, such as Arabic, Hindi, and Chinese. During testing, if a ground-truth line and the corresponding detected line share at least 90% pixels, a text line is considered to be detected correctly. From a total of 2,691 ground-truth lines, their approach correctly detected 2,303 (85.6%) lines. At the text line level, the connected-component-based method performs significantly worse where only 951 (35.3%) text lines were detected correctly. Errors were caused by signatures, two adjacent text lines overlapping significantly, the correction in the gap between two lines, and the severe noise introduced during scanning.

The proposed text line detection method in [11] which uses a Block-Based Hough Transform approach was tested on unconstrained handwritten Greek documents using 20 document images taken from the historical archives of the University of Athens for which corresponding text line detection ground truth was manually created. A number of 450 text lines were detected in the images with 96.87% accuracy. Difficulties concerning the extraction of text lines include the variety of accents appearing above or under the text line body and the small difference in the skew angle in the text lines.

In [8], overlapped text lines and text lines where the interline distance is smaller than the intra line distance will cause segmentation errors. Only well separated text lines were correctly segmented. The algorithm in [18] which used a chain code representation was tested on text in church registers ranging 300 years. The algorithm was applied to images from 61 paragraphs in 7 pages. All 61 paragraphs consisted of 300 lines. The algorithm produced good results with constant values on different handwriting styles. 222 lines (90%) of 246 lines in 49 paragraphs containing six different handwritings were reconstructed correctly. The experimental results are summarized in Table 2.

Table 2: Experimental results

Author	Experiment data	Experiments	Accuracy	Segmentation errors
Zahour et. al [12]	1000 lines in 100 samples	Implemented in a C++ language on a 200MHz PC	97%	Baseline-skew variability, and overlaps between characters and diacritical marks in the first column. Association of diacritic symbols distant from the separating border line, to a text line.
Tripathy and Pal [13]	1627 lines in single column pages with different writing styles	Draw boundary lines between two consecutive text lines. Then, from the computer's display, line segmentation accuracy was calculated manually.	60%	When two consecutive words touch, or distance between two consecutive words is very small.
Arivazhagan et. al [14]	11,581 lines in 720 documents including English, Arabic and children's handwriting	The cut-through accuracy corresponds to the proportion of components correctly classified as overlapping components or cut-through error.	97.31%	A normal component, spanning across two or more lines or lying in between two lines of text
Sesh Kumar et al. [15]	Documents scanned from a Telugu book titled "Aadarsam", printed in 1973 containing 256 pages.	The performance of segmentation is calculated using a segmentation quality metric	Segmentation corrections required on a new page are close to zero after adapting to the first 44 pages. The remaining 212 pages were segmented correctly.	Not stated
Weliwitage et. al [18]	30 images from NIST special database in 34 text boxes from 2100 forms	Text boxes in the form corresponding to a text paragraph of 52 words were abstracted for text line segmentation.	96%	Very short text lines of not longer than one word and text lines not starting from left margin of the image and unclear separation of text lines with merging ascenders and descenders may be detected as an extra text line.
Likforman-Sulem et. al [7]	Unconstrained handwritten rough drafts, address blocks, letters and manuscripts.	Alignments found as text lines are crossed by a line, components belonging to the same line share the same identification number inscribed above their enclosing rectangles. Alignments which were	Not stated	When fluctuation is combined with proximity of text lines, merging lines may appear.

		found in the Hough domain, but invalidated in a second stage are crossed by a dashed line. Ambiguous components are inscribed in dashed rectangles.		
Tímár et. al [17]	10 pages of a handwriting database containing 7000 words from the LOB corpus (Lancaster-Oslo / Bergen) written by a single writer	Conducted using the MatCNN simulator in Matlab software	Correctly segmented each line in every page	Not stated
Li et. al [19]	More than 10,000 diverse handwritten documents Arabic, Hindi, and Chinese script	If a ground-truth line and the corresponding detected line share at least 90% pixels, a text line is considered to be detected correctly.	From 2,691 ground-truth lines, correctly detected 2,303 (85.6%) lines. At the text line level, only 951 (35.3%) text lines were detected correctly.	Errors were caused by two adjacent text lines overlapping significantly, signatures, the correction in the gap between two lines, and the severe noise introduced during scanning.
Louloudis et. al [11]	Unconstrained handwritten Greek documents using 20 document images taken from the historical archives of the University of Athens	Used Block-Based Hough Transform approach for which corresponding text line detection ground truth was manually created	450 text lines were detected in the images with 96.87% accuracy	Various accents above or under the text line body and the small difference in the skew angle in the text lines
Nicolas et. al [8]	Collection of handwritten French novelist Gustave Flaubert drafts		Only well separated text lines were correctly segmented.	Overlapped text lines and text lines where the interline distance is smaller than the intra line distance will cause errors.
Feldbach and Tönnies [9]	Text in church registers from 61 paragraphs consisting of 300 lines in 7 pages ranging 300 years.	Chain code representation	222 lines (90%) of 246 lines in 49 paragraphs containing six different handwritings were reconstructed correctly.	Serious errors occur when the difference between the reconstructed base and centre lines and the real lines was higher than the script size, if a text line was not found, or if an extra line was found at a wrong place.
Our approach	Scanned documents of “Hikayat Hang Tuah” manuscript from the 17th century	robust real time line segmentation using histogram projection	96 % accuracy	Inaccuracies occur due to severe overlapping between lines.

Experiments on our proposed line segmentation algorithm were conducted in the Matlab software for simulation purpose as shown in the last row in Table 2. The advantage of the fast approach achieved by the proposed algorithm compared to other techniques is that our algorithm avoids contour and boundary tracing procedures. Therefore, it increases productivity and performance by reducing the processing overhead. The algorithm is also easy to implement on hardware because of the simple mathematical calculations applied. This achieves our goal to produce a real-time Jawi text line segmentation hardware design.

6.0 CONCLUSION

Our fast algorithm compares favorably and shows highly accurate line segmentation although there are a few cut-off characters. These can be eliminated through a local analysis of every each line. We envisage the on-chip implementation of the algorithm will also mirror similar fast processing performance based on the simplicity of our approach.

REFERENCES

[1] T. Pudjiastuti, “Looking at Palembang Through Its Manuscripts”, in *Indonesia and the Malay World*, Vol. 34, No. 100, 2006, pp. 383 – 393.

[2] S. M. S. M. Omar, “Preservation Of Malay Manuscripts As A National Documentary Heritage: Issues And Recommendations For Regional Cooperation”, Sekitar Perputakaan (ISSN: 0127-1172), No. 33, 2001, pp. 5-11.

[3] A. Toru, “Jawi Study Group”, Islamic Area Studies in Japan, Annals of Japan Association for Middle East Studies (AJAMES), No. 20-2, 2005, pp. 399-404.

[4] A. Ymin, Y. Aoki, “On the Segmentation of Multi-Font Printed Uygur Scripts”, in *Proceedings of the 13th International Conference on Pattern Recognition*, August 1996, Vol. 3, pp. 215-219.

[5] A. Cheung, R. A. Ammar, M. Abdalla, “A Recognition-based Arabic Optical Character Recognition System”, in *Proceeding of Second IEEE Symposium on Computer and Communication*, July 1997, pp. 286-291.

[6] A. Amin, ”Recognition of Arabic Handprinted Mathematical Formulae,” *The Arabian Journal of Science and Engineering*, Vol. 16, No. 4B, October, 1991, pp. 532-542.

[7] L. Likforman-Sulem, C. Faure, "Extracting text lines in handwritten documents by perceptual grouping", *Advances in handwriting and drawing : a multidisciplinary approach*, C. Faure, P. Keuss, G. Lorette and A. Winter Eds, Europia, Paris, 1994, pp. 117-135.

[8] S. Nicolas, T. Paquet, L. Heutte, "Text Line Segmentation in Handwritten Document Using a Production System", *Proceedings of the 9th IWFHR*,Tokyo, Japan, 2004, pp. 245-250.

[9] M. Feldbach, K. D. Tönnies, "Line Detection and Segmentation in Historical Church Registers", *Sixth International Conference on Document Analysis and Recognition, Recognition*, September, 2001. pp. 743-747,

[10] L. Likforman-Sulem, A. Hanimyan, C. Faure, “A Hough based algorithm for extracting text lines in handwritten documents", *Third International Conference on Document Analysis and Recognition*, Vol. 2, August 1995, pp. 774-777.

[11] G. Louloudis, B. Gatos, I. Pratikakis, K. Halatsis,”A Block-Based Hough Transform Mapping for Text Line Detection in Handwritten Documents”, *Proceedings of the Tenth International Workshop on Frontiers in Handwriting Recognition*, La Baule, Oct. 2006.

[12] A. Zahour, B. Taconet, P. Mercy, and S. Ramdane, “Arabic Hand-written Text-line Extraction”, in *Proceedings of the Sixth International. Conference on Document Analysis and Recognition*, ICDAR 2001, Seattle, USA, September 10-13 2001, pp. 281–285.

[13] N. Tripathy and U. Pal. ,“Handwriting Segmentation of Unconstrained Oriya Text,” in *International Workshop on Frontiers in Handwriting Recognition*, 2004, pp. 306–311.

[14] M. Arivazhagan, H. Srinivasan, S. N. Srihari, "A Statistical Approach to Handwritten Line Segmentation", in *Proceedings of SPIE Document Recognition and Retrieval* XIV , San Jose, CA, February 2007.

[15] K.S. Sesh Kumar, A. M. Namboodiri, C.V. Jawahar, "Learning Segmentation of Documents with Complex Scripts", in *Fifth Indian Conference on Computer Vision, Graphics and Image Processing*, Madurai, India, LNCS 4338, 2006, pp.749-760.

[16] B. Yanikoglu and P. A. Sandon, "Segmentation of Off-line Cursive Handwriting using Linear Programming", *Pattern Recognition*, Vol. 31, No. 12, 1998, pp. 1825-1833.

[17] G. Tímár, K. Karacs, Cs. Rekeczky, "Analogic Preprocessing and Segmentation Algorithms For Offline Handwriting Recognition", *Proceedings of IEEE CNNA'02*, World Scientific 2002, pp.407-414.

[18] C. Weliwitage, A. L. Harvey, A. B. Jennings, "Handwritten Document Offline Text Line Segmentation", in *Proceedings of Digital Imaging Computing: Techniques and Applications*, 2005, pp. 184-187.

[19] Y. Li, Y. Zheng, D. Doermann, and S. Jaeger, "A new algorithm for detecting text line in handwritten documents," in *International Workshop on Frontiers in Handwriting Recognition*, 2006, pp. 35–40.

[20] I.S.I. Abuhaiba, S. Datta, M.J.J. Holt, "Line Extraction and Stroke Ordering of Text Pages", *Proceedings of the Third International Conference on Document Analysis and Recognition*, Montreal, Canada, 1995, pp. 390-393.

BIOGRAPHY

Zaidi Razak obtained his Bachelor's degree in Computer Science and Master's degree in Chip Design from University of Malaya. Currently, he is a lecturer at the Faculty of Computer Science and Information Technology, University of Malaya. His research areas include discrete wavelet, image processing, Jawi character recognition, and System on Chip (SoC) design. He has published a number of papers related to these areas.

Khansa Zulkiflee obtained her Bachelor's degree in Computer Science from University of Technology. Currently she is a research assistant at the Faculty of Computer Science and Information Technology, University of Malaya. Her research areas include image processing, Jawi character recognition, and System on Chip (SoC) design.

Rosli Salleh obtained his Bachelor's degree in Computer Science from University of Malaya. He then obtained his Master's degree in Data Communication Networking and PhD in Computer Science majoring both in Virtual Reality, Tele-surgery and Networking from University of Salford Manchester United Kingdom. He has also obtained CCNA professional qualifications. Currently, he is a lecturer at the Faculty of Computer Science and Information Technology, University of Malaya. His specializations include Bluetooth network, networking, and Virtual Reality. His current research interests are Bluetooth Scatternet Formation, Public Key Infrastructure, Network Security, Authentication Server, Virtual Simulation, Virtual Reality, and Laparoscopic Surgical Training.

Mashkuri Yaacob is currently the third vice-chancellor of Universiti Tenaga Nasional (Uniten). Mashkuri, who obtained his Bachelor's degree in Electrical Engineering from the University of New South Wales, Sydney, also holds master's and doctoral degrees in electronics and computer engineering from the University of Manchester in the United Kingdom, was named deputy dean of the Engineering Faculty in 1983 and later dean of the Computer Science and Information Technology Faculty in 1994. Mashkuri's career at UM culminated with his appointment as deputy vice-chancellor (academic) from June 2000 to May 2003. Mashkuri's main research interests are discrete wavelet, operating system, and integrated information.

Emran Mohd Tamil obtained his Bachelor's degree in Electrical-Robotic Engineering from University Technology Malaysia in and Master's degree in Information Technology from University Technology MARA. Currently, he is a lecturer at the Faculty of Computer Science and Information Technology, University of Malaya. His specializations are system and network and current research interests are embedded systems, network security, SCADA, and chip design.