

**PREPROCESSING AND SEGMENTATION OF HANDWRITTEN
ARABIC DOCUMENTS FOR WRITER-INDEPENDENT
AUTOMATIC RECOGNITION**

By

Ahmad Shehdeh Helmi Al-Hourani

Supervisor

Dr. Gheith Ali Abandah, Prof.

**This Thesis was Submitted in Partial Fulfillment of the Requirements for
the Master's Degree in Computer and Networks Engineering**

**School of Graduate Studies
The University of Jordan**

May, 2017

COMMITTEE DECISION

This Thesis/Dissertation (Preprocessing and Segmentation of Handwritten Arabic Documents for Writer-Independent Automatic Recognition) was Successfully Defended and Approved on _____

Examination Committee

Signature

Dr. Gheith Ali Abandah, (Supervisor)
Professor of Computer Science and Engineering

Dr. Khalid A. Darabkh, (Member)
Professor of Computer Engineering

Dr. Iyad F. Jafar, (Member)
Associate Professor of Computer Engineering

Dr. Ali M. Al-Haj, (Member)
Associate Professor of Computer Engineering
(Princess Sumaya University for Technology (PSUT))

DEDICATION

*To my parents and wife who always hoped and worked for the best, and without whom,
I would not have gotten this far today...*

ACKNOWLEDGEMENT

Gratitude is due to you, Prof. Gheith Abandah, for your guidance, knowledge, support, patience, and warm-heartedness.

TABLE OF CONTENTS

Subject	Page
Committee Decision	ii
Dedication	iii
Acknowledgement	iv
Table of Contents	v
List of Tables	vii
List of Figures.....	viii
List of Abbreviations	ix
Abstract.....	x
1 Introduction.....	1
1.1 Research Objectives	1
1.2 Motivation.....	2
1.3 Research Methodology.....	2
1.4 Thesis Structure.....	4
2 Background and Literature Review	5
2.1 Background	5
2.2 Literature Review.....	8
2.2.1 Recognition Systems	8
2.2.2 Ruled lines detection and removal	10
2.2.3 Segmentation.....	10
2.2.4 Skew.....	13
3 Investigating MADCAT Data Set	15
3.1 Existing Problems	15
3.1.1 Pepper noise	15
3.1.2 Vertical lines	16
3.1.3 External graphical elements	17
3.1.4 Writing errors	18
3.1.5 Lined paper.....	19
3.1.6 Text lines overlapping.....	21
3.1.7 Intra-word spacing	23
3.1.8 Slant	23
3.1.9 Skew.....	24
3.1.10 Bleed-through text.....	25
3.1.11 Varying text body.....	26

3.1.12	Irrelevant text or numerals.....	26
3.2	Nonexistent Problems and Unrequired Preprocessing Operations.....	27
3.2.1	Binarization	27
3.2.2	Height normalization.....	27
3.2.3	Line thickness variations/manipulations	27
3.2.4	Multiple text bodies.....	27
3.2.5	Special characters separation.....	28
3.2.6	Baseline detection.....	28
3.3	A Suggested Sequence for Document Recognition System.....	28
4	Requirements Outline	31
4.1	Selected Recognition Algorithm	31
4.2	Scope of the Thesis.....	34
4.3	Noise Removal	36
4.3.1	Performance of the algorithm.....	38
5	Ruled Line Removal.....	41
5.1	Problems to be Resolved	41
5.2	Approaches to Ruled Line Removal.....	42
5.3	Ruled Line Removal Algorithms Evaluation and Selection.....	46
6	Segmentation.....	50
6.1	Problems to be Resolved	50
6.2	Approaches to Segmentation.....	50
6.3	Segmentation Algorithms Evaluation and Selection.....	56
6.3.1	Line segmentation algorithm.....	61
6.3.2	Word segmentation algorithm.....	63
7	Resulting System.....	64
8	Conclusions, Recommendations, and Future Work.....	67
	References	70
	Abstract in Arabic	77

LIST OF TABLES

NUMBER	TABLE CAPTION	PAGE
Table 1	Ruled Line Detection and Removal Methods Summary	46
Table 2	Segmentation Methods Summary	57

LIST OF FIGURES

NUMBER	FIGURE CAPTION	PAGE
Figure 1	Typical OCR Steps (Saeed, 2014).	6
Figure 2	Samples of the documents used in the OpenHaRT evaluation.	8
Figure 3	(a) Sparse dots distribution. (b) and (c) Dense noise dots distribution.	16
Figure 4	(a) and (b) Ruled vertical lines and how they can affect text. (c) and (d) Noise-generated vertical lines.	17
Figure 5	Different undesired graphical elements.	18
Figure 6	Different types of writing errors.	18
Figure 7	Text on unlined area of a page.	19
Figure 8	Ruled line shape, close up (Kumar and Doermann, 2011).	19
Figure 9	Cases where inattentive line removal can deform words. Line width is comparable to text stroke width and baseline or other text elements can completely overlap with ruled line.	20
Figure 10	Complete overlapping between text and ruled line that can cause destruction of the word after ruled line removal.	20
Figure 11	Effects of overlapping lines.	21
Figure 12	Different types of line spacing in pages.	22
Figure 13	Intra-word space examples.	23
Figure 14	Slant examples.	24
Figure 15	Forms of skew.	25
Figure 16	Bleed-through text.	26
Figure 17	Irrelevant text example.	26
Figure 18	Suggested sequence for document recognition preprocessing system.	29
Figure 19	The steps of the selected recognition algorithm, JU-OCR2 (Abandah, et al., 2014).	32
Figure 20	Devised noise removal algorithm.	38
Figure 21	(a) Enlarged illustration of Figure 3(c) that shows the nature of the severe case of pepper noise. (b) The same portion of the image after applying a simple noise removal technique (c) Bleed-through text sample of Figure 16 after applying the same noise removal technique. (d) Effect of the noise removal algorithm on types of binarized faint lines, including faint ruled lines.	40
Figure 22	The proposed flow diagram of the preprocessing system.	64

LIST OF ABBREVIATIONS

ABBREVIATION	TERM
BLSTM	Bi-directional Long Short-Term Memory
CITlab	Computational Intelligence Technology Lab
CTC	Connectionist Temporal Classification
DSCC	Directional Singly Connected Chain
HCR	Handwritten Character Recognition
HMM	Hidden Markov Model
HPP	Horizontal Projection Profile
IFN/ENIT	Institut für Nachrichtentechnik/ Ecole Nationale d'Ingénieurs de Tunis (Institute of Telecommunications Engineering/ National School of Engineers of Tunisia)
JU-OCR2	The selected character recognition system (Abandah, et al., 2014)
KdK	Kabinet der Koningin (The Cabinet of the Dutch Queen)
LDC	Linguistic Data Consortium
LOG	Laplacian of a Gaussian
MADCAT	Multilingual Automatic Document Classification Analysis and Translation
MRF	Markov Random Fields
NIST	National Institute of Standards and Technology
OCR	Optical Character Recognition
OpenCV	Open Computer Vision Library
OpenHaRT	Open Handwriting Recognition and Translation Evaluation
PAW	Part of Arabic Word
PDE	Partial Differential Equation
PSL	Piece-wise Separating Lines
RNN	Recurrent Neural Network
SVM	Support Vector Machine
VPP	Vertical Projection Profile
WER	Word Error Rate
XML	Extensible Markup Language

**PREPROCESSING AND SEGMENTATION OF HANDWRITTEN ARABIC
DOCUMENTS FOR WRITER-INDEPENDENT AUTOMATIC RECOGNITION**

By

Ahmad S. Al-Hourani

Supervisor

Dr. Gheith A. Abandah, Prof.

ABSTRACT

In this thesis, we selected an excellent OCR system (JU-OCR2) that handles word-level recognition and augmented it with a preprocessing system in order to extend its scope to document image recognition. Therefore, we have investigated and identified the problems that could face such preprocessing system by studying a representative set of Arabic handwritten document images, the MADCAT set. These problems include pepper noise, ruled line, skew, line and word segmentation, bleed-through text, slant, external graphical elements, writing errors, and others. We identified the most common issues to be discussed in this thesis. These problems include ruled line removal, line and word segmentation, skew, and noise removal. We then reviewed approaches to resolve these common problems from the existing literature. The approaches were then subjected to a set of selection criteria designed for compatibility with JU-OCR2 and for achieving best results. All relevant approaches in literature were assessed. As a result of the selection process, three algorithms for ruled line removal, line segmentation, and word segmentation were selected. The selected ruled line removal and line segmentation algorithms contained built-in skew removal procedures. We have also devised an algorithm to remove pepper noise and bleed-through noise. We came up with modifications and recommendations to construct the desired preprocessing system that fits the selected recognition system.

1 INTRODUCTION

Optical character recognition (OCR) is one of the most successful applications in the fields of artificial intelligence, computer vision, and pattern recognition (Renjini and Rubeena, 2015). OCR is useful in many applications such as banking, signature verification, automatic mail classification, product identification, and many other fields. Benefits of OCR for businesses, as summarized in (Alginahi, 2013), are that the data in document images become editable, searchable, and more accurate, and document classification becomes possible. Digitizing and running OCR on documents minimizes data entry and saves physical space.

In this thesis, we select a recent OCR algorithm that achieved impressive and competitive results in recognizing Arabic words, namely JU-OCR2 (Abandah, et al. 2014) and augment it with preprocessing algorithms in order to extend its recognition scope to include document images and to meet the output standards set by OpenHaRT evaluation standard (NIST, 2013b). Preprocessing includes noise removal, ruled line detection and removal, and segmentation. Standard algorithms for each preprocessing step are used and combined with the solution.

We investigate the prominent preprocessing problems, survey the available approaches to each preprocessing problem, evaluate each approach with respect to JU-OCR2 suitability, and provide recommendations and improvements to what is best-suited to be used in our preprocessing system.

1.1 Research Objectives

The main objectives of this study can be summarized as follows:

1. Investigating the requirements and problems of MADCAT to identify the scope of our preprocessing system.

2. Investigating the available preprocessing and segmentation techniques (e.g., ruled-line removal, segmentation, and skew detection and correction).
3. Compare available approaches and select the techniques to be involved in the document image handwritten character recognition (HCR) system.

1.2 Motivation

Although Arabic is the second most widespread script, with over 300 million native speakers (Saeed, 2014) and around 27 languages use its alphabet, including Arabic, Pashto, Persian, Kurdish, Urdu (Lewis, 2009), making Arabic script the third most widely used script (with Latin first and Chinese second) (Saeed, 2014), research in Arabic OCR is humble when compared to its counterpart in Latin script, and little work was done to comprehensively characterize and analyze Arabic handwriting (Abandah and Khedher, 2009).

Arabic writing has barely changed over a long time. Thus, the availability of full Arabic document images HCR system will, in addition to all the aforementioned benefits of OCR, enable automatic searching and reading of over three million historical Arabic manuscripts, thus increasing the availability of their content (Parvez and Mahmoud, 2013).

This thesis is dedicated to help building a full HCR system starting with an accurate system for handwritten words recognition and building the required services for it to extend its scope to full document HCR as per the OpenHaRT evaluation standard.

1.3 Research Methodology

We perceive that, by following the devised research methodology steps below, we will be able to identify, evaluate, and select the proper preprocessing algorithms that comprise our preprocessing system:

1. **Investigation of MADCAT/ OpenHaRT requirements and data formats:** since there is no current investigation to identify the problems that one need handle in a document preprocessing system, we are the first to investigate the requirements and data formats of MADCAT/ OpenHaRT to form a complete knowledge of the minimum preprocessing-related problems that need to be resolved prior to feeding the segmented words to JU-OCR2 algorithm. We determine the problems that are expected to be eliminated or mitigated to specify the exact capabilities needed from the preprocessing system.
2. **Investigating and surveying the available preprocessing and segmentation techniques:** We investigate available preprocessing techniques (i.e., ruled-line removal and segmentation) related to the problems that must be resolved as per MADCAT/OpenHaRT database. In literature, each problem was processed using different approaches. Only the top-cited papers or the most recent ones are considered in this thesis for selection. This raises the par and quality of the approaches considered for selection.
3. **Selection of preprocessing techniques:** To achieve the best overall results without functionality duplication, we form a comprehensive preprocessing solution by selecting a set of preprocessing techniques depending on the principle of operation, function, services, and reported accuracy and time consumption parameters of the surveyed algorithms. Specific selection criteria for each preprocessing category are established in order to determine which preprocessing technique optimally meets the required capabilities at minimum cost. In this phase, the selected algorithms may be kept intact,

modified, or even some novel techniques may be proposed to ensure best attainable results.

4. **Performance and compatibility evaluation:** The selected preprocessing solutions are subjected to performance evaluation by itself and after combining it with JU-OCR2 algorithm. Concentration is mainly put on the error rate and time consumption.

1.4 Thesis Structure

The remaining of this thesis is structured as follows: Chapter 2 provides the background to OCR systems and surveys the related literature. Chapter 3 investigates the problems present in the MADCAT data set and identifies the problems that do not exist in this data set and thus need not to be included in this thesis. Chapter 4 outlines the requirement needed from the preprocessing system. Chapters 5 and 6 discuss the available approaches to various preprocessing problems and select the approaches best-suited for JU-OCR2 and MADCAT. Chapter 7 discusses the resulting system after selection. In Chapter 8, conclusions and future work are drawn and discussed.

2 BACKGROUND AND LITERATURE REVIEW

2.1 Background

Optical character recognition (OCR) attempts to mimic human reading. It has been an active research area since the development of digital computers. It is inherently complex and utilizes multiple fields of study like machine vision, pattern recognition, and artificial intelligence (Alginahi, 2013). Handwritten character recognition (HCR) is a subcategory of OCR that is concerned with the recognition of handwritten words and documents. Latin inscription OCR systems started as early as mid-1940s (Alginahi, 2013) and commercial OCR systems for Latin inscription appeared in the 1950s. Arabic text recognition has begun in 1975. There are few commercial printed Arabic text OCR systems but there is no accurate commercial Arabic HCR system yet (Parvez and Mahmoud, 2013).

A typical OCR system is comprised of a number of phases, see Figure 1. The first step is image acquisition, which can be achieved using any digital image capturing device. The second phase is pre-processing in which the non-text elements are removed, the digital image is binarized, morphological operations are performed, and slant/skew correction techniques are applied to improve text quality and facilitate text handling. In the segmentation phase, the document image is divided at various levels; depending on the requirements of the feature extraction phase. Segmentation may be at block, line, word, sub-word, character, or sub-character/grapheme levels. Feature extraction and classification phases are the core of OCR/HCR in which discriminative features of the segmented parts are extracted to provide an interpretation of segmented text. The optional step of post processing can enhance the recognition accuracy by verifying the recognized text using syntax and semantic analysis (Saeed, 2014).

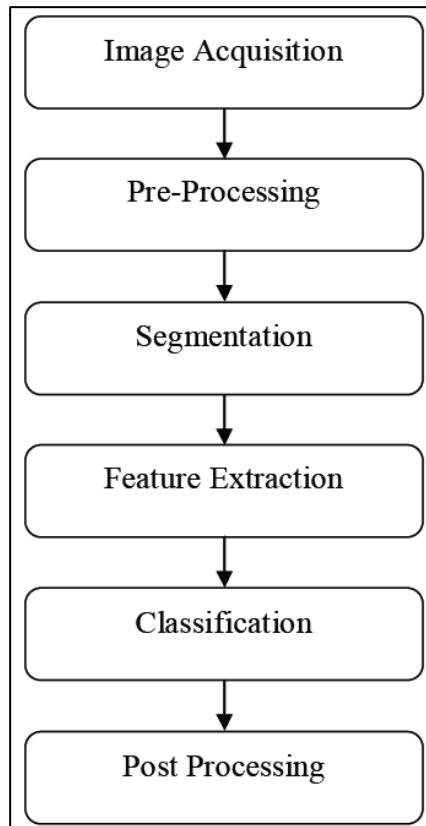
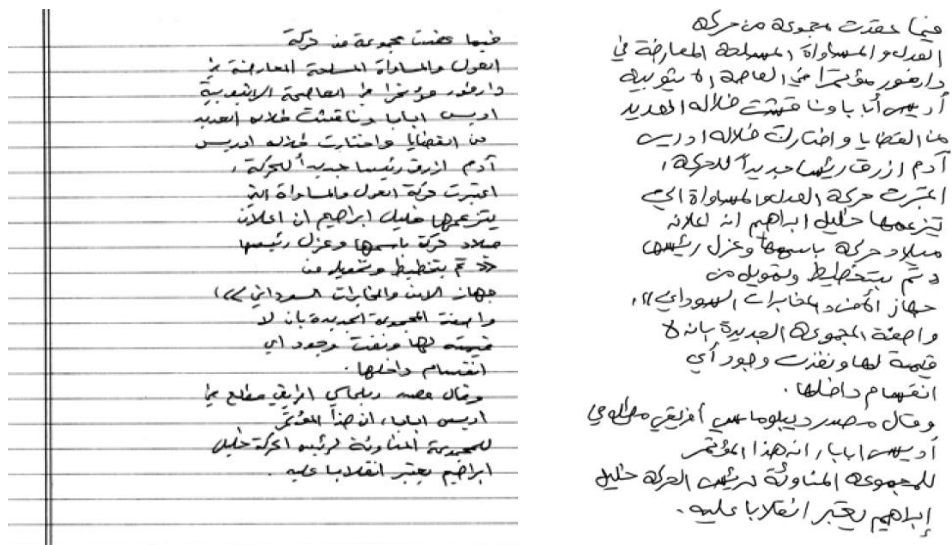


Figure 1. Typical OCR Steps (Saeed, 2014).

Offline handwritten recognition of cursive text can be, by far, the most difficult type of OCR since it includes all the problems from which an OCR system may suffer. Historically, OCR started with printed text recognition due to its relative simplicity, but was extended in the past few decades to HCR when it became achievable. Complexity of HCR is mainly caused by the large writer-to-writer style variation, overlapping of characters, and cursive nature (Saeed, 2014). Online recognition is more accurate than offline recognition techniques because the representation of data utilizes the time function and the chronological order of the strokes is taken into account (Manimurugan, et al., 2014). Also, preprocessing of the text image is not required and the running text length is usually shorter. To appreciate the gravity of cursive text impact on accuracy; some methods enjoying 100% accuracy over separate-character written documents fall to 60% accuracy over cursive-written text (Manimurugan, et al., 2014).

In order to advance the research of Arabic OCR/HCR, many efforts were made in various related fields like creation of standard databases, specification of evaluation criteria, and organization of workshops and contests. Many of the recent Arabic OCR studies were based on the IFN/ENIT database (Pechwitz, 2002), which is comprised of 26,459 binary images of 937 handwritten names of Tunisian towns and villages inscribed by 411 writers (Saeed, 2014). A database for document images was created by the Linguistic Data Consortium (LDC) and placed at the disposal of the Open Handwriting Recognition and Translation Evaluation (OpenHaRT) competition. It contains multiple instances of supervised handwriting for many document images each scribed by different writers. Each document image was subject to manual ground truthing and is saved with its ground truth in an extensible markup language (XML) file as the standard for evaluation (NIST, 2013b).

The Multilingual Automatic Document Classification Analysis and Translation (MADCAT) program holds the biennial evaluation event OpenHaRT, in which complete Arabic document images recognition and optionally translation systems are evaluated and ranked according to accuracy and complexity (NIST, 2013a). Figure 2 shows samples of the documents used in the OpenHaRT evaluation.



فيما عقدت مجموعة من حركة العدل والمساواة المسلحة المعارض في دارفور مؤتمرا في العاصمة الإثيوبية اديس أبابا وناقشت خلاله العديد من القضايا واختارت خلاله ادريس ادم ازرق رئيسا جديدا للحركة، اعتبرت حركة العدل والمساواة التي يتزعمها خليل ابراهيم انه اعلان ميلاد حركة باسمها وعزل رئيسها «> تم بتخطيط وتمويل من جهاز الامن والمخابرات السوداني» راجعه المجموعة الجديدة بان لا قيمة لها ونفت وجود احيى انقسام داخلها وقال مصدر دبلوماسي انريقي مطلع في اديس ابابا، ان هذا المؤتمر للمجموعة المناوئة لرئيس الحركة خليل ابراهيم يعتبر انقلابا عليه .

يتبعه عدد من مجموعات من حركة العدل والمساواة المعارضة في دارفور مؤتمرا في العاصمة الإثيوبية اديس أبابا وناقشت خلاله العديد من القضايا واختارت خلاله ادريس ادم ازرق رئيسا جديدا للحركة (اعتبرت حركة العدل والمساواة التي يتزعمها خليل ابراهيم ان اعلان ميلاد حركة باسمها وعزل رئيسها «> تم بتخطيط وتمويل من جهاز الامن والمخابرات السوداني» راجعه المجموعة الجديدة بان لا قيمة لها ونفت وجود احيى انقسام داخلها .

Figure 2. Samples of the documents used in the OpenHaRT evaluation.

It is worthwhile noting that preprocessing of document images normally enhances the accuracy of HCR as handwritten text is more susceptible to noise and normalization issues, hence noise removal, baseline detection, skew and slant detection and correction are the set of the tasks done in the preprocessing stage of OCR (Saeed, 2014). However, segmentation tends to pull accuracy back due to the complexity and difficulty of segmentation due to the cursive nature of Arabic writing, touching letters, and variations in writer styles.

2.2 Literature Review

In this section, we review the mainstream literature regarding recognition, and other preprocessing approaches.

2.2.1 Recognition Systems

Parvez and Mahmoud (2013) surveyed the advances in Arabic HCR in the period 1975-2005. They discussed that preprocessing of document images may contain several steps, not all of them are applied in each Arabic HCR method. Generally, preprocessing may start with data representation, which is sometimes overlooked or moved to the

feature extraction phase, where a more concise representation of text is offered. Text representation is improved using any combination of skeletonization (thinning), contour (border) detection, rotation, shearing, height normalization, and width normalization and rethickening of the skeleton is used as deemed required by the algorithm designers.

The Computational Intelligence Technology Lab (CITlab) submitted the CITlab ARGUS for Arabic handwriting recognition (Leifert, et al. 2013) for evaluation at OpenHaRT 2013. The system has simple preprocessing algorithms to detect/correct the skew/slant of the line and to normalize the height of the text to 80 pixels above the estimated baseline and 60 pixels below it. The system then uses an improved version of an existing recurrent neural network (RNN) (Graves and Schmidhuber, 2009). The RNN does not require segmentation of the input text image as it is processed on the pixel level. Viterbi decoding and a dictionary are used to determine the most probable character sequence. The system was affected by the authors' lack of knowledge of Arabic.

Ait-Mohanad and Paquet (2013) proposed another complete recognition system. The system starts with image enhancement and binarization algorithm taken from (Sauvola, 2000). De-skew, de-slant, and height normalization algorithms are then applied to the text. The enhanced text image is input, without segmentation, to a sliding window feature extractor that searches for 128 features at each position. The features are then fed to a hidden Markov model (HMM) that uses a 60,000-word dictionary and a language model to provide a set of probable answers (hypotheses). A graph that is comprised of all possible hypotheses is created. A Viterbi searching algorithm finds the most probable hypothesis. To enhance the results, the authors propose a novel combination framework that takes the outputs of several HMM based recognition

systems, combine them into a single network, and then selects the best word sequence hypothesis.

2.2.2 Ruled lines detection and removal

Line processing is used in form/invoice processing, engineering drawing processing, layout processing, musical note recognition, and other applications (Chen and Lopresti, 2014). Lined paper introduces problems to the feature extraction phase since it distorts the shape of the segmented unit.

Some works in literature included removal of the ruling line without reconstructing the original image or the text strokes (Kavallieratou, et al., 2011), some other works only detected ruled lines (Chen and Lopresti, 2014) and others detected and removed ruled lines then regenerated the text strokes to reduce the distortion generated by the line removal process (Cao, et al., 2009).

Different approaches have been discussed in the literature to detect and remove lines. Some approaches used the run length analysis (Blumenstein, et al., 2002), horizontal projection profile (Saleem, et al., 2009) and (Arvind, et al., 2007), template matching (Cao and Govindaraju, 2009), and stochastic approach using the HMM (Zheng, et al., 2005).

2.2.3 Segmentation

Segmentation, as discussed earlier, can be done at various levels and due to the complexity of handwriting and cursive text, many segmentation techniques have emerged (Parvez and Mahmoud, 2013).

Literature publications seem to have agreed on some terms regarding word processing and used other terms in different directions. Al-Dmour, et al. (2014) identified four terms in word processing: word segmentation, where an image of a word

is segmented into its forming characters, word separation or extraction, where we separate a line into word images, word spotting where a queried word is located in a set of document images, and word recognition where a word image is converted into text.

Various techniques have been used in order to achieve segmentation at different levels. They can be generally categorized as projection-based, Hough-based, or smearing-based. Projection-based methods excel in segmenting printed text but face major issues in handwritten text, due to text overlap, text orientation variations, and other factors (Kumar, et al., 2010) and (Louloudis, et al., 2009).

Projection-based algorithms are implemented either by taking the horizontal projection histogram and specifying its minima as possible points for segmentation or taking both horizontal and vertical projections. This kind of algorithms enjoys simplicity but suffers poor accuracy and robustness due to significant irregularities in cursive handwriting.

Projection profile approaches either processes the whole document (Manmatha and Rothfeder, 2005) or divides the document into vertical stripes and runs projection profile for each stripe independently and then processes the results of each stripe projection profile analysis (Arivazhagan, et al., 2007). The local minima of the projection profile are usually considered for segmentation points for line or word segmentation.

Hough-based approaches can handle variations in text orientation but their performance degrades if the text baseline is not straight (Yanikoglu and Sandon, 1998). In Hough transform approaches, certain points are selected from the original image (e.g., (Likforman-Sulem, et al., 1995) used connected components (CCs) centroids,

while (Pu and Shi, 1998) CCs minima points). The best-fit lines of those points are extracted as the lines comprising the document.

Smearing or grouping-based algorithms handle complicated layouts but are sensitive to text overlap since they use connected component to identify and link components of words or lines. In one of these approaches (Shi, Govindaraju, 2004), text is smeared and pixel-wise fuzzy run-length is applied which measures how far can one along the horizontal direction see if standing on a pixel. The fuzzy run-length calculation generates a greyscale image using which image is segmented.

Some grouping techniques like the one in (Zahour, et al., 2001) are based on the connected components analysis to segment the image where each connected component is identified. Information of each component like the component size, height, centroid, and corner pixels are used to filter text lines and words out of the original image.

Other approaches have been discussed in literature like seam-carving, which is originally used to scale the images without harming the most significant content by identifying the routes with least energy/information “seams” and use them as separation lines (Garz, et at., 2012). These algorithms may split ascender or descenders or may face issues in assigning secondary components to their primary letters.

Contour-based algorithms, taking advantage of the way Arabic letters connect and some morphological features of the Arabic script, use the skeleton or contour of the handwritten text to determine segmentation points depending on the criteria they choose (Alginahi, 2013).

In the segmentation validated by recognition techniques, the text contour or skeleton is over-segmented then fed to a recognition algorithm that is used to determine

the best locations for segmentation. The reported results of such algorithms are too inferior to be used in real-life applications (Parvez and Mahmoud, 2013).

In special-purpose segmentation, the segmentation of two touching numerals where the bounding box of the numerals is divided into two boxes iteratively and fed to a recognition algorithm. The iterations results are subject to classification to determine the most probable result (Parvez and Mahmoud, 2013).

Regarding word segmentation, there are two main approaches. The first calculates a metric between connected components (e.g. Euclidian distance, convex hull metric, or bounding box metric) in the same line and then establishes a threshold to classify distances as intra-word spaces and inter-words spaces. The other approach involves considering the distance classification as a task of the text line recognition and leaving it to the text line recognizer that may use HMM for instance to discriminate inter- and intra-word spaces (Luthy, et al., 2007).

By far, explicit segmentation can be a source of error in HCR systems (i.e., line and word level segmentation). Some systems which are based on HMM use implicit segmentation thus reducing the possible error due to segmentation. However, these systems are out of the scope of this study (Parvez and Mahmoud, 2013).

2.2.4 Skew

When the document is fed to the scanner manually or by mechanical means, skew of the scanned document is inevitable. Approaches to skew estimation algorithms can be classified into analysis of projection profiles approach, principal component analysis approach, and connected components clustering approach. In analysis of projection profiles (Liolios, et al., 2002), the document undergoes a computationally expensive and document layout sensitive procedure through which it is rotated multiple times and at

each time the projection profile is calculated to determine the skew angle. The principal component analysis determines the most significant eigenvector that leads to the skew angle of the text body (Smith, 2002). This method cannot eliminate outliers' effects. The connected components clustering method assumes that the distance between words in the same line are much smaller than that between words on different lines, thus 'inflating' the connected elements will cause each line words to connect to each other causing the line to become a single unit. Approximating the line to an ellipse and estimating its major axis will yield the skew angle of the line. This method provided the best accuracy among all three and took moderate time to execute (Sarfraz, et al., 2007).

3 INVESTIGATING MADCAT DATA SET

After acquiring the MADCAT training sets, an extensive review of the files was performed to search for possible problems available in these Arabic handwritten documents. It is noteworthy to mention that no similar investigation was found in the reviewed literature.

Regarding the required data format, the training sets contain a document that completely describes the required output data format.

The following sections summarize the findings of the aforementioned investigation.

3.1 Existing Problems

This section describes the problems that were found in the study database.

3.1.1 Pepper noise

During the image acquisition and binarization phases, noise dots may be produced over the scanned document. The noise dots concentration varies from one document to another. In some documents, the noise dots are at minimum where so few or none may exist as in Figure 3(a). In other documents, noise dots may be densely scattered all over the document where they even could cluster and form larger bodies as in Figure 3(b) and Figure 3(c).

Mostly, even smallest text elements are much larger than the largest continuous bodies the noise dots can create. This can be exploited to overcome this issue. But care should be taken to account for various text sizes, writing tip widths, and low dpi/resolution of the image.

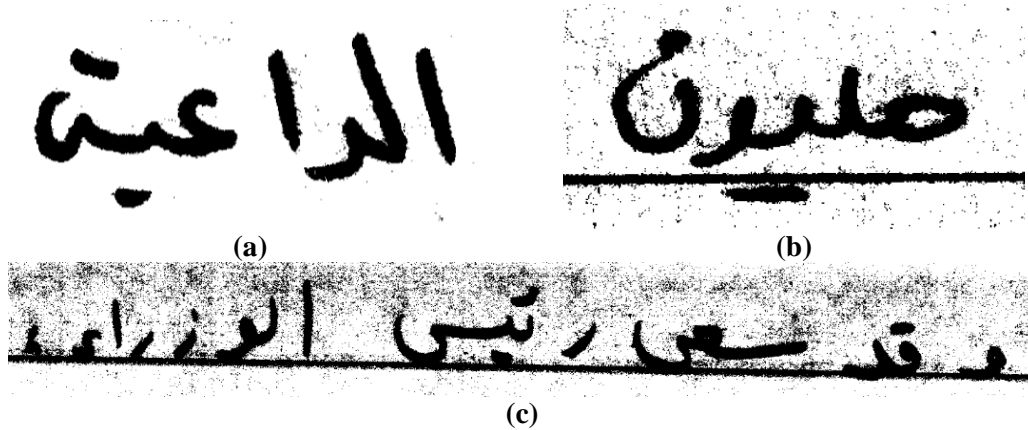
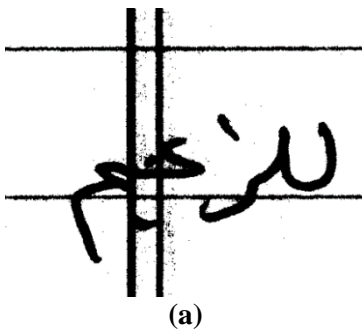


Figure 3. (a) Sparse dots distribution. (b) and (c) Dense noise dots distributions.

3.1.2 Vertical lines

There are different types of vertical lines. They can be mainly categorized into two categories depending on the source. The first, and the more influential, is the ruled vertical lines; that is, the vertical lines which are imposed by the nature of the page on which the scribe wrote the text. The number of adjacent ruled vertical lines range from one to three lines. Furthermore, ruled vertical lines may intersect with the handwritten text and distort it as shown in Figure 4(a) and Figure 4(b).

The other type includes vertical lines generated during the image acquisition phase. Those happen in some copying/scanning machines that tend to create faint lines in the acquired images. Another source of the faint lines is the shadow of the edges of the scanned pages. By nature, these lines are faint and mostly discontinuous. Figure 4(c) and Figure 4(d) illustrate this issue.



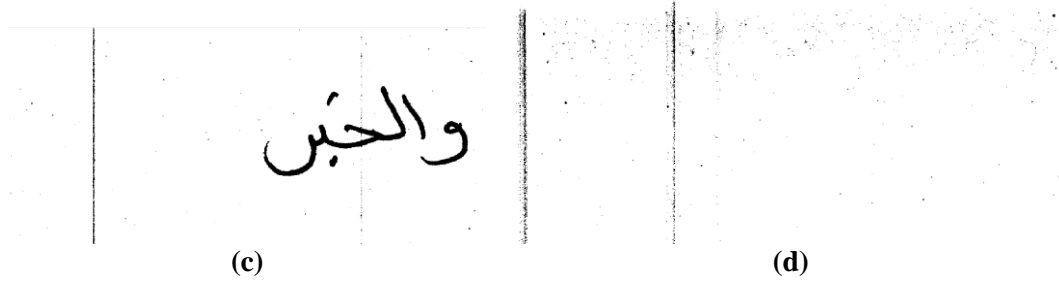


Figure 4. (a) and (b) Ruled vertical lines and how they can affect text. (c) and (d) Noise-generated vertical lines.

3.1.3 External graphical elements

The various paper types available in the MADCAT training sets and in the real life introduce various non-textual shapes that need to be filtered prior the recognition phase. Depending on the paper source and image acquisition conditions, various shapes may be generated, as in Figure 5.

In case these elements are fed to recognition, they may be erroneously recognized as punctuation marks and separate letters. Also, these shapes can in some cases distort the vertical and horizontal histograms of the documents significantly, as in the first and last examples in Figure 5. In some extreme cases, these elements may contain higher black dots density than the main text body. This needs to be considered if main text body is to be detected.

Those shapes are usually on the edge of the page where humans do not tend to write text. This creates a significant distance between the main text body and those elements that can be utilized to mitigate the effects of those elements on the segmentation and recognition phases.

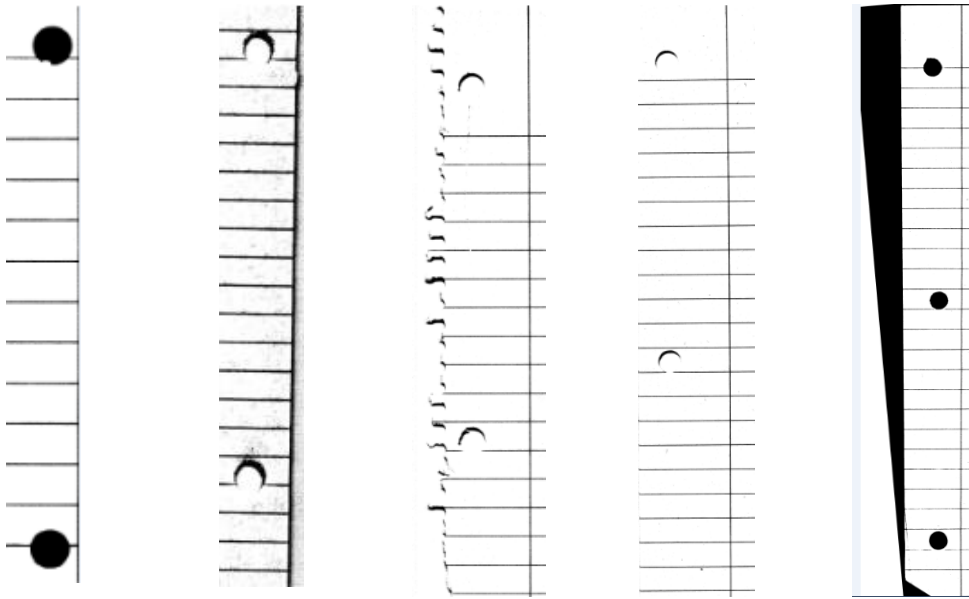


Figure 5. Different undesired graphical elements.

3.1.4 Writing errors

The ground truth for MADCAT training sets identify the errors made by the scribes when copying the text. These errors better be identified and sorted out by the recognition system. Some errors can be easily identified as clusters of ink or as shapes with too many intersections while others are harder to find as they only strike out the wrong word with a single line. Writing errors that are not struck out or scribbled will only cause wrong recognition of the word and may only be detected by dictionary word matching or even context matching. Figure 6 shows examples the above mentioned writing errors.

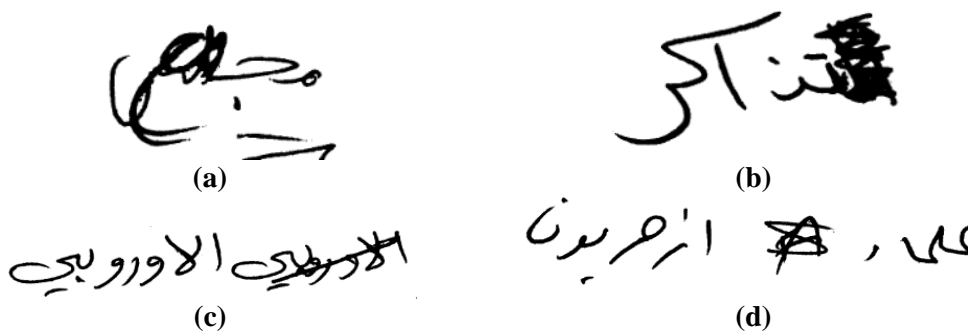


Figure 6. Different types of writing errors.

3.1.5 Lined paper

Lined paper is common for notebooks. Different ruling types are available in real life like narrow-ruled, college-ruled, law-ruled, and journal. Many lined paper types have vertical lines in different positions of the page. Ruled line pages usually comply with the following conditions:

- A ruled line, horizontal or vertical, usually extends from one end of the page all the way to the other.
- Horizontal lines are parallel to each other and perpendicular to vertical lines. This is always true unless some distortion occurred to the page in the image acquisition stage.
- If the page is lined, handwritten text is aligned to the lines and suffers no writer skew, even though text could be on unlined spaces of the page like in Figure 7.

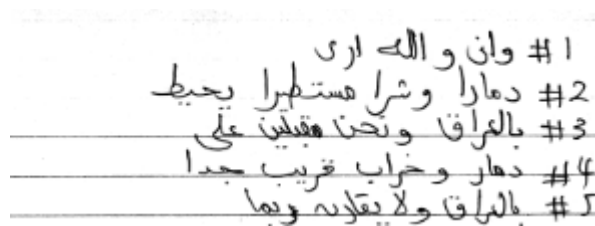


Figure 7. Text on unlined area of a page.

- A ruled line, close-up, is not a perfect rectangle, instead, as shown in Figure 8, it is a rectangle-like shape with variable widths and many imperfections including complete line breaks at some points. These disfigurements are created by the inconsistencies encountered along the processes of ruled line printing, capturing, and binarization.

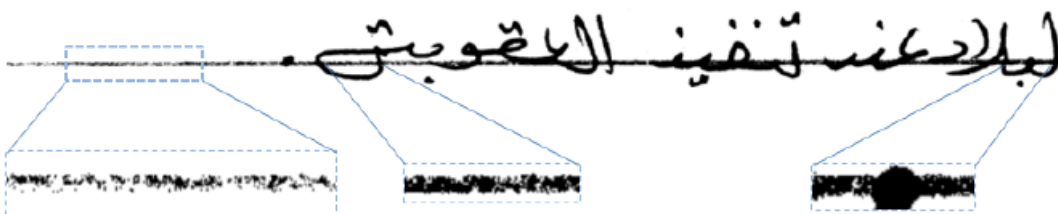


Figure 8. Ruled line shape, close up (Kumar and Doermann, 2011).

Sometimes ruled lines are of the same color – especially if the image is binarized, thickness and orientation of text and thus can significantly overlap with text. This combined with the characteristic horizontal baseline of Arabic handwriting makes inattentive line removal capable of causing severe deterioration to words. This is illustrated in Figure 9.

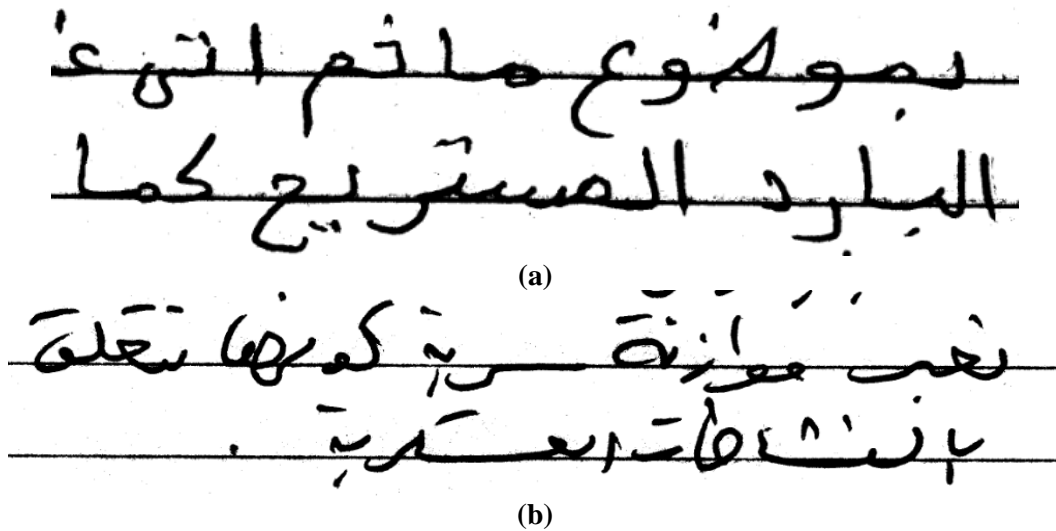


Figure 9. Cases where inattentive line removal can deform words. Line width is comparable to text stroke width and baseline or other text elements can completely overlap with ruled line.

In some cases, the problem is much more severe that words can be destroyed after ruled line removal. This normally happens when complete text-ruled line overlapping occurs. Examples of such cases are shown in Figure 10.

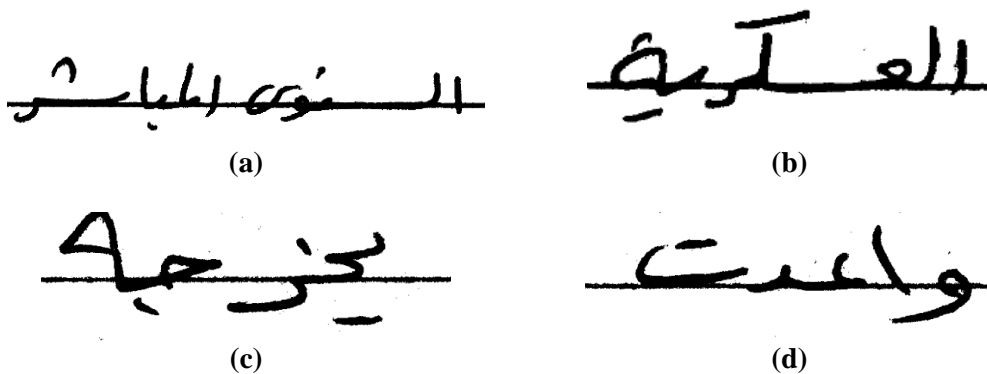


Figure 10. Complete overlapping between text and ruled line that can cause destruction of the word after ruled line removal.

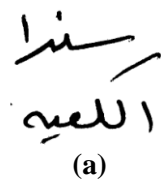
The effect of ruled line removal is expected to be minimal when ruled line width is considerably thinner than the text line stroke.

3.1.6 Text lines overlapping

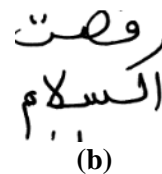
Irregularity and inconsistency are common traits of handwriting. Such traits create problems that we would not face in printed text such as text lines overlapping. That is; a straight line separating consecutive text lines cannot be found. This problem would have a major impact on the performance of the word segmentation, grapheme separation, and recognition unless an efficient and deterministic approach is followed to solve it.

The effects of lines overlapping include the following:

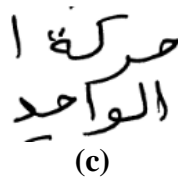
- Some word elements like strokes and secondary components may be mistaken to be belonging to another line as in Figure 11(a) where the upper stroke of “kaf” can be mistaken to be the letter “ra” for the upper word thus erroneously recognizing both words. Figure 11(b) shows a similar case but contrariwise.
- Sometimes overlapping produces connected elements between two consecutive lines as in Figure 11(c) and Figure 11(d). In such cases, a point of disconnection should be specified.



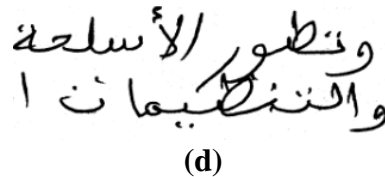
(a)



(b)



(c)



(d)

Figure 11. Effects of overlapping lines.

In terms of inter-line spacing, pages can be divided into three categories:

1. Distant lines, where all the lines in the pages are visibly segregated and easily discriminated as in Figure 12(a). This type of spacing can be easily

identified and special treatment can be introduced to it to reduce the amount of processing needed for line segmentation of such pages.

2. Crammed lines, where all lines of the page are mingled and no clear line can separate successive lines, as in Figure 12(b). This would require the maximum amount of processing to separate all the lines with minimum errors.
3. Mixed line spacing, where some lines in the page are easily separable and others are crammed, as in Figure 12(c).

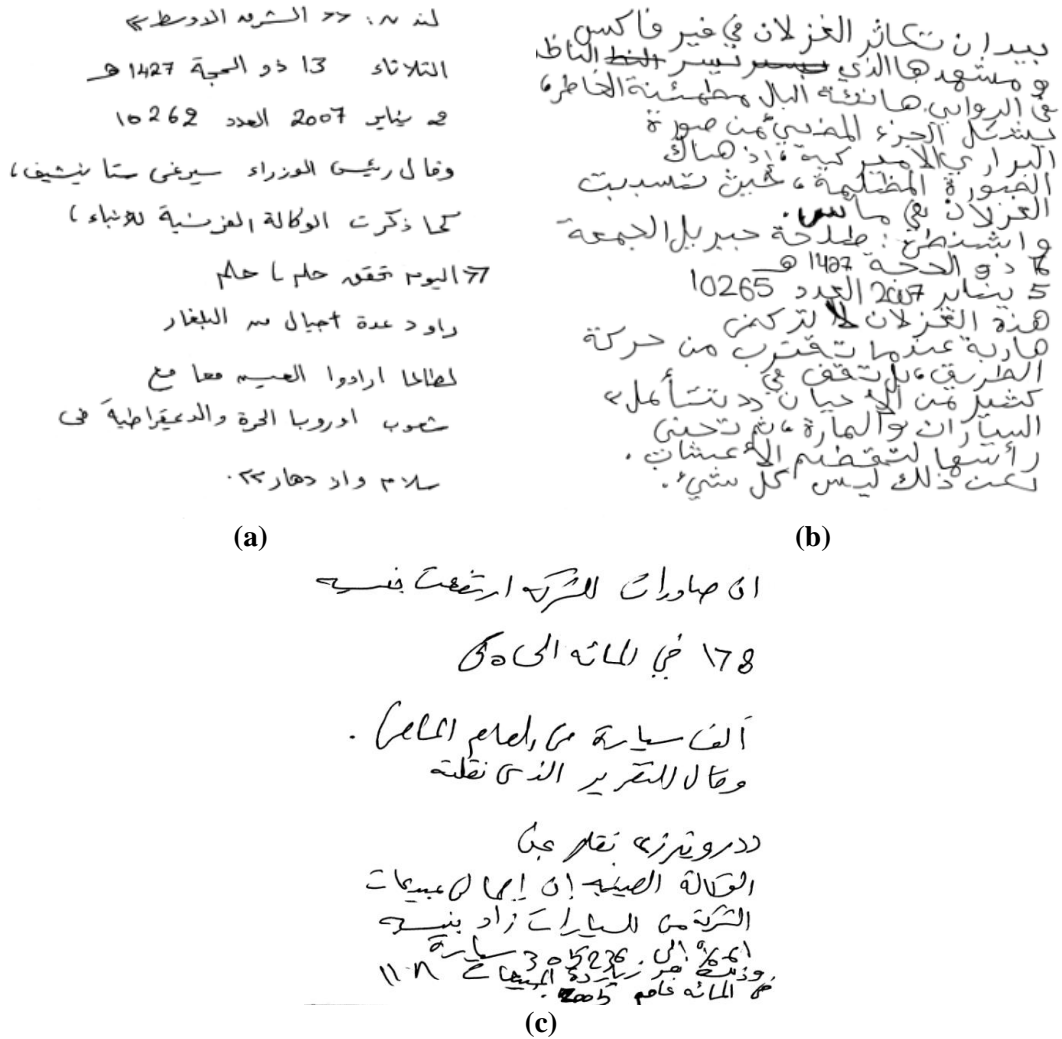


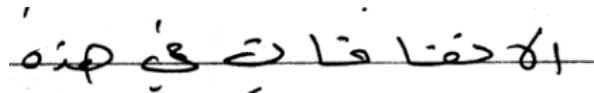
Figure 12. Different types of line spacing in pages.

3.1.7 Intra-word spacing

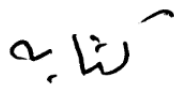
Arabic is cursive in nature, however, some letters in Arabic are never connected from the left, causing many words to break at a single or multiple positions creating the intra-word discontinuity or space. As a standard in writing, inter-word space should be significantly larger than the intra-word space. This is usually the case but that is not always followed in handwriting. In some cases, intra-word space is significant and in some cases it even surpasses its inter-word counterpart as Figure 13(a).

If a letter is composed of multiple strokes, which are written separately, the scribe may generate discontinuities in the single letter creating intra-word space at the letter level as in Figure 13(b). This also needs to be identified and segmented properly.

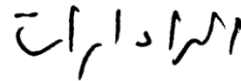
Some words are composed of several parts that the number of separate parts cannot be determined sharply. The word in Figure 13(c) is composed of eight horizontally aligned parts.



(a)



(b)



(c)

Figure 13. Intra-word space examples.

The ability to distinguish complete words without breaking them down improves the segmentation and helps in building a more meaningful document later after recognition.

3.1.8 Slant

Slant, as shown in Figure 14, is the inclination of the downward strokes in handwriting. It originates from the scribes writing style. This has an impact on the

validity of the extracted features –especially the characteristic angles in letters, of the handwritten text and the consequent sequence processing and recognition applied to features matrices. Slant can be preprocessed before feeding word images to the recognition system or can be implemented in the recognition system itself.

It is noticed that slant is not frequent in MADCAT training set. In some documents, slant only exists in parts of the document.

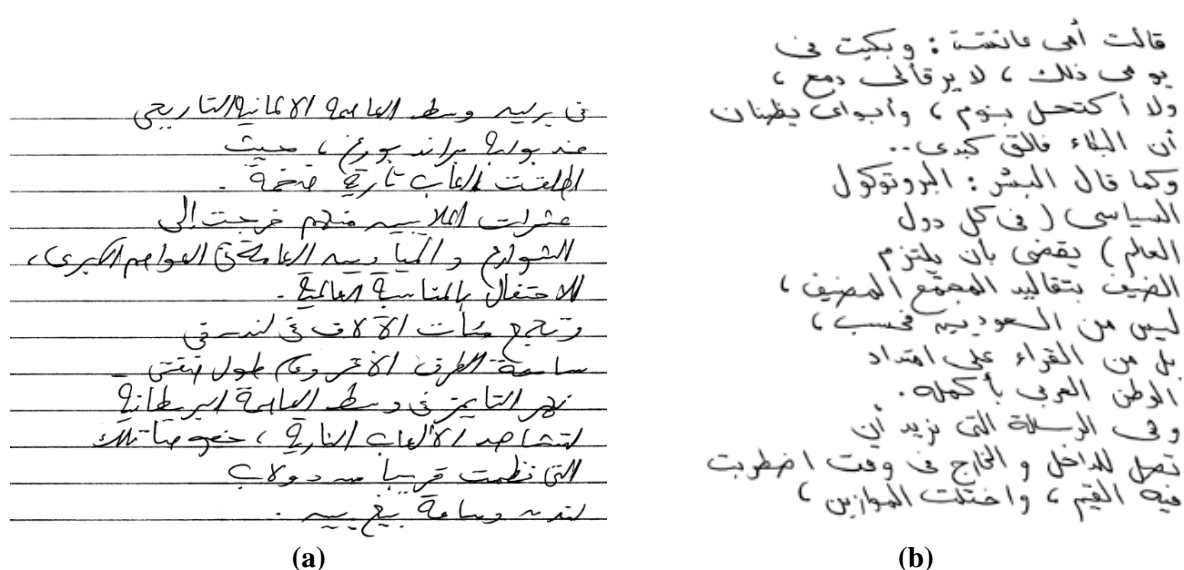


Figure 14. Slant examples.

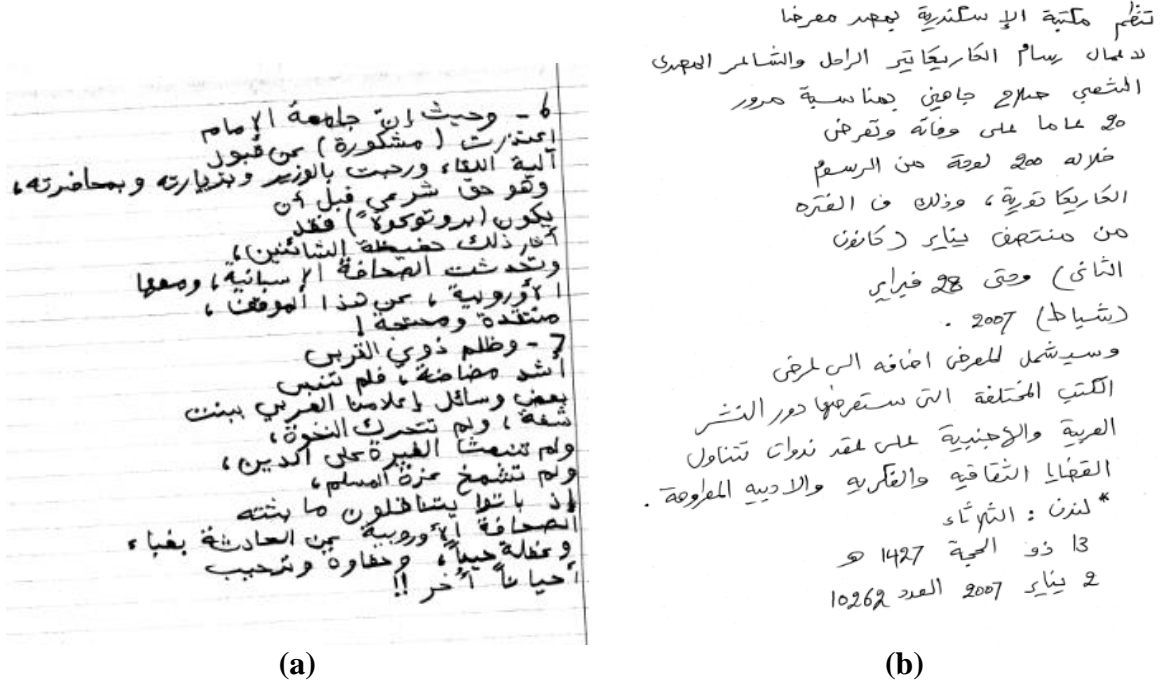
3.1.9 Skew

Skew of hand written text is the angular deviation of the text from the horizontal line as shown in Figure 15(a). Skew has a significant impact especially on the Arabic recognition system because many systems use baseline estimation algorithms that are not robust against skew. Wrong estimation of the baseline will lead to wrong feature detection and hence erroneous detection of letters and words.

After inspecting the MADCAT training sets, it was noticed that skew can be categorized as follows:

1. Skew from image acquisition phase, or image skew where all lines share a relative skew angle. See Figure 15(a).

2. Skew from writer, or writer skew where not all lines have the same skew angle as shown in Figure 15b.



(a)

(b)

Figure 15. Types of skew.

Hence, the total skew is the vector sum of image skew and writer skew components. Image skew is simpler and can be corrected using collective algorithms that analyze the whole text body. Writer skew on the other hand is more complex and requires approaches that process each line or line component in the text body to detect and correct the skew of each line or each part of the line separately.

It was also noticed that if the paper is lined, then ruled lines orientation can determine the image skew and the document is free of writer skew.

3.1.10 Bleed-through text

A single sheet of paper is not a completely opaque object due to its nature and thickness. In some cases, a problem arises when multiple pages are stacked one over another or when transcriptions exist on both sides of the paper sheet. Traces of the transcription on faces other than the one being scanned may appear like a bleed-through

text as in Figure 16. Fortunately, the density of bleed-through text is much less than the original text and thus may be treated as noise.

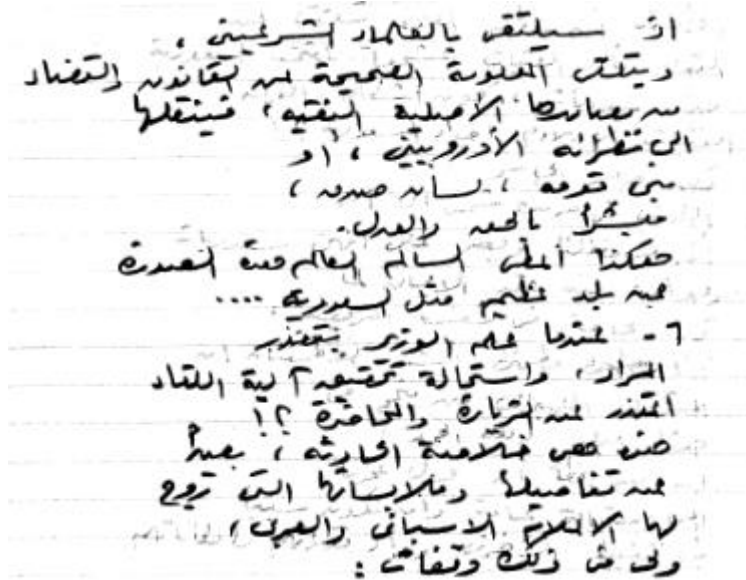


Figure 16. Bleed-through text.

3.1.11 Varying text body

Images provided in MADACAT training set have almost the same resolution of around 5100x6600 at 600 dpi. However text body highly varies in size, font height and size, line density, and reference position. In some cases, text starts immediately from the right or top edges of the image with no margins.

3.1.12 Irrelevant text or numerals

It was noticed that some documents have some side notes or some irrelevant alphanumeric characters distant from the main text body as in Figure 17. These characters have no reference in the corresponding ground truth files. Thus, they should be dropped, especially when written in foreign languages.

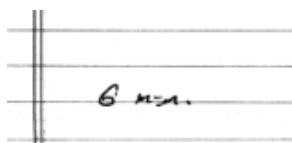


Figure 17. Irrelevant text example.

3.2 Nonexistent Problems and Unrequired Preprocessing Operations

Some problems and preprocessing operations are ignored either because they were not found in the MADCAT training set or because they are treated in the selected recognition system. These problems and preprocessing operations are described in the following subsections.

3.2.1 Binarization

All the analyzed documents of the MADCAT training set are binarized. Hence, there is no need for binarization.

3.2.2 Height normalization

Once text body words are segmented and fed to the JU-OCR2 recognition system, it is capable of recognizing the letters regardless of their heights since the algorithm depends, not on pixel count, but on the features of the thinned lines and stroke angles, making it insensitive to characters heights.

Also, the pixel density for the documents in MADCAT is high. The number of pixels per word is sufficient for processing even for the smallest size of words and the thinnest of pen strokes. Thus, even though height variations are significant across different files, no height normalization is deemed necessary.

3.2.3 Line thickness variations/manipulations

The JU-OCR2 system contains an efficient thinning algorithm that would obsolete any line thickness variations/manipulation.

3.2.4 Multiple text bodies

All of the inspected documents in MADCAT training set contained a single plain text body that did not include any tables, footnotes, or any other subsidiary text bodies.

All documents should be assumed to have a single plain text body while other parts of the document will be discarded to reduce the computational cost of document preprocessing.

3.2.5 Special characters separation

The special characters (punctuation marks, percentage sign, dollar sign, etc.) might create problems if they are to be segmented as separate entities, but since they do not really mean much for extracted/recognized text, they should be left to JU-OCR2 recognition system.

3.2.6 Baseline detection

This is one of the first operations carried out in JU-OCR2. Even though it is a basic operation in JU-OCR2 system, it does not account for inclined baselines. Nevertheless, baseline detection will still be left to JU-OCR2.

3.3 A Suggested Sequence for Document Recognition System

After analyzing MADCAT training set and finding the aforementioned problems, we propose the following sequence (shown in Figure 18) to be followed when creating a comprehensive document recognition system to avoid unnecessary processing and computations. Some operations are proposed to solve multiple problems. For example, noise removal operation is expected to resolve the pepper noise and bleed-through text whereas text body extraction is expected to resolve irrelevant text or numerals, external graphical elements, and text body variation.

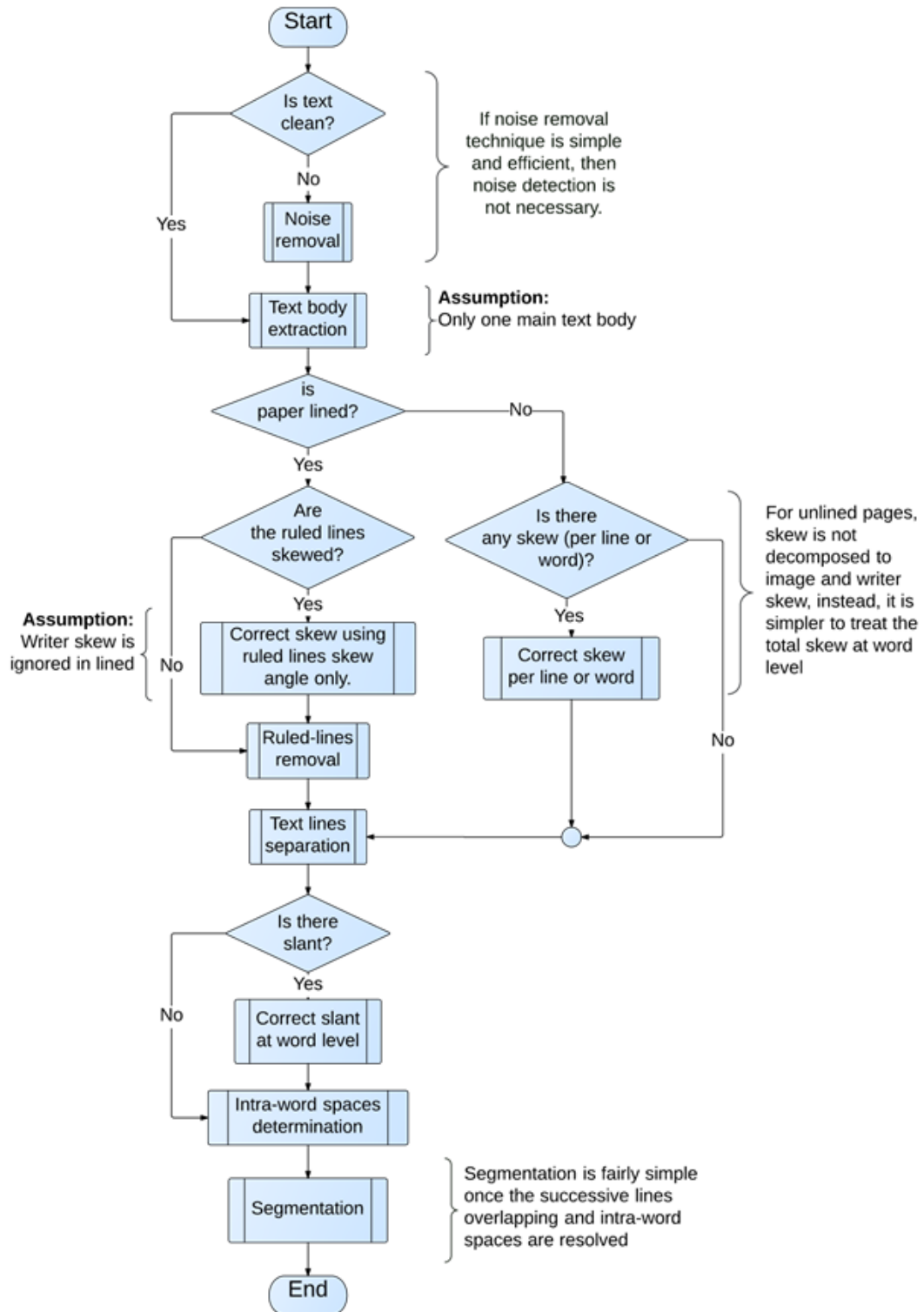


Figure 18. Suggested sequence for document recognition preprocessing system.
 Many operations in Figure 18 are entirely or partially independent of each other.

Thus, further modifications can be introduced to this sequence to support parallel processing in order to better utilize computer's resources and improve the system's performance.

Depending on the complexity of the noise removal algorithm, a simple noise detection algorithm can be implemented to decide whether the noise is to be removed or not. This is done in case the noise removal algorithm was complex or time consuming to improve the performance of the system when handling clean images.

In order to handle the general document case, text body extraction can then be applied to separate the text body from the rest of the image. In some complex applications like form processing, textbook processing, or newspaper processing, layout analysis may also be needed to analyze the document structure including multiple text bodies, tables and graphics. However, it has been noticed that in MADCAT set, there is only one text body in the document image.

A ruled line detection algorithm can be used to determine if a document is lined or not. If the paper is lined, the skew angle of the ruled lines is considered as the skew of each word in the document. The skew is corrected and then the ruled lines are removed. If the document is not lined, then the skew is handled per line or per segment of image to account for writer skew.

The image is then segmented to text lines, slant detection and correction is applied at the line level to remove the slant if the recognition system is sensitive to slant.

The spaces in each text line are then classified into inter-word spaces or intra-word spaces. Word segmentation is directly based on this classification of spaces.

4 REQUIREMENTS OUTLINE

Depending on the algorithms proposed earlier in this thesis, we establish selection criteria for each process in the algorithm depending on the problems the process is expected to resolve. We also take into consideration the selected recognition system, JU-OCR2, its capabilities and its vulnerabilities, through the process of selecting the optimum preprocessing scheme.

The input images of MADCAT are binarized with resolution of 5100x6600 pixels at 600dpi.

4.1 Selected Recognition Algorithm

As indicated earlier, we have chosen a recent recognition system proposed by (Abandah, et al. 2014), called JU-OCR2. It is augmented in this thesis with selected preprocessing algorithms in order to extend its scope from word-level recognition to document recognition.

In order to do so, we need first to inspect the steps performed within the JU-OCR2 in order to clarify its capabilities and limitations.

In a glance, (Abandah, et al., 2014) built a novel OCR system, see Figure 19. The system was an updated version and a continued effort of the earlier version that is described in (Abandah and Jamour, 2010) that used the contributions of different studies like (Abandah and Jamour, 2014) and (Abandah and Malas, 2011). Their system is comprised of five main stages; sub-word segmentation in which the authors segment the words into sub-words. A novel rule-based algorithm then further segments the sub-words into graphemes. The graphemes are then fed to a robust feature extraction algorithm that produces a feature vector for each grapheme. The feature vectors are then sent to a bi-directional long short-term memory (BLSTM) recurrent neural network

(RNN) with connectionist temporal classification (CTC). These features of the RNN provide high recognition accuracy and facilitate its training. Using graphemes as the recognition unit allows for open vocabulary recognition while an optional word matching step improves accuracy for the limited-vocabulary recognition.

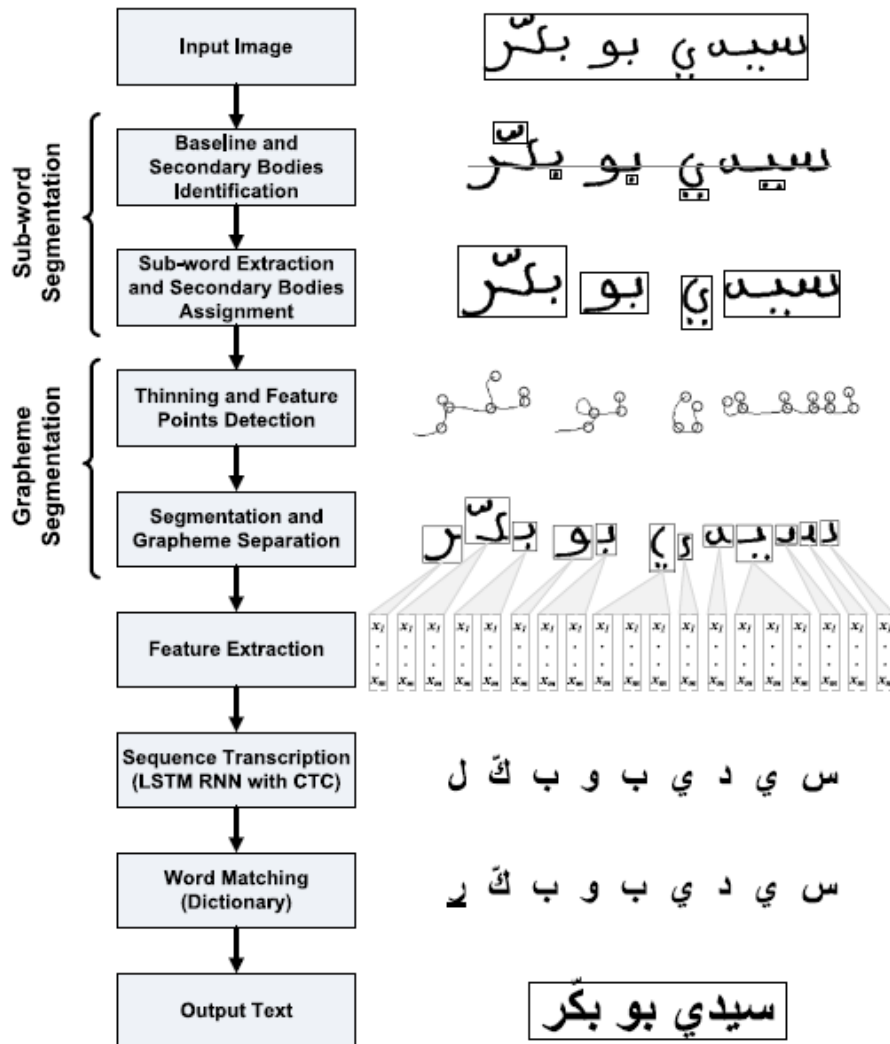


Figure 19. The steps of the selected recognition algorithm, JU-OCR2 (Abandah, et al., 2014).

Figure 19 above shows the basic steps implemented in the JU-OCR2. First, the system can only handle word segmentation, thus the preprocessing system should feed word images to the JU-OCR2 system to be recognized.

During baseline identification, only one baseline is calculated as the algorithm is not designed to handle multiple lines of text.

The baseline identification is based on the maxima of the horizontal projection profile (HPP). From this we conclude that this stage is sensitive to skew and ruled lines since both may shift the HPP maximum value to a location other than the baseline, eventually leading to erroneous baseline detection. It is worthwhile to mention though that the baseline estimation is only used in recognizing secondary bodies and extracting some configuration features.

As a speculation regarding this stage, there seems to be a mechanism that handles very small objects during the feature extraction phase. This may affect the determination of the secondary bodies. This means that the algorithm can handle noise, especially pepper noise where noise is comprised of scattered foreground dots. The noise clusters may form an issue but the probability of forming a significant cluster that, at the pixel density at which MADCAT files are scanned, compared to true secondary bodies of text is rather minute.

During the thinning phase, the text skeleton is extracted; this means that the algorithm is robust against the writer style variations and text thickness variations.

During the phase of feature points detection, the features of the sub-words are detected; this is the part of the algorithm where ruled lines would mostly affect. Wrong features will be extracted crippling the rest of the algorithm and causing major recognition failures. Ruled line removal would be crucially needed in preprocessing.

During classification of the graphemes and their features, considering the features that are taken, Arabic language properties, and relating this to the nature of the slanted text, the algorithm is supposedly trainable to handle slanted text.

In summary, in terms of preprocessing, JU-OCR2 has the capacity to handle the following issues:

1. Noise, especially pepper noise. However, to reduce the load on the algorithm, it is better to reduce the noise to a minimal level.
2. Text stroke thickness variations due to thinning.
3. Slant, being a trainable system, JU-OCR2 can be trained to handle slanted text.

The selected system, JU-OCR2, have the following limitations that should be handled in preprocessing:

1. Ruled lines. All ruled lines should be removed for JU-OCR2 to perform correct baseline detection and correct feature extraction.
2. Line segmentation, the system is not designed to handle multiple lines of text, especially the large number of lines found in a completely filled document image.
3. Word segmentation, the system is only designed to read single words.
4. Text skew. Skewed would cause erroneous detection of the “less-important” baseline. Severely skewed text may cause wrong features to be extracted. We believe the system can handle low to moderate skew level.

4.2 Scope of the Thesis

There are many problems available in handwritten documents. Each problem has a different level of impact on the performance of JU-OCR2 character recognition system (e.g. JU-OCR2 can tolerate text slant but cannot handle multiple lines of text). The complexity of many of these problems are much greater than the impact of the problem

(e.g. the struck-through text is hard to be detected and processed in cursive text and is rare and has minor impact on the overall accuracy of the system). In addition, the solution designed for one problem may eliminate or mitigate another problem due to similar properties of the two problems (e.g. removing pepper noise may remove or mitigate the bleed-through text). We avoid amplifying one problem by a solution of another problem by careful selection of algorithms and proper ordering of algorithms within our preprocessing system.

Considering the rationale above, it is only rational that dealing with all of those problems is unnecessary and should be beyond the scope of this thesis. Thus, we concentrate the efforts in this thesis on the most common and frequent problems found in handwritten documents analyzed thus far and the problem that are believed to have a significant impact on JU-OCR2 when it is extended to handle handwritten document images.

The problems included in the scope of this thesis are:

1. Segmentation

The selected recognition system at the core of the proposed system, JU-OCR2, is only capable of recognizing separate words. All the documents in MADCAT training set are comprised of multiple lines making segmentation at line, word, or sub-word levels crucial for the recognition system to be capable of processing a whole document.

In order to ensure proper segmentation, especially if it is performed at the word level, intra-word spaces should be identified to avoid creating segmentation cuts at those spaces so we obtain the best results for the final document since, most probably, a space character will be automatically inserted at each segmentation-position equivalent after recognition takes place.

2. Ruled lines

About a third of the training set documents are of lined paper (NIST, 2013b). Given the impact and distortion introduced by the ruled lines on handwritten text, it will greatly affect the horizontal projection profile (HPP) and thus the detection of the words baseline. Also, it will impact the features extracted from text leading to erroneous recognition. It is only natural that any document recognition system should be capable of detecting and removing ruled lines, horizontal or vertical.

3. Skew

It was noticed during the analysis of MADCAT training set documents that writer skew is common in unlined documents, and a fair share of documents suffered image skew. Since JU-OCR2 assumes text to be horizontal. This thesis will discuss the detection and correction of text lines skew.

4. Additional problems

One or more problems like noise removal and bleed-through text are tackled over the course of this thesis in order to come out with a sequence that covers the vast majority of problems that prominently affect HCR performance with minimal effort. However, due to the simplicity, clarity, and straightforwardness of those problems, they will not be discussed and have their related literature reviewed the way we will handle the abovementioned problems. Instead, only a simple process will be proposed to handle these problems.

4.3 Noise Removal

As discussed in Section 3.1.1, pepper noise is mainly separate scattered individual noise pixels as observed in MADCAT files. They mainly arise from image capturing

and binarization phases. However, when pepper noise is dense, some larger noise pixels clusters may be formed in some severe cases as shown in Figure 3(c). This can be more clearly seen in Figure 21(a), which is an enlarged version of Figure 3(c).

Noise may have considerable distortion on the image since it can affect both HPP and vertical projection profile (VPP). It can affect the connected component analysis, principal component analysis, text body extraction, and a number of other approaches used in document image processing.

Examination of faint vertical lines and bleed-through text in MADCAT binary images shows that the lines resemble pepper noise. They are dissolved and do not form eight-way connected components or clusters. This essentially renders those problems equivalent in effect to the pepper noise. In summary, all three problems; noise, bleed-through text and vertical lines, have the same characteristics; they all generate eight-way disjoint pixels with occasional pixel clusters.

Noise removal is a well-established branch in image processing with multiple techniques available (Motwani, et al., 2004) and (Farahmand, et al., 2013). Many of those techniques are readily available in image processing libraries and packages such as Open Computer Vision Library (OpenCV). However, taking the following properties into consideration simplifies the noise problem to the largest extent:

1. Noise is mostly in the form of eight-way disjoint pixels.
2. Images have high resolution, 5100x6600 pixels, at 600 dpi thus even the smallest text component is comprised of a considerable number of pixels.
3. Noise objects sizes are mostly not comparable to text objects sizes.

The simplicity of the available noise, mostly pepper noise with occasional clusters, makes the search of a proper simple algorithm in literature not worthwhile. Instead, we devised the simple two-iteration morphological opening procedure to handle noise. As illustrated in Figure 20, the first iteration with a rectangular 3x3 kernel of pixels and the second with 5x5 rectangular kernel to remove the remaining larger pixel clusters.

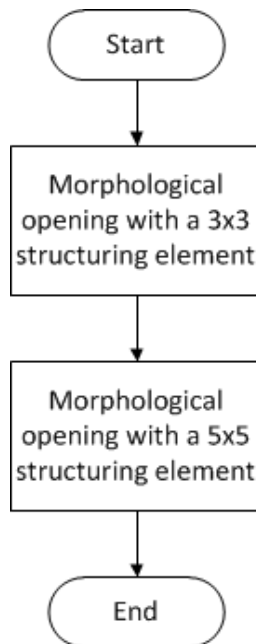


Figure 20. Devised noise removal algorithm.

As per what was discussed earlier, the algorithm we proposed expects the noise removal algorithm to be able to eliminate or mitigate the following problems:

1. Pepper noise.
2. Faint vertical lines generated during image acquisition.
3. Bleed-through text.

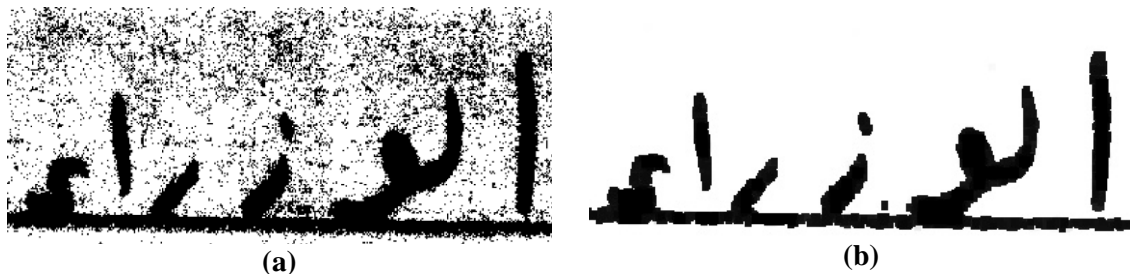
4.3.1 Performance of the algorithm

Having a simple technique for noise removal, we are only interested in verifying that it would be sufficient to remove noise without loss of features of the text objects.

Thus, we define the performance of this algorithm as the ability to remove noise pixels and retain text pixels at low computational cost.

Figure 21 shows the results of applying this algorithm on different cases, it can be noticed that all text features were maintained and only noise was removed from the image. It can also be noted that some residual noise remains due to the size that few noise clusters can grow to. We deem this residual noise inevitable yet acceptable overall due to their observed rareness against the additional computational effort and complexity that would have been needed in case we need to eliminate them.

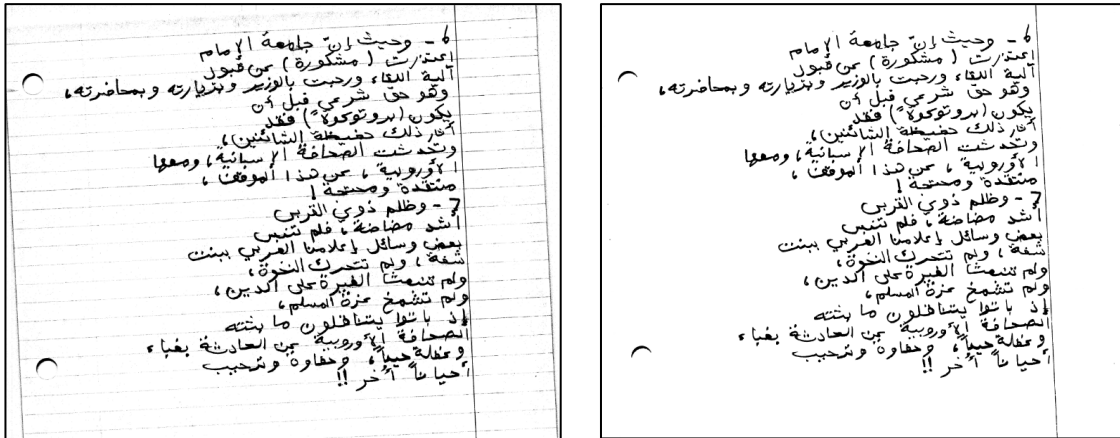
Having devised a simple noise removal technique, we will not need to answer the conditional at the beginning of the algorithm regarding text cleanliness, see Figure 18. Instead, it shall be removed; the algorithm shall begin with unconditional noise removal.



(a) (b)

از سلتق بالعلماء ابراهيم ،
 ديتلت الملوحة العميقة له تقانده التقاد
 سه صارت الاصلية اتيه ، فينتقل
 الى نظراته الأروبية ، او
 بنى قومه ، لانه صده ،
 منبكر بالحد والحد .
 صكنا الله اسام العالم حبة اصدت
 مع بلد تنظيم مثل لبريه ...
 6 - لغتنا مع الوزير يتقدر
 المراد ، واستقالة تمسحه آية اللناد
 المنذ مع ابراهيم والحاشية ؟
 صنه صه فلامنة اشارته ، بعد
 مع تفاضلا وملاياترا التي تروج
 لها الامم الايباني والعري ،
 وفي من ذلك وتفايته :

(c)



(d)
 Figure 21. (a) Enlarged illustration of Figure 3(c) that shows the nature of the severe case of pepper noise. (b) The same portion of the image after applying the simple noise removal technique (c) Bleed-through text sample of Figure 16 after applying the same noise removal technique. (d) Effect of the noise removal algorithm on types of binarized faint lines, including faint ruled lines.

5 RULED LINE REMOVAL

Ruled lines are common in handwriting since they are designed to help people write straight on paper. Lined paper come in various forms and shapes. Ruled lines tend to connect text elements, which, if left untreated, would make the use of any connected-components-based algorithm unreliable. This is due to the assumption that separate words produce separate connected components. Moreover, their presence leads to erroneous baseline detection and distorts the shape of any character that touches them. In the case of vertical ruled lines presence, different lines will be connected, thus the majority of foreground pixels will form a single connected component (Kumar and Doermann, 2011).

In Chapter 3, we deduced the general properties of ruled lines in MADCAT images set. We survey and select here the optimum algorithm available in the current literature for our selected recognition system, the JU-OCR2.

5.1 Problems to be Resolved

Ruled lines in MADCAT files have a specific set of characteristics. However, lined paper images are randomly distributed among MADCAT files, we do not have any a priori information whether the page is lined or not. We need an algorithm that can handle the following:

1. In case ruled lines are present, the algorithm should be able to handle them without distorting the text characters.
2. The algorithm should be able to handle different grades of line degradation as illustrated in Figure 8.
3. The algorithm should not have any effect on unlined pages; this can mainly be achieved in case a proper and non-destructive detection algorithm is utilized.

4. Can handle both horizontal and vertical ruled lines. Nevertheless, this is not mandatory because some simple techniques exist which expand the capability of the algorithms to process horizontal ruled lines to vertical ruled lines. For example, the image can be rotated 90 degrees and reprocessed by the same algorithm.

5.2 Approaches to Ruled Line Removal

Current literature concerning document images ruled line removal offers multiple approaches to the matter. Ruled line removal approaches can be generally classified into two categories; model-based and heuristic-based (Kumar and Doermann, 2011) we survey the available approaches in the coming part based on the general approach to detection/removal of ruled lines.

5.2.1.1 Run-length analysis

Blumenstein, et al. (2002) proposed a simple straight line removal technique that basically depends on the run-length of straight line strokes in word images. Removal is based on the average thickness of the line strokes. The algorithm is sensitive to noise that may lead to failures in line removal.

Arvind, et al. (2007) proposed a line removal system in which documents go through four steps. First, the document image is cleaned of noise then the image is segmented into blocks. Entropy and horizontal projection profiles are used to de-skew each block. Lines are then detected using the projection profiles and run length analysis. After lines are removed, strokes are detected and corrected for gaps created by line removal. The minimum stroke-line angle that can be detected and corrected is 10 degrees.

5.2.1.2 Hidden Markov Model

Zheng, et al. (2005) proposed an algorithm for detection of parallel lines without removal. The algorithm first roughly filters text components using Directional Singly Connected Chain (DSCC) to minimize text interference with lines, then the image is deskewed And the horizontal projection profile is calculated. The authors modeled the resulting horizontal profile with a trainable hidden Markov model that was decoded using Viterbi algorithm to find the optimum positions of the ruled line, simultaneously. The algorithm is robust against line degradation.

5.2.1.3 Linear Subspaces

Abd-ALMaged, et al. (2009) used a set of empty lined images to incrementally construct the vertical and horizontal lines subspaces representation. Moment and histogram-based features are later extracted from the test images. Feature vector is then projected on the subspace model. Pixel classification as ruled line pixel or text pixel is based on the reconstruction error; if the error is larger than an experimentally-determined threshold, the pixel is a foreground pixel. Otherwise, it is part of a ruled line.

5.2.1.4 Projection profiles analysis

In the method proposed by (Cao, et al., 2009), pepper noise is first removed by morphological opening operation followed by filtering connected components smaller than $\left(\frac{1}{75} dpi\right)^2$ where dpi is the image pixel density in dots per inch. For ruled line detection, the image is split into 10 vertical strips and the local maxima of the HPP of each strip are found. The peaks which were detected in at least six stripes with similar heights are elected as valid ruled lines. The ruled lines height is estimated empirically. The height of the ruled line is increased a little to accommodate the noise contiguous to

the ruled line. The skew angle is estimated using the information obtained through the line detection phase by calculating the slope angles between each line's pixels detected in the HPP's of the strips and finding the most frequent slope angle, which is equivalent to using the mode of the slope angles. Three simple heuristic approaches are then used to tackle the common text strokes disfigurements accompanying most line removal processes, namely, broken strokes, thinned strokes, and broken baselines.

Shi, et al. (2010) based their algorithm on local profiling of the document image. They applied the n adaptive local connectivity transform using the fuzzy run-length as proposed by (Shi, et al., 2005) where the image is transformed based on the value of the neighboring pixels of a foreground pixel in a given direction; horizontal or vertical. To detect the ruled lines, the transformed image is binarized using a modified local thresholding algorithm. Linear regression is then used to reconstruct the ruled lines and group broken segments of each ruled line. Assuming that text/ruled line overlap areas are generally thicker than ruled lines themselves, lines are then removed using vertical run-length analysis of the lines' pixels at the original image. The algorithm does not affect images with no ruled lines.

Kavallieratou, et al. (2011) method uses the common properties ruled lines have like uniform width, consistent spacing, and position relative to text. Without prior preprocessing, this algorithm first computes the upper profile of the image; i.e. the first black pixel on each column, starting from the top. If a considerable amount of pixels in the top profile of the image vary vertically by 0 or 1 pixel from the adjacent cell, then ruled lines are detected, regardless of their skew. The thickness of the ruled line is estimated. Then the top left point of the first detected segment is chosen as the first point. The top right point of that segment is the second point. The third point at the other end of the document is calculated by posing the determinant of the three collinear

points equal to zero. The detected ruled line is then deleted. The image is then whitened for an area of five lines thickness and the whole process is repeated to detect and eliminate the next ruled line. For vertical lines, the image is rotated 90 degrees and the whole algorithm is repeated with the exception that vertical lines spacing can be narrower. Noise can affect the performance of this approach.

Kumar and Doermann (2011) presented a fast way to compute Horizontal Projection Profile and Vertical Projection Profile features using integral images introduced by (Viola and Jones, 2004). The projection profiles are then fed to one of the most popular methods for supervised classification, Support Vector machine (SVM) for the pixels to be classified into text pixels or ruled line pixels. To reduce the number of pixels used in training of the SVM, they applied a data selection method based on incremental subspace learning.

5.2.1.5 Hough transform and multi-line regression

Lopresti and Kavallieratou (2010) proposed a simple robust method utilizing the common properties of ruled lines such as predictable spacing, uniform thickness, brokenness in binarized images is due to their original light color, and that ruled lines commonly overlap with text. The algorithm starts with examination of the left and right columns of the image with black pixels to estimate the thickness of the ruled lines. Pairs of endpoints for each line are then assigned from the left and right edges. The straight line equation $y = ax + b$ is then used to in conjunction with the line thickness to find and eliminate lines pixels. Separation between ruled line and text pixels is based on the assumption that text stroke's thickness is greater than that of a ruled line's. Post processing helps dealing with lines with deteriorated edges. The proposed process has no effect on unlined pages and thus requires no prior discrimination.

Chen and Lopresti (2014) proposed a model-based algorithm that uses Hough transform to detect ruled lines. The ruled lines model properties are then utilized to detect the ruled lines that were missed. The authors employ multi-line linear regression to estimate model parameters.

5.3 Ruled Line Removal Algorithms Evaluation and Selection

Table 1 shows a summary of the evaluation results of each of the approaches described earlier. In concept, all the approaches have shown varying degrees of positive and promising results. It has to be noted though, looking at the table below, that there is no common standard that the authors used for performance evaluation of their respective approaches. Some authors tested their methods at pixel level using accuracy, others using F-score while some linked their line removal approach to an existing character recognition system and investigated the impact on the word error rate (WER). One other difference in their performance evaluation approaches is the significant differences in the test samples used.

Table 1. Ruled Line Detection and Removal Methods Summary

Reference	Technique/ method	Performance Evaluation		Notes
		Result	Test Sample	
Zheng, et al., 2005	Hidden Markov Model	96.8%	Tested on 1596 lines in 68 Arabic handwritten images (2274 lines were used for training)	- Ruled lines detection only - No removal and strokes regeneration offered
Abd-Almageed, et al., 2009	Linear Subspaces	88% Accuracy	50 Arabic handwriting documents	- Ruled lines detection only - No removal and strokes regeneration offered
Kumar and Doermann, 2011	Projection Profile with trained SVM	91.4-94%	Around 18 images of constructed images and real document images	- Ruled lines detection only - No removal and strokes regeneration

Reference	Technique/ method	Performance Evaluation		Notes
		Result	Test Sample	
Chen and Lopresti, 2014	Multi line regression model with the global solution of the least square error	95% detection accuracy	100 documents	- Ruled lines detection only - No removal and strokes regeneration
Blumenstein, et al., 2002	Run-length analysis	97.16%	Tested on the 317 word images from CEDAR benchmark database of handwritten cursive words	No regeneration of text strokes
Kavallieratou, et al., 2011	Top profile analysis and the fact that the determinant of three collinear points is zero	Precision: 0.81% Recall: 0.92%	100 pages of different languages	No regeneration of text strokes
Shi, et al., 2010	Directional Local Profiling for detection For removal, adaptive vertical run-length search	98%	274 MADCAT images, 155 of which include ruled-lines	Assumes that text strokes are thicker than ruled lines
Lopresti and Kavallieratou, 2010	Multi-line Interpolation	Precision: 96% Recall: 90% F1: 93%	20 synthetic images of Arabic text	Assumes that text strokes are thicker than ruled lines
Cao, et al., 2009	Projection Profile of image segments	WER was reduced from 47.8% to 35.2% Pixel-level mismatch: 2.07%	The algorithm was used in conjunction with a recognition process proposed by (Natarajan, et al., 2007) Pixel-level mismatch measured on a single synthetic image	- De-skews lined pages - Robust against broken ruled lines - Has no effect on unlined pages - Handles vertical and horizontal lines - De-noises images - Works with high resolution images
Arvind, et al., 2007	Run-length analysis	86.33% Accuracy	300 English documents	Includes: - De-noising - Line removal - Regeneration of strokes that intersect the ruled lines at angles greater than 10 degrees

Bearing in mind that all approaches have competitive results in line removal, we will base our selection on the features and functionalities offered by each of these approaches and how they can benefit our selected recognition system, JU-OCR2. The algorithm that would meet all the requirements and offer the most features shall be selected for line removal.

Some of the algorithms only offer detection of the ruling lines without any accompanying removal technique (Zheng, et al., 2005), (Abd-ALMaged, et al., 2009), (Kumar and Doermann, 2011), and (Chen and Lopresti, 2014). Those algorithms are excluded and are not suitable for our preprocessing system.

Other algorithms perform line removal but ignore the regeneration of text strokes that originally overlapped the lines, resulting final text distortion (Blumenstein, et al., 2002) and (Kavallieratou, et al., 2011). This is also a critical requirement that is not met in these algorithms and thus are not suitable for our system.

Some algorithms have based their line removal techniques on the assumption that text strokes are thicker than ruled lines (Shi, et al., 2010) and (Lopresti and Kavallieratou, 2010). During the study of MADCAT set, it was found that this is not always the case and that text could sometimes be written with some thin strokes depending on the writing tool and the writer style. Thus, this assumption is considered impractical for our case of study since our preprocessing system should be robust against writer style variations. These approaches are not suitable for our system also.

Among the two remaining methods, the process proposed by (Cao, et al., 2009) is preferred over the one proposed by (Arvind, et al., 2007) due to the completeness of the features package in the former. While the latter only features de-noising and

regeneration of strokes that intersect the ruled lines at angles greater than 10 degrees,

Cao, et al. (2009) line detection/removal process includes:

1. De-skewing for lined pages
2. Robustness against broken ruled lines
3. No effect on unlined pages
4. Handling of vertical and horizontal lines

This process was also designed to handle high-resolution images like MADCAT set images. The process achieves all the requirements set earlier and is thus selected as the line removal technique in our system.

6 SEGMENTATION

Having the image de-noised, ruled lines cleaned, all document images at this stage are equivalently comprised of text lines that need to be segmented in order to be fed to the JU-OCR2 for processing.

6.1 Problems to be Resolved

The selected segmentation system is expected to be able to resolve the following issues;

1. Perform line and word segmentation. JU-OCR2 only recognizes single words.
2. Should handle consecutive lines overlapping and handle touching lines and split components stretching between two consecutive lines.
3. Properly handle crammed lines where lines are close to each other as illustrated in Figure 12(c).
4. Assign diacritics and secondary components to their corresponding lines/words.

6.2 Approaches to Segmentation

The following is a brief description of the segmentation approaches available in literature.

Amin and Al-Sadoun (1992) described a method for segmenting letters of printed Arabic text. At the preprocessing phase, image is thinned using binary tree tracing. A 3x3 window sliding from right to left is used to create the binary tree and generate modified Freeman code that describes the thinned image. The binary tree is then minimized and segmented. Tree traversal is later used to identify segmentation points. Tree is segmented such that each letter is contained in one or more sub-trees.

Motawa, et al. (1997) method started with binarization followed by slant detection using the morphological erosion operation with a structuring element that depends on the average thickness of the word. The average slope of all strokes is then calculated to find the overall slope of the text. Slant correction is done by transforming pixels based on the slope of the processed image. After preprocessing, connected components are used for segmentation based on the assumption that Arabic characters are connected by horizontal lines, namely, baselines. Thus, the following is done; text is smoothed out and noise is reduced by filtering. Singularities are then found by opening while regularities are found by subtracting the singularities from the document. The segmentation points are determined by scanning the regularities.

Manmatha and Srimal (1999) proposed the use of scale space techniques in word segmentation for the first time. At the preprocessing stage, vertical and horizontal lines caused by photocopying are removed. To perform the line segmentation, the horizontal profile for the greyscale image is calculated then smoothed by applying a Gaussian filter to reduce the sensitivity. Local maxima, which indicate spaces between lines, are then found by determining the location at which the derivative of the smoothed profile equals zero. Blob analysis stage was next, at which, instead of using Laplacian of a Gaussian (LOG) to create blobs in each line's image, they used an anisotropic differential operator to form the blob. The use of anisotropic filtering rather than isotropic filters is due to the fact that the scale of most words is larger in the x-axis than in y-axis. After blob scaling by using appropriate values for the anisotropic filter, each blob would comprise a whole word. The word images are then extracted using the bounding boxes for each blob.

Tripathy and Pal (2006) line segmentation method is done following these steps; first, the text image is divided into vertical stripes of width that is almost equal to the

word length. The word length is estimated from the document image itself. Then piece-wise separating lines (PSL) are computed from each of these stripes by finding the zero values of the histograms. The potential PSLs are computed from the PSLs obtained in the second step by finding the normal distances between PSL's in each stripe, the distances are found for all stripes, the upper PSL of each pair of PSLs spaced by less than the statistical mode of the measured distances are then eliminated. The rightmost top potential PSL is chosen and extended from right to left to the previous stripe depending on the normal distance with other PSLs in the previous stripe. Also in this step, lines connecting consecutive PSLs are detected and separated. This right-to-left PSL extension and joining procedure continues until the left boundary of the left-most stripe is reached. The length of the line drawn is checked whether it equals the width of the document. If yes, the algorithm proceeds to the next step. Otherwise, PSL line extension is done to the right until the right boundary of the document is reached. The next step is repeating the steps concerning PSL selection and extension for the potential PSLs not considered for joining thus far. Once there are no more PSLs for joining, the line segmentation is complete. Word segmentation utilizes the vertical projection scheme for each line. To overcome line strokes that could reduce the inter-word distance, the authors utilize round-like general shape of Oriya language characters and use the distance between uppermost and lowermost black pixels in each column of the line to sort out the single strokes that should be considered as spaces.

Srihari, et al. (2006) segmentation process, which is also used in (Ball, et al., 2006), starts by finding all connected components (CC) and enveloping them with convex hulls. Minor CCs are merged with major ones into clusters. Internal minor CCs are ignored as gaps are indifferent to them. The width and height are checked for each segment to look for a separate 'Alef' character which implies the beginning of a new

word. Nine features are extracted for each pair of clusters. The feature vector is then fed to a neural network to classify the gap as intra-word or inter-word gap.

Kumar, et al. (2007) segmentation method breaks the image into three parts: text part, picture or graphical part, and background part. It employs global matched wavelet filters tuned using 3000 images of pure text and pure non-text. The results are then classified using Fisher classifier to determine if each pixel is a text or non-text pixel. The results are then post-processed using Markov random fields (MRF) formulation-based pixel labeling to form contextual information that can be used to correct the misclassification, thus enhancing the segmentation results.

Bulacu, et al. (2007) designed an algorithm to work for the archive of the cabinet of the Dutch Queen, Kabinet der Koningin (KdK), whose documents have a fixed format and were written and kept carefully. The pages and lines have negligible skew. The algorithm first combines the HPP, ink-paper transitions, and ink run-length to correctly detect the text lines and eliminate any over-detection of lines due to the occasional horizontal dashes. The initial line segmentation locations are then used as starting points for the droplets path. The text is dilated in order to restore stroke continuity in case the ink was faint and to keep the droplet path away from text. The droplet then starts the path and uses Moore's algorithm to follow the contours of the ascenders and descenders with two radial sweeps; clockwise and counter clockwise to search for contour direction. The search fails if the contour reaches the level with maximum HPP value in the line. In this case, the line is split where the droplet path initially intersected with it.

Zheng, et al. (2008) questioned the robustness of connected component-based algorithms of segmenting text lines against touching lines. They believe that it should be treated probabilistically. The initial assumptions of the algorithm are: binary image

used, orientation of text lines is locally uniform, and skew is less than $\pm 10^\circ$. Since the probability of a pixel to be black is higher at the center of a text line than on the gaps between lines, they estimated the probability map of the image which is equivalent in effect to blurring the image with an anisotropic Gaussian kernel. To minimize computational cost, the support region of the Gaussian kernel is often truncated in discrete implementation with a window size of three standard deviations for x and for y to minimize truncation error. To obtain a unique segmentation of each line after blurring, the authors used a level set technique instead of the error-prone thresholding technique. For blurred images, the estimated initial boundary should evolve into its final state in guidance of a partial differential equation. The algorithm utilizes a priori knowledge that the text lines are usually spread in the horizontal direction. Evolution iterations stop when the difference between iteration is small or after the maximum number of iterations is reached. To overcome the fragments issue resulting from massive within-line spacing, rule based post processing approach is implemented to link the isolated fragments.

Du, et al. (2008) procedure is comprised of three main steps: blurring the text image to enhance text lines, segmentation of text lines by the Mumford-Shah model (Mumford and Shah, 1989), and text line detection by morphing approach. The Mumford-Shah approach is region-based; the solution tries to find the piecewise smooth approximation of each region, text line and background, present in a given image. Segmentation, or as viewed in terms of Mumford-Shah, finding the boundary of each region, depends on minimizing the Mumford-Shah energy functional instead of evolution steps. The procedure uses the piecewise linear approximation of the Mumford-Shah model proposed by (Chan and Vese, 2001) which requires solving only one partial differential equation (PDE) to minimize the energy functional. Based on the assumption that text

lines are horizontal, a rectangular window is used to blur the text with Gaussian filter to enhance text lines. Morphing is used to remove overlaps and connect broken lines by first shrinking/eroding the text line region along the horizontal direction to eliminate the connections between successive lines. Then they prolonged/dilated lines in horizontal direction more than they already shrunk so they connect horizontally spaced line fragments and finally shrinking lines back to their original size. A comparison is then made to sort out noise lines by height. The authors selected the same parameters for all experiments.

In the method proposed by (AlKhateeb, et al., 2009), baseline is detected by finding the peak of horizontal projection at the lower half of the line. CCs are bound by rectangles using an iterative method that scans the eight neighbors of each pixel for connectivity check in the binary image and consequently expanding the bounding box if any two neighboring pixels were found to be black. CC's overlapping in the x-axis are then merged to form combined larger components. Distance between pairs of successive sub words is then calculated. Word separation is done using vertical histograms; word and sub-word distances are identified by the amount of space (distance) between groups of peaks along the x-axis of the histogram. Gaps lengths are determined by the length of zero sequence in the histogram matrix. Threshold between intra-word distance and inter-word distance was manually calculated using the Bayesian criterion of minimum classification error applied on 100 images with 250 words. Segmentation of words is simply achieved by comparing the distance value with the threshold value.

Kumar, et al. (2010) proposed a novel method to segment a document image into text lines. Their method starts with finding the connected components and excluding the diacritic components based on CCs characteristics. The local orientation is then calculated for each component using the piece-wise linear approximation. Breadth-first

search and affinity propagation are then used to form and extract the coarse text lines. Projection profile analysis is then used to detect and correct errors of overlapping lines. The last step is to assign diacritics and accents to their corresponding text lines using a rule-based algorithm that relies on the distance between diacritics and the text line or the orientation line.

Boiangiu, et al. (2014) approach is based on the hypothesis that information energy of a pixel indicates the contribution (i.e. information content) of this pixel to the content of the document. Text lines areas have high information energy while spaces between lines have low energy content. This algorithm uses a window-based method to calculate the information energy map of an input document. Line segmentation is performed then by finding the minimal energy values within the energy map.

Surinta (2014) proposed a method in which the document image is first binarized using Sauvola's algorithm (Sauvola and Pietikäinen, 2000). The document image is divided into chunks. A smoothed HPP is calculated for each chunk. The valleys in each chunk are considered the starting point for the text lines. A straight line is generated at the starting state points between the beginning and the end of each line. The image is then rotated 90 degrees. Starting from the starting states, an artificial water droplet is moved from the top to the bottom of the page around the straight line and the contiguous ink. Droplet path is governed by the lowest cost path to stay as far as possible from ink and in the same time tries to take the shortest path. The final path of the droplet is considered the separator between the two consecutive text lines.

6.3 Segmentation Algorithms Evaluation and Selection

Table 2 shows a summary of the evaluation results of each of the segmentation approaches described in the previous section. As noted before, there is no common

standard that the authors took for evaluating performance of their respective approaches. Some authors did not provide quantitative evaluation and some others tested their methods at pixel level using accuracy. Also, there are significant differences in the test samples size and nature.

Table 2. Segmentation Methods Summary

Reference	Technique/method	Performance Evaluation		Notes
		Result	Test Sample	
Amin and Al-Sadoun, 1992	Segmentation to candidate letters using binary tree and Freeman code	N/A	N/A	No performance results available
Du, et al., 2008	Text line segmentation Script independent using the Mumford-Shah model and morphing to remove lines overlap and connect broken ones	N/A	N/A	No performance results available
Boiangiu, et al., 2014	Segmenting text lines using information energy of pixels	No numeric results provided	Around 500 document images; handwritten and printed	No performance results available
Motawa, et al., 1997	Segmentation to candidate letters using mathematical morphology tools	Up to 81.88% of good segmentation (character level)	Few hundred words written by different scribes ranging from poor to acceptable quality	- Does not include line segmentation - Performs letter segmentation which is already done in JU-OCR2
Dinges, et al., 2011	Segmentation to candidate letters Identification of single letters	Recognition: Up to 88.01%	Author's database: 323 words each written by 12 subjects For testing, they used 200 words from a different writer than the one used for training	- Segmentation to candidate letters - No word segmentation - No line segmentation
Kumar, et al., 2007	Text body segmentation using a global matched wavelet filters	84.8% segmentation precision rate	27 images	The algorithm is for text body extraction out of images with graphical content. It does not handle line or word segmentation

Reference	Technique/method	Performance Evaluation		Notes
		Result	Test Sample	
Tripathy and Pal, 2006	Segmenting documents into lines and words Water reservoir concept-based scheme for segmentation	On 984 lines: 100% accuracy On 290 lines: 97 – 99.9% accuracy On 353 lines: <97% accuracy	1627 lines	Handles skewed and crammed lines but uses Oriya-specific text features that do not apply to Arabic text for word segmentation
Al-Dmour and Fraij, 2014	HPP for line extraction. Columnar projection profile (CPP) for connected components and gaps extraction Clustering to find the thresholds of within-word gap, between words gap, letter, sub-word and word lengths thresholds	84.8% correct segmentation with FCM clustering	AHDB; 25 images were used for different writers	- HPP is used for line segmentation which creates hard segmentation lines that split words components between lines and does not tolerate image or writer skew - Does not include processing to diacritics
Manmatha and Srimal, 1999	Space scale techniques for segmenting words	87% correct segmentation	Around 30 gray level images randomly picked from different sections of the George Washington Corpus of 6400 document images	- Does not handle touching lines as it eventually uses connected components to separate words - Ascenders and descenders may be lost during line segmentation

Reference	Technique/method	Performance Evaluation		Notes
		Result	Test Sample	
Bulacu, et al., 2007	Droplet line segmentation	62 missed (0.2%) 173 over-detected (0.5%) Total error (0.7%) Accuracy (99.3%)	32816 lines in the KdK collection	- Handles touching lines - Assumes the text is neat thus uses the horizontal projection profile to determine the starting point of line segmentation, this is not always true in random handwritten documents with crammed lines - Does not perform word segmentation - Customized for the KdK
Li, et al., 2008	Text line segmentation Script independent using PDF of the image and the level set method	Pixel hit rate: With ruled lines: 98% Freestyle: 93%	Results are based on 100 document images The UMD test data set (7,528 handwritten documents in 9 scripts (Cyrillic, Greek, Hebrew, Hindi, Korean, Japanese, Persian, Chinese, and Thai) scanned at 300dpi) Also a collection of 166,071 Arabic documents were collected for the experiments	- Handles touching lines - Language independent - Does not perform word segmentation - Does not include processing to diacritics
Surinta, et al., 2014	A* path planning – artificial intelligence	99.8%	1429 lines from the Saint Gall dataset and on 995 lines from Monk Line Segmentation (MLS) data set	- Handles crammed and touching lines - Sensitive to skew - No word segmentation - Does not include processing to diacritics

Reference	Technique/method	Performance Evaluation		Notes
		Result	Test Sample	
Kumar, et al., 2010	Graph-based; calculating similarity between text components based on local orientation detection and shortest path in graphs then using these similarities to cluster the text components using Affinity propagation and Breadth-first search	96% accuracy	125 Arabic document images	<ul style="list-style-type: none"> - Line segmentation - No word segmentation - Tolerates, detects, and corrects writer skew - Assigns diacritics to their words - Handles touching and crammed lines
AlKhateeb, et al., 2009	<p>Single text line images only</p> <p>Baseline detection: Peak of horizontal line in the lower half of the line Segmentation: Vertical projection with a manually set threshold Assumes the line is straight</p>	<p>Segmentation: Correct: 85% Under-segm: 9% Over-segm: 4% Misplaced: 2%</p>	<p>Baseline detection: IFN/ENIT: the first 1000 from sets a, b, c, and d: total is 4000 images</p> <p>Segmentation: 500 images</p>	<ul style="list-style-type: none"> - No line segmentation, only word segmentation of a single line - Does not include processing to diacritics
Srihari, et al., 2006	Paper is about word spotting but for word segmentation: feature extraction and neural network	<p>60% using a set of 7 extracted features</p> <p>Higher performance can be achieved using a more complex set of features</p>	A database of 20,000 word images contained in 100 documents; 10 writers writing 10 documents each	<ul style="list-style-type: none"> - Line segmentation is not discussed in the paper, thus we assume that it may not be capable to handle crammed and touching lines - Diacritics are not assigned to words

We aim to select an algorithm with good segmentation results and that achieves all of the requirements listed in Section 6.1.

The systems proposed by (Amin and Al-Sadoun, 1992), (Du, et al., 2008), and (Boiangiu, et al., 2014) do not have performance evaluation. We cannot speculate their performance and thus we exclude them from our selection.

Some techniques are designed to perform character segmentation. This is not required to avoid the function duplication as it is already performed by JU-OCR2. We exclude the systems proposed by (Motawa, et al., 1997) and (Dinges, et al., 2011).

The system proposed by (Kumar, et al., 2007) is designed for text body extraction out of images that contain graphical content. It does not handle line or word segmentation. Therefore, it is not suitable for our case of study and is excluded.

Tripathy and Pal (2006) system handles skewed and crammed lines but uses Oriya-specific text features that do not apply to Arabic text for word segmentation like the water reservoir technique for word separation. It is also excluded.

Each of the remaining algorithms performs either line segmentation or word segmentation. We are thus forced to combine a selected line segmentation algorithm with a word segmentation algorithm.

6.3.1 Line segmentation algorithm

Al-Dmour and Fraij (2014) line segmentation algorithm uses an enhanced HPP for line segmentation. Unavoidably, this creates hard straight segmentation lines that split parts of words components like ascender or descenders between different lines. This could form a problem to the word segmentation and recognition algorithm that follow as those split components may be processed as secondary components. In addition, this method does not include processing to diacritics. This approach is excluded.

Manmatha and Srimal (1999) system is unable to handle touching lines as it eventually uses connected components to separate words. Also, ascenders and descenders may be lost during line segmentation. This does not meet the requirements in section 6.1 and thus should be excluded.

Bulacu, et al. (2007) algorithm yielded outstanding results and could handle touching lines but it was custom-built for the KdK documents which have a strict document layout and are extremely tidy. This is not the case in many of the documents of the MADCAT set. The lines do not always start at the same point, images and lines suffer from skew, and lines can be crammed into a small space. Those problems are not accounted for in the algorithm and would degrade its performance in case applied to MADCAT. It is also excluded.

This leaves only three line segmentation approaches that could handle the requirements of Section 6.1 for line segmentation (Li, et al., 2008), (Surinta, et al., 2014), and (Kumar, et al., 2010).

As they all have shown competitive results, however, (Li, et al., 2008) disregards the assignments of diacritics to their respective lines, this creates problems especially when dealing with crammed lines, this violates the third and fourth requirements set in section 6.1. we have selected the approach presented in (Kumar, et al., 2010) since it is more suited for Arabic text documents and for the JU-OCR2 as it takes into account the languages diacritics and assigns them to their corresponding lines instead of being left unprocessed. Also, fortunately, this algorithm contains a complete de-skewing process. Skew angle is calculated for each part of line to find the local orientation and the image can be easily corrected based on this information. Surinta, et al. (2014) algorithm can be sensitive to skew as the water droplet may take a route that crosses a skewed line in its search for the least costly path causing wrong segmentation.

6.3.2 Word segmentation algorithm

The line segmentation algorithm we have selected segments the skewed lines into straight lines and assigns the diacritics to each line. Thus we do not need a complex word segmentation technique. We only need an algorithm that performs well.

Only two word segmentation algorithms remain, (AlKhateeb, et al., 2009) and (Srihari, et al., 2006). Bearing in mind that we only need a simple system since the line segmentation method does most of the work. Performance will be our only selection criteria for word segmentation. We select the approach of (AlKhateeb, et al., 2009) since it outperforms the other approach with 85% accuracy compared to 60% for the approach of (Srihari, et al., 2006).

7 RESULTING SYSTEM

The selections that we have made alter the original expected flow of the preprocessing system. Figure 22 shows the modified flow diagram of the preprocessing system.

The preprocessing system is designed and selected to handle the problems found on MADCAT image set: pepper noise, vertical lines, lined paper, text lines overlapping, skew, and bleed through text. With the aid of JU-OCR2, the integrated system is also expected to handle slant and intra-word spacing.

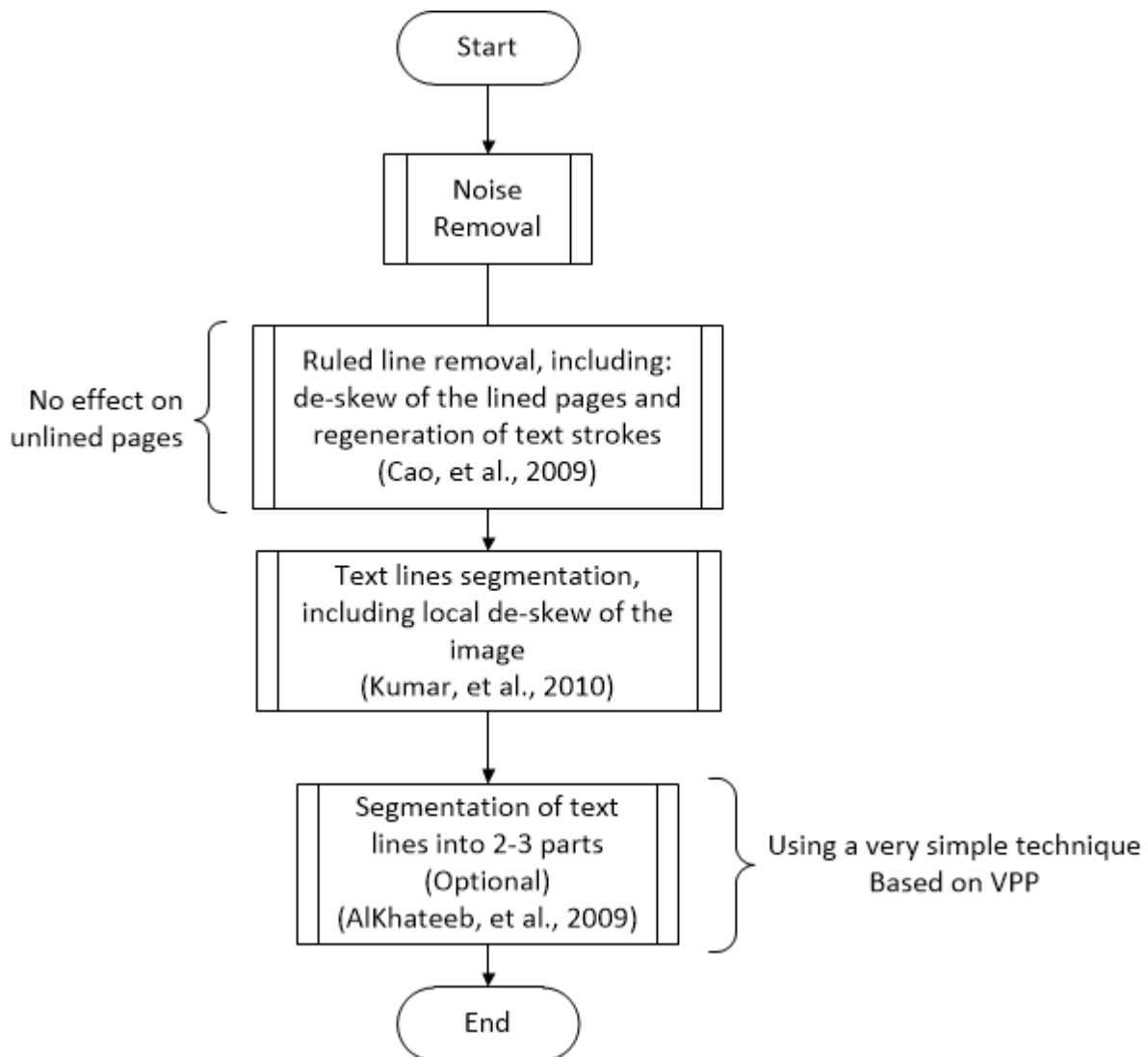


Figure 22. The proposed flow diagram of the preprocessing system.

The system starts with the simple noise removal algorithm that was proposed in Section 4.3. This algorithm includes two consecutive morphological opening operations with 3x3 and 5x5 structuring elements. Since bleed-through text and faint vertical lines share the attributes of pepper noise, the noise reduction system is capable to solve these three problems.

The ruled lines removal algorithm comes next. It starts with skew detection and correction of the page skew of the lined pages. The ruled lines are then detected and removed. Text strokes affected by the removal of the ruled lines are reconstructed regardless of the width of the text stroke or the intersection angle between the text stroke and the ruled line. This algorithm has no effect on unlined pages.

After the execution of the ruled lines removal algorithm, if the image was originally lined, then it would be free of noise, skew, and ruled lines. Otherwise, it will only be free of noise.

Then the text line segmentation algorithm is fed a general document that is free of noise and ruled lines. However, if the document is not originally lined, then it is not corrected for skew. The algorithm tracks each text line in the document and determines the local skew along different parts of the line. This is used to correct the page skew and writer skew at the same time. The lines are then extracted with no skew.

Having most of the problems handled, the word segmentation algorithm is fed images of individual lines that are free of noise and ruled lines. This significantly simplifies the operation of the word segmentation algorithm. The selected word segmentation algorithms only analyzes the VPP of the text line image and sorts the gaps to inter-word or intra-word gaps based on the gap size. This ends the function of the segmentation and preprocessing system.

The proposed system is fed document images of a single text body. It performs the necessary operations to output a sequence of single word images that are fed to the recognition system, JU-OCR2 that can handle single words. Thus, expanding the scope of the system from single word recognition to document recognition.

8 CONCLUSIONS, RECOMMENDATIONS, AND FUTURE WORK

In this thesis, we have identified the set of issues that a handwritten document image preprocessing system needs to handle by investigating a large set of handwritten document images, the MADCAT data set. Among the problems are the pepper noise, vertical scan lines, and bleed-through text. These problems were found to be equivalent in effect and were processed by a simple proposed noise removal algorithm.

Other problems included lined paper, skew, text segmentation at line and word levels, slant, external graphical elements, and writing errors. We concentrated our research on the most common problems and the ones with greatest impact on the performance of JU-OCR2 in recognizing Arabic handwritten words.

Then we investigated the preprocessing approaches available in attempt to extend the scope of the selected recognition system, JU-OCR2, from word recognition to document image recognition that could resolve the most common preprocessing issues and the ones with the deepest impact on the performance of the JU-OCR2; segmentation, nondestructive ruled line removal, and skew. To achieve this, we have based our selection of methods on approaches with high performance in their respective fields and tried to maximize the number of resolved issues in each approach. The resulting preprocessing system is selected to handle almost all the problems identified in the representative set of Arabic handwritten images, MADCAT set.

The selection criteria of the ruled line removal algorithm mandated that the algorithm should not create or leave distortion in processed image. The algorithm should manage levels of ruled lines degradation. The algorithm should have no effect on unlined images and is preferred to handle both vertical and horizontal ruled lines. The algorithm should also have competitive performance results.

The selected algorithm, the one proposed by (Cao, et al., 2009) has shown promising results at pixel level (2.07% pixel-level mismatch). It offered a full package of features including skew detection and removal, broken ruled-lines processing, not affecting unlined pages, and handling vertical and horizontal lines. The algorithm is also designed for high-resolution images like the images of MADCAT dataset.

The segmentation algorithm is required to handle line and word segmentation. It should separate touching or overlapping consecutive lines. It should be robust against crammed lines. It also is preferred to attach diacritics to their corresponding lines and words. The algorithm should also have competitive performance results.

After reviewing the available literature, we could not find a system that meets all of our requirements. Hence we were forced to select a line segmentation algorithm then combine it with a different word segmentation algorithm.

The selected line segmentation algorithm, the one proposed by (Kumar, et al., 2010), has 96% accuracy score on Arabic documents. It is resilient against crammed and touching lines. It calculates the local skew of each line segment. It also assigns diacritics to their respective lines.

MADCAT dataset contains over 42,000 images written by 355 native Arab scribes. The scribes were both males and females, left- and right-handed. Pens and pencils were used in writing. Lined and unlined paper was used in writing. Slow, normal, and fast writing speeds were used to create this dataset.

By selecting only two approaches and recommending some modifications, we can handle noise removal, segmentation, ruled line detection, nondestructive ruled line removal, document skew correction, and writer skew correction in a manner that is compatible with JU-OCR2 and with no functionality duplication.

Suggested future work would be to implement the selected algorithms, integrate them with JU-OCR2, and fine-tune the overall performance of the system. The system can also be generalized to different layouts of documents. Layout analysis could expand the scope of the system to documents with multiple text bodies, tables, and/or graphics. Binarization can also be investigated to accommodate grayscale and colored document images.

REFERENCES

- Abandah, G. and Khedher, M., (2009), Analysis of handwritten Arabic letters using selected feature extraction techniques. **International Journal of Computer Processing of Languages**, 22(01), pp.49-73.
- Abandah, G., Jamour, F., (2010), Recognizing handwritten Arabic script through efficient skeleton-based grapheme segmentation algorithm. **International Conference Intelligent Systems Design and Applications**, pp. 977–982
- Abandah, G. and Malas, T., (2011), Feature selection for recognizing handwritten Arabic letters. *Dirasat: Engineering Sciences*, 37(2).
- Abandah, G., Jamour, F., and Qaralleh, E. (2014), Recognizing handwritten Arabic words using grapheme segmentation and recurrent neural networks. **International Journal on Document Analysis and Recognition (IJ DAR)**: 1-17.
- Abandah, G. and Jamour, F., (2014). A Word Matching Algorithm in Handwritten Arabic Recognition Using Multiple-Sequence Weighted Edit Distances. **International Journal of Computer Science Issues (IJCSI)**, 11(3), p.18.
- Abd-Elmageed, W., Kumar, J., & Doermann, D. (2009), Page rule-line removal using linear subspaces in monochromatic handwritten arabic documents, **Document Analysis and Recognition IEEE**, 10th International Conference, 2009,768-772.
- Agrawal, M., & Doermann, D. (2011), Stroke-like pattern noise removal in binary document images, **In Document Analysis and Recognition (ICDAR)**, International Conference, IEEE, September, 2011,17-21.
- Ait-Mohand, K. and Paquet, T. (2013) OpenHaRT 2013 evaluation: description of the LITIS handwriting recognition system. **In Proceedings of the NIST 2013 OpenHaRT Workshop**.
- Al-Dmour, A., & Fraij, F. (2014), Segmenting Arabic Handwritten Documents into Text lines and Words, **International Journal of Advancements in Computing Technology**, 6(3): 109-119.
- Alginahi, Y. M. (2013). A survey on Arabic character segmentation, **International Journal on Document Analysis and Recognition (IJ DAR)**, 16(2): 105-126.
- AlKhateeb, J. H., Jiang, J., Ren, J., & Ipson, S. (2009), Interactive knowledge discovery for baseline estimation and word segmentation in handwritten Arabic text, **INTECH Open Access Publisher**.

- Amin, A., & Al-Sadoun, H. B. (1992), A new segmentation technique of Arabic text, In **Pattern Recognition, Vol. II. Conference B: Pattern Recognition Methodology and Systems**, Proceedings, 11th IAPR International Conferenc, IEEE, September, 1992, 441-445.
- Arivazhagan, M., Srinivasan, H., and Srihari S. N., (2007), A Statistical Approach to Handwritten Line Segmentation, in **Document Recognition and Retrieval XIV, Proceedings of SPIE**, San Jose, CA, pp. 6500T-1-11.
- Arvind, K. R., Kumar, J., & Ramakrishnan, A. G. (2007), Line removal and restoration of handwritten strokes, **In Conference on Computational Intelligence and Multimedia Applications**, International Conference, IEEE, 2007, 208-214.
- Barlas, P., Adam, S., Chatelain, C., & Paquet, T. (2014), A typed and handwritten text block segmentation system for heterogeneous and complex documents. **In Document Analysis Systems (DAS)**, 11th IAPR International Workshop, IEEE, April, 2014, 46-50..
- Blumenstein, M., Cheng, C.K. and Liu, X.Y., (2002), New preprocessing techniques for handwritten word recognition. In **Proceedings of the Second IASTED International Conference on Visualization, Imaging and Image Processing (VIIP 2002)**, ACTA Press, pp. 480-484.
- Boiangiu, C. A., Tanase, M. C., & Ioanitescu, R. (2014), Handwritten documents text line segmentation based on information energy, **International Journal of Computers Communications & Control**, 9(1): 8-15.
- Bukhari, S. S., Shafait, F., & Breuel, T. M. (2011), Improved document image segmentation algorithm using multiresolution morphology, **In IS&T/SPIE Electronic Imaging**, International Society for Optics and Photonics, January, 2011, 78740D-78740D.
- Bulacu M., Koert R., Schomaker L., Zant T., (2007), Layout analysis of handwritten historical documents for searching the archive of the Cabinet of the Dutch Queen , Proc. of 9th **International Conference on Document Analysis and Recognition (ICDAR 2007)**, **IEEE Computer Society**, vol. I, 23 – 26, pp. 357-361.
- Cao, H., Govindaraju, V., (2009). Preprocessing of Low-Quality Handwritten Documents Using Markov Random Fields. **IEEE Trans. Pattern Anal. Machine Intell.** 31, 84–96.
- Cao, H., Prasad, R., & Natarajan, P. (2009), A stroke regeneration method for cleaning rule-lines in handwritten document images, **the International Workshop on Multilingual OCR**, 4.

- Chan, T. F., & Vese, L. A. (2001), Active contours without edges, **IEEE Transactions on image processing**, 10(2): 266-277.
- Chen, J., & Lopresti, D. (2014), Model-based ruling line detection in noisy handwritten documents, **Pattern Recognition Letters**, 35: 34-45.
- Dave, N. (2015), Segmentation methods for handwritten character recognition, **International journal of signal processing**, image processing and pattern recognition, 8(4): 155-164.
- Dinges, L., Al-Hamadi, A., Elzobi, M., Al Aghbari, Z. and Mustafa, H., (2011), Offline automatic segmentation based recognition of handwritten Arabic words. **International Journal of Signal Processing, Image Processing and Pattern Recognition**, 4(4), pp.131-143.
- Du, X., Pan, W., & Bui, T. D. (2009), Text line segmentation in handwritten documents using Mumford–Shah model, **Pattern Recognition**, 42(12): 3136-3145.
- Garz, A., Fischer, A., Sablatnig, R., and Bunke, H., (2012), Binarization-free text line segmentation for historical documents based on interest point clustering, in Document Analysis Systems (DAS), **10th IAPR International Workshop**, pp. 95–99.
- Graves A. and Schmidhuber J. (2009), Offline handwriting recognition with multidimensional recurrent neural networks. **Advances in Neural Information Processing Systems 21**: 545-552.
- Huang, C., & Srihari, S. N. (2008), Word segmentation of off-line handwritten documents, **In Electronic Imaging**, International Society for Optics and Photonics, January, 2008, 68150E-68150E.
- Katsouros, V., & Papavassiliou, V. (2011), Segmentation of handwritten document images into text lines, **INTECH Open Access Publisher**.
- Kavallieratou, E., Lopresti, D., & Chen, J. (2011), Ruling line detection and removal, **In IS&T/SPIE Electronic Imaging**, International Society for Optics and Photonics, January, 2011, 78740V-78740V.
- Khademi, M., Golestani, M. R., Nikookar, A., & Farahani, A. (2013), A New Method for Detecting Segmentation Points in Persian Cursive Words, **Journal of Basic and Applied Scientific Research (JBASR)**, Special, 1.
- Kumar, S., Gupta, R., Khanna, N., Chaudhury, S., & Joshi, S. D. (2007), Text extraction and document image segmentation using matched wavelets and MRF model, **IEEE Transactions on Image Processing**, 16(8): 2117-2128.

- Kumar, J., Abd-Almageed, W., Kang, L., & Doermann, D. (2010), Handwritten Arabic text line segmentation using affinity propagation. **In Proceedings of the 9th IAPR International Workshop on Document Analysis Systems**, ACM, June, 2010, 135-142.
- Kumar, J., & Doermann, D. (2011), Fast rule-line removal using integral images and support vector machines, **In Document Analysis and Recognition (ICDAR)**, International Conference, IEEE, 2011, 584-588.
- Leifert, G., Labahn R., and Straub T. (2013), CITlab ARGUS for Arabic Handwriting. **Proceedings of the NIST 2013 OpenHaRT Workshop**.
- Lewis, M.P. (2009), *Ethnologue: Languages of the World*. SIL International, Dallas.
- Li, Y., Zheng, Y., Doermann, D., & Jaeger, S. (2008), Script-independent text line segmentation in freestyle handwritten documents, **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 30(8): 1313-1329.
- Likforman-Sulem, L., Hanimyan, A., Faure, C., (1995), A Hough Based Algorithm for Extracting Text Lines in Handwritten Documents, **Proceedings of the Third International Conference on Document Analysis and Recognition**, Montreal, Canada, pp. 774-777.
- Liolios, N., Fakotakis, N., Kokkinakis, G., (2002), On the generalization of the form identification and skew detection problem. **Pattern Recognition** 35, 253–264.
- Lopresti, D., & Kavallieratou, E. (2010), Ruling line removal in handwritten page images, **In Pattern Recognition (ICPR)**, 20th International Conference, IEEE, 2010, 2704-2707.
- Louloudis, G., Gatos, B., Pratikakis, I., & Halatsis, C. (2009), Text line and word segmentation of handwritten documents, **Pattern Recognition**, 42(12): 3169-3183.
- Luthy F., Varga T., Bunke H., (2007), Using Hidden Markov Models as a Tool for Handwritten Text Line Segmentation, **Ninth International Conference on Document Analysis and Recognition**, Curitiba, Brazil, pp. 8-12.
- Mahadevan U., Nagabushnam R. C., (1995), Gap metrics for word separation in handwritten lines, **Third International Conference on Document Analysis and Recognition**, Montreal, Canada, pp. 124-127.
- Manimurugan, S., Priya Mohan, and Sissy Annamma Johnson (2014), Handwritten Pattern Recognition-A Survey. **International Journal of Emerging Trends in Science and Technology** 1(01): 68150E-68150E.

- Manmatha R., Rothfeder J. L., (2005), A Scale Space Approach for Automatically Segmenting Words from Historical Handwritten Documents, **IEEE Trans. Pattern Analysis and Machine Intelligence**, Vol.27, No.8, pp. 1212-1225.
- Manmatha, R., & Srimal, N. (1999), Scale space technique for word segmentation in handwritten documents, **In International conference on scale-space theories in computer vision**, Springer Berlin Heidelberg, September, 1999, 22-33.
- Marti U. V. and Bunke H., (2001), Text Line Segmentation and Word Recognition in a System for General Writer Independent Handwriting Recognition, **Sixth International Conference on Document Analysis and Recognition**, Seattle, WA, USA, pp. 159-163.
- Motawa, D., Amin, A., & Sabourin, R. (1997), Segmentation of Arabic cursive script, **In Document Analysis and Recognition**, Proceedings, of the Fourth International Conference on, IEEE, August, 1997, 625-628.
- Mumford, D., & Shah, J. (1989), Optimal approximations by piecewise smooth functions and associated variational problems, **Communications on pure and applied mathematics**, 42(5): 577-685.
- Natarajan, P., Saleem, S., Prasad, R., MacRostie, E., & Subramanian, K. (2008), Multilingual offline handwriting recognition using hidden Markov models: A script-independent approach. In Arabic and Chinese Handwriting Recognition, **Springer Berlin Heidelberg**, 4768: 231-250.
- National Institute of Standards and Technology (NIST), NIST 2013 Open Handwriting Recognition and Translation Evaluation Workshop Proceedings, **http://www.nist.gov/itl/iad/mig/hart2013_wrkshp.cfm**, last visited in 08/12/2014.
- National Institute of Standards and Technology (NIST), OpenHaRT 2013 Information Page, **<http://www.nist.gov/itl/iad/mig/hart2013.cfm>**, last visited in 08/12/2014.
- Papavassiliou, V., Stafylakis, T., Katsouros, V., & Carayannis, G. (2010), Handwritten document image segmentation into text lines and words, **Pattern Recognition**, 43(1): 369-377.
- Parvez, Mohammad Tanvir, and Sabri A. Mahmoud (2013), Offline Arabic handwritten text recognition: a survey. **ACM Computing Surveys (CSUR)** 45(2): 1-35.
- Patel, C., Patel, A., & Shah, D. (2013), A Review of Character Segmentation Methods, **International Journal of Current Engineering and Technology**, 3(5): 2075-2078.

- Pechwitz, M., Maddouri, S.S., Märgner, V., et al. (2002) IFN/ENIT-Database of Handwritten Arabic Words. **Proceedings of the 7th Colloque International Francophone sur l'Ecrit et le Document, Hammamet, Tunis**: 129-136.
- Pu, Y. and Shi, Z., (1998), A Natural Learning Algorithm Based on Hough Transform for Text Lines Extraction in Handwritten Documents, **Proceedings of the 6 International Workshop on Frontiers in Handwriting Recognition**, Taejon, Korea, pp. 637-646.
- Renjini L., Rubeena B., (2015), A Comprehensive Study On Handwritten Character Recognition System. **IOSR Journal of Computer Engineering (IOSR-JCE)**, Volume 17, Issue 2, Ver. IV: 01-07.
- Saeed, Usman (2014), Automatic Recognition of Handwritten Arabic Text: A Survey. **Life Science Journal** 11(3s): 232-235.
- Saha, S., Basu, S., Nasipuri, M., & Basu, D. K. (2010), A Hough transform based technique for text segmentation, **arXiv preprint arXiv**: 1002-4048.
- Saleem, Shirin, Huaigu Cao, Krishna Subramanian, Matin Kamali, Rohit Prasad, and Premkumar Natarajan (2009), Improvements in BBN's HMM-based offline Arabic handwriting recognition system. **Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference, IEEE**: 773-777.
- Santosh, J., & Livingston, L. M. (2013), Text detection from documented image using image segmentation, **International Journal of Technology Enhancements and Emerging Engineering Research**, 1(4).
- Sarfraz, Muhammad, S. A. Mahmoud, and Zeehasham Rasheed (2007), On Skew Estimation and Correction of Text. **Computer Graphics, Imaging and Visualization (CGIV)**: 308-313.
- Sari, T., Souici, L., & Sellami, M. (2002), Off-line handwritten Arabic character segmentation algorithm: ACSA, **In Frontiers in Handwriting Recognition**, Proceedings. Eighth International Workshop on, IEEE, 2002, 452-457.
- Sauvola J. and Pietikäinen M. (2000), Adaptive document image binarization. **Pattern Recognition**, 33(2): 225–236.
- Seni G., Cohen E., (1994), External Word Segmentation of Offline Handwritten Text Lines, **Pattern Recognition**, 27(1): 41-52.
- Shi Z., Govindaraju V., (2004), Line Separation for Complex Document Images Using Fuzzy Runlength, **First International Workshop on Document Image Analysis for Libraries**, pp. 306.

- Shi, Z., Setlur, S., & Govindaraju, V. (2005), Text extraction from gray scale historical document images using adaptive local connectivity map, **Document Analysis and Recognition**, Proceedings, Eighth International Conference, IEEE, 2005, 794-798.
- Shi, Z., Setlur, S., & Govindaraju, V. (2010), Removing rule-lines from binary handwritten arabic document images using directional local profile, **In Pattern Recognition (ICPR)**, 20th International Conference, IEEE, 2010, 1916-1919.
- Smith, L.I, A tutorial on Principal Components Analysis; February 26, 2002.
- Srihari, S.N., Srinivasan, H., Huang, C. and Shetty, S., (2006), Spotting words in Latin, Devanagari and Arabic scripts. **Vivek-Bombay-**, 16(3), p.2.
- Surinta, O., Holtkamp, M., Karabaa, F., Van Oosten, J. P., Schomaker, L., & Wiering, M. (2014), A path planning for line segmentation of handwritten documents, **In Frontiers in Handwriting Recognition (ICFHR)**, 14th International Conference on, IEEE, September, 2014, 175-180.
- Tong A.N., Przybocki M., Maergner V., and El Abed H. (2013), NIST 2013 Open Handwriting Recognition and Translation (OpenHaRT13) evaluation. **Proceedings of the NIST 2013 Open Handwriting and Recognition Workshop**.
- Tripathy, N., & Pal, U. (2006), Handwriting segmentation of unconstrained Oriya text, **Sadhana**, 31(6): 755-769.
- Viola, P., & Jones, M. J. (2004), Robust real-time face detection, **International journal of computer vision**, 57(2): 137-154.
- Yanikoglu, B., Sandon, P. A., (1998) Segmentation of off-line cursive handwriting using linear programming, *Pattern Recognition*, 31(12) 1825-1833.
- Zahour A., Taconet B., Mercy P., Ramdane S., (2001), Arabic handwritten text-line extraction, In Proceeding(s) of **the Sixth International Conference on Document Analysis and Recognition (ICDAR)**, vol.37, pp. 281–285.
- Zheng, Y., Li, H., & Doermann, D. (2005), A parallel-line detection algorithm based on HMM decoding, **IEEE transactions on pattern analysis and machine intelligence**, 27(5): 777-792.

المعالجة القبلية والتجزئة للوثائق العربية المكتوبة للتعرف الآلي بصرف النظر عن الكاتب

إعداد
احمد شحده حلمي الحوراني

المشرف
الاستاذ الدكتور غيث علي عبدة

ملخص

في هذه الأطروحة اخترنا نظاما ممتازا للتعرف الضوئي على الحروف وهو نظام (JU-OCR2) الذي يعالج التعرف على مستوى الكلمة ثم اضفنا له نظام معالجة قبلية من أجل توسيع نطاقه لمستوى التعرف على صور الوثائق العربية. لقد قمنا بدراسة المشاكل التي يمكن أن يواجهها نظام المعالجة المسبقة من خلال دراسة مجموعة من الوثائق العربية المكتوبة بخط اليد وهي مجموعة MADCAT. وتشمل هذه المشاكل التشوش، والخطوط المسطرة، والانحراف والتقطيع لاسطر وكلمات ورشم النص وميلان النص والعناصر الرسومية الخارجية وأخطاء الكتابة وغيرها. لقد حددنا المشاكل الأكثر شيوعا التي سيتم مناقشتها في هذه الرسالة وتشمل هذه المشاكل إزالة الخطوط المسطرة وتقطيع السطر والكلمة والانحراف وإزالة التشوش. ثم بحثنا عن النهج المتبعة لحل هذه المشاكل من الأدبيات الموجودة. واخضعنا هذه النهج لمجموعة من معايير الاختيار مصممة للتوافق مع JU-OCR2 وتحقيق أفضل النتائج. تم تقييم جميع النهج ذات الصلة في الأدبيات. ونتيجة لهذا التقييم تم اختيار خوارزمية لإزالة الخطوط المسطرة، وخوارزمية لتقطيع السطور، وخوارزمية لتقطيع الكلمات. حيث تحتوي خوارزميات إزالة الخطوط المسطرة وتقطيع السطور المختارة على إجراءات مدمجة لإزالة الانحراف. كما طورنا خوارزمية لإزالة التشوش وتوصلنا إلى تعديلات وتوصيات لإنشاء نظام المعالجة المسبق المطلوب الذي يناسب نظام التعرف JU-OCR2.