ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

RETRIEVAL FROM REAL-LIFE AMHARIC DOCUMENT IMAGES

BINIAM ASNAKE

JUNE 2012

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

RETRIEVAL FROM REAL-LIFE AMHARIC DOCUMENT IMAGES

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By

BINIAM ASNAKE

JUNE 2012

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

RETRIEVAL FROM REAL-LIFE AMHARIC DOCUMENT IMAGES

By

BINIAM ASNAKE

Name and signature of members of the examining Board

Name	**Title**	**Signature**	**Date**
Melaku Girma (M.Sc)	Chairperson	_____	_____
Dr. Million Meshesha (Ph.D)	Advisor	_____	_____
Dr. Dereje Teferi (Ph.D)	Examiner	_____	_____

Dedication

To my father,

Ato Asnake Kefale,

for your love of education!

Acknowledgment

Before all, I praise the almighty **GOD** and his mother **ST. MARY** for making everything the way it is. Next, I am so glad to express my warm gratitude to my advisor **Dr. Million Meshesha (PhD)** for being with me in all the ups and downs of this work. He is honestly the GREATEST teacher and advisor I have ever known. Thank you so much for showing and guiding me to an interesting research direction and recognizing my potential in addition to the care, support and valuable comments you gave me. What you did is beyond acknowledgment.

I would like to offer special thanks to all my family members specially **Asnake Kefale**, **Elsabet G/Meskel**, Frehiwot A., Tigist A., Ephrem A., Biruktawit A. and Hiruy A. Special people (Anteneh Girma, Dr. Eyayu Molla, Eng. Mihret Demissie, Tsi, Girma Aweke, Negasi G., Endalk and Mekdy) who encourage, love and support me with all you have, I have nothing except THANK YOU.

Many thanks to Mr. Adane Letta, for providing me all the resources, the precious advices and supporting ideas you gave me. My sincere thanks goes to Mr. Daniel Mamo, head of library at the Institute of Ethiopian Studies, for providing me with valuable books, thesis and data for my thesis.

I am also indebted to Haramaya University for granting me the chance to attend my post-graduate study at Addis Ababa University. Thank you so much all my friends (in Addis Ababa and Haramaya) and classmates. You know who you are!

Biniam Asnake

Abstract

Bulk of real life documents contain vital information and knowledge about history, culture, economy, politics, religion and science that are available in written form in Ethiopic script. This knowledge ought to be shared and the advancement of technology and research in Information Retrieval (IR), Artificial Intelligence (AI) and related fields bring the need to digitize documents and make it available for public use. The two major approaches of retrieving information from document images are recognition-based (optical character recognition /OCR/) and recognition-free (document image retrieval without explicit recognition /DIR/). The first approach is a long term process, error-prone and registers minimized performance for noisy documents, where as document image retrieval without explicit recognition is the preferred one.

A few researches have been conducted to develop a recognition-free document image retrieval system that extracts information from document images relying on image features only. These systems are highly affected by noise in real life documents which results from paper aging, folding, scanning and printing errors. In this study, an attempt is made to integrate effective noise reduction and thresholding techniques to enhance the effectiveness of the system in searching within real-life document images. This study also improves the online searching process of the system by accepting multiple query terms then retrieving documents in recall-oriented manner and achieve 77.33% F-measure.

A combination of three noise reduction techniques: median, adaptive median and wiener filters, and three thresholding techniques: Otsu's, Niblack's and Sauvola's techniques are experimented in printed real-life documents plagued by low, medium, high and very high noise. Performance analysis shows that the best performing combination of denoising and thresholding techniques are wiener filtering and Otsu thresholding. Finally, the performance of the system is evaluated before and after the integration of the selected preprocessing techniques in which an average overall performance of 82.37% F-measure is registered in documents having low, medium, high and very high levels of noise. The major challenge is segmentation error where the current system either considers multiple separate words as one because of noise or a single word as multiple words when the noise is removed and the space between characters of a single word is large enough to be a word (segmentation threshold value) by the segmentation algorithm.

Table of Contents

List of Tables

List of Figures

List of Equations

List of Algorithms

List of Sample Codes (Listings)

List of Acronyms and Abbreviations

ADIRS:	Amharic Document Image Retrieval System
AI:	Artificial Intelligence
AMF:	Adaptive Median Filter
DAR:	Document Analysis and Recognition
DIP:	Digital Image Processing
DIR:	Document Image Retrieval
DIRS:	Document Image Retrieval System
DPI:	Densities Per Inch
DTW:	Dynamic Time Warping
IDF:	Inverse Document Frequency
IES:	Institute of Ethiopian Studies
IR:	Information Retrieval
MATLAB:	MATrix LABoratory
MSE:	Mean Square Error
OCR:	Optical Character Recognition
PSNR:	Peak Signal to Noise Ratio
TF:	Term Frequency

CHAPTER ONE

INTRODUCTION

1.1. Background

Communication and codification of information and knowledge can be done using writing, speaking, drawing, gesture and signs (non-verbal way). Modern ways of codification includes recording information in different multimedia formats such as text, image, audio, video and animation. Over the centuries, paper documents have been the principal instrument to make the progress of the humankind permanent. Nowadays, most information is still recorded, stored, and distributed in paper format. The widespread use of computers for document editing, with the introduction of personal computers and word-processors in the late 1980's had the effect of increasing, instead of reducing, the amount of information held on paper [1].

Load of textual information and knowledge is available in different writing styles like handwritten, typewritten, or computer printout formats. With storage becoming cheaper and imaging devices becoming increasingly popular, efforts are on the way to digitize and archive large quantity of multimedia data, especially text from every page of printed, typewritten or handwritten documents [2] [3]. The advancement of technology and research in the fields of Information Retrieval (IR), Artificial Intelligence (AI), Digital Image Processing (DIP) and Pattern Recognition brings the need to digitize, store, query, search and retrieve different documents to make the information available for public use efficiently and accurately [4].

Scanning and storing documents as images clearly has advantages over storing hardcopy, and unlike converted documents, digital document images can provide a precise, high-quality representation of the original document, including graphics and images. Loss of material, misfiling, limited numbers of each copy, and even degradation of materials are common problems, and may be improved by document analysis techniques. Having the ability to capture an accurate representation of both the content and structure is especially important considering the complex, multi-modal documents that can be created with today's desktop publishing systems [5] [6].

Ethiopia, a country at the Horn of Africa, has its own writing system in addition to its own calendar and other tourist attraction sites and artifacts. Ethiopia is a mosaic of about 100 languages that can be classified into four groups—Semitic, Cushitic, Omotic, and Nilotic [7].

Among the various languages in Ethiopia, Amharic is the dominant one that it is spoken by roughly 30% of the population as a mother tongue, and an additional 20% as a second language, totaling about half of the population [22]. It is also one of the Semitic languages that use Ethiopic script and the official language of the country and medium of communication and working language in most of the regional states (such as Addis Ababa, Amhara, etc) [14]. Additionally, Amharic is the second most spoken Semitic language in the world, next to Arabic. It had been the language of the official court and the dominant political elite in Ethiopia since the rise of the Solomonic dynasty at the end of the 13th century [23] [24]. It is the most commonly learned second language throughout the country, next to English and the medium of instruction in secondary schools and higher educational institutions [11] [25].

Amharic language is very useful to scholars in anthropology, history, and archaeology as well as in linguistics, since Ethiopia is a land of great history and treasures. Ethiopia provides a rich resource to geologists and biologists [10]. The language has its own script called Ethiopic or Amharic or Ge'ez script using which a number of documents are published in the form of books, magazine, newspapers, etc [11].

Huge collections of documents are archived written in Amharic/Ethiopic script in formats such as handwritten, typewritten or computer printouts [7], which need to be converted into electronic form for easy searching and retrieval as per users' information need or query. Suffice is to mention the huge amount of documents piled high in information centers, libraries, offices in the form of correspondence letters, magazines, newspapers, pamphlets, books [9]. These documents contain information related to religion, history, literature, politics, economics, philosophy, tradition, culture, nature and other essential evidences of the different nations, nationalities and people of Ethiopia. Revealing and retrieving the knowledge preserved using Ethiopic script will have a positive impact on social and historical studies.

Digitizing this information and allowing public access to these documents will decrease the problems of manual searching and retrieval, and provide vital importance for researchers,

historians, tourists and in general to build good image about the country and the people in general. Information retrieval strategies, algorithms, techniques, and tools can be used for this purpose.

Information retrieval is the process of searching for information that satisfies information need of users from large collection of in unstructured documents (composed of text, image, audio, video and other multimedia objects) [12]. Ricardo and Ribeiro-Neto [13] define information retrieval (IR) as "the representation, storage, organization of, and access to information items". Retrieval is an operation of accessing information from document collection [14]. The representation and organization of the information items which is stored in files and databases should provide the user with easy access to the information in which s/he is interested.

Modern technology has made it possible to produce, process, store, and transmit document images efficiently. In an attempt to move toward the paperless office, large quantities of printed documents are digitized and stored as images in files and databases. The popularity and importance of document images as an information source is evident [15]. Searching and retrieval from document images can be conducted with the help of recognition-based (OCR), recognition-free (DIR) or a combination of these two approaches [11]. Optical Character Recognition (OCR) systems take scanned images of paper documents as input and automatically convert them into digital format for computer-aided data processing. Designing robust recognizers that are applicable for documents varying in quality, fonts, sizes and styles is known to be a long term solution.

OCR systems can be applied for the purpose of document image retrieval. However, the performance of the system relies heavily on the quality of the scanned images. It deteriorates severely if the images are of poor quality or have complicated layout [16]. Other demerits of using OCR, according to Tan et al. [17], are requiring human correction, language dependence and waste of effort. A promising alternate direction is document image retrieval without explicit recognition or recognition-free approaches that search for relevant documents in the image domain [11].

Various researches have been conducted since 1997 to adopt OCR techniques to Amharic text of different formats such as computer printed [9] and [26] - [31], typewritten [32], handwritten [33]

[34] and special handwriting style ("Qum Tsehfet"/'ቁም ጽሕፈት') [8]. However, developing an applicable OCR system is known to be a long-term process. On the other hand, with the present digitization of document images at large scale, there is a pressing need for document understanding tools and methods [11]. To come up with a robust system, researchers propose document image retrieval as a short term process. An alternative approach to OCR and a promising direction is to search for relevant documents using only image properties, without explicit recognition [3] [35].

An immediate need for effective access to digital libraries initiates document retrieval without explicit recognition. Document image retrieval (DIR) or recognition-free retrieval is based on keyword spotting, where the document image is first segmented into words, and the user's keywords are located in the image by word-to-word matching. Document image retrieval without explicit recognition becomes a very attractive field of research with the continuous growth of interest and requirements for the development of the modern society. It is a direct approach to access the digitized document image itself with certain level of performance. Especially for historical printed and handwritten document images, it is a promising approach to design an applicable retrieval system [11].

Digital images have great impact on our day-to-day life activities as well as on technology research areas [18]. The process of retrieving information from document image runs from digitization to searching for relevant documents based on users' query [19]. First, paper-based documents should be digitized using any of the image acquisition devices such as scanner and digital camera. After digitization, six major tasks are involved to retrieve information from document images. These are preprocessing, segmentation, feature extraction, indexing, matching and displaying relevant document images in ranked order. The overall structure of document image retrieval system consists of two different parts: the offline and the online procedures [20].

The digitized image is the raw input to document analysis. The aim of the preprocessing module is to prepare the image for retrieval. Preprocessing involves binarization (or thresholding) and skew correction. It also undergoes some image enhancements such as filtering out noise and increasing the contrast [11]. The objective of binarization is to automatically choose a threshold that separates the foreground from the background region. After that, it converts a gray-scale or color image to binary image (0s and 1s).

Then, the image is segmented to separate the set of words in a document. Segmentation occurs at two levels. On the first level, text, graphics and other parts are separated. On the second level, text lines and words in the image are located [11]. Image segmentation leads to more compact image representations by partitioning an image into a set of disjoint words to represent document images [19].

Segmentation is followed by feature extraction, which involves extracting the meaningful information from the document images [11] [19]. Based on feature vectors, document images are indexed and then searched to retrieve relevant documents as per information need of users. To speed up searching from a document collection, it is necessary to develop an index. Indexing is an offline process that enables to organize document images using extracted features of word images [19].

During searching the system accepts query from users and then the text query is converted to image (by a process called query rendering) in order to be compared with set of document images. To represent word images, an image feature based on the pixel values in the predefined area is extracted. The idea is to calculate vector value for the image and represent each word image in one vector [14]. Then, matching in document images can identify the word images of the documents that are more similar to the query word through the extracted feature vectors [19]. Similarity measurement is a central problem in computer vision and pattern recognition, which has wider applications in multimedia retrieval, especially in content based image retrieval [11].

The search results are finally presented to the user in ranked order. The main aim of ranking is to sort the retrieved documents according to their degree of relevance to the query provided by the user. Consequently, ranking algorithm is at the core of IR systems and operates according to basic premises of document relevance in which different set of premises yield distinct document retrieval models. Finally, performance evaluation of information retrieval systems is important to measure the efficiency and effectiveness of the retrieval process [11] [19] and identify further research areas.

Datasets collected by means of image sensors are generally contaminated by noise. Hence, noise filtering techniques needs to be integrated so as to ease retrieval of relevant documents from noisy and historical documents.

1.2. Statement of the Problem and Justification

Handwritten, typewritten and printed Amharic real-life and historical documents contain numerous information and knowledge regarding different aspects of human life. These documents should be digitized and be accessible to the public with the objective of sharing and utilizing the numerous knowledge embedded in them. For this dream to come true, researchers are moving towards recognition-free approach (DIR) since OCR has various limitations.

The groundbreaking research work on document image retrieval without explicit recognition (DIR) for Amharic documents together with English and Hindi text is conducted by Million [11]. The researcher presented an efficient word image matching scheme that is crucial for content-based document image retrieval from printed text collection. Mesfin [35] designed a document image retrieval system that provides response to user's query without the requirement of character recognition. As continuation of Mesfin's [35] work, Abreham [14] designed a retrieval system that accepts user query and searches from Amharic document image corpus. Amharic DIRS was conducted by Tilahun [25] with the aim of designing and developing a generic model for Amharic Document Image Retrieval.

The latest work by Adane [19] explored feature extraction and matching techniques that are crucial to enhance the performance of the document image retrieval system. Adane's [19] system accepts a single query from the user, similar to [14] [35]. That means, no information retrieval model is used to determine which documents are relevant and which are not.

Since noise removal technique is not integrated to the system, the performance of the system is greatly affected, as shown in Table 1.1. To improve the effectiveness of the system, integrating noise removal scheme is recommended by all the previous researchers [11] [14] [19] [25] [35].

Level of Noise	**Recall**	**Precision**	**F-measure**
Low	100%	68.88%	80.46%
Medium	100%	50.00%	66.66%
High	83.33%	44.04%	55.82%
Average	**94.44%**	**54.3%**	**67.64%**

Table 1.1: Summary of Adane's system performance on different levels of noise

Noise might be introduced by scanning devices and transmission media. Also aging, photocopying, faxed documents, imperfect instruments, problems with data acquisition process and inferring natural phenomena, transmission errors and compression can all degrade document images [11] [19] [21] [18]. As stated by Jawahar et al. [2], word images particularly in newspapers and old books are of extremely poor quality. Popular artifacts in printed document images include excessive dusty noise, large ink-blobs joining disjoint characters or components, vertical cuts due to folding of the paper, cuts at arbitrary direction due to paper quality or foreign material, degradation of printed text due to the poor quality of paper and ink from facing pages.

According to Million [11], one of the major reasons for the difficulty of Amharic document recognition and retrieval is degradation. Denoising (noise reduction, cleaning and removal) is often necessary and the first step to be taken before the image data is analyzed. However, image denoising still remains a challenge for researchers because noise removal introduces artifacts and causes blurring of the images [18]. Among other things, effective retrieval from document images especially real-life documents is only possible, if noise detection and removal module is developed and integrated to the DIRS.

Hence, the main concerns of this study are to explore the application of different image preprocessing techniques such as noise filtering and thresholding, as well as adopting an information retrieval (IR) model and enable the system to search using multiple query words in order to design an effective retrieval system for real-life printed Amharic document images.

Towards this end, the following research questions are explored and answered in the present work.

- What are the special features of the Ethiopic script and the noise distribution prevalent in printed real-life documents?
- What suitable image preprocessing techniques for noise reduction and thresholding need to be integrated to improve the quality of real-life documents?
- How much improvement is registered on the performance of DIR system after the integration of preprocessing techniques and multiple querying?
- How can IR model be integrated to enable the system to accept multiple query terms?

1.3. Objectives of the Study

The following general and specific objectives are outlined in order to solve the problems that initiate this study.

1.3.1. General Objective

The main objective of this research is to integrate effective image preprocessing techniques of noise reduction and thresholding to enhance the effectiveness of relevant document retrieval from printed real-life Amharic document images by accepting multiple word queries from the user.

1.3.2. Specific Objectives

To realize the aforementioned general objective, the following specific objectives are drawn.

- To review previously conducted researches and other literatures (such as books, journal articles, conference papers and the Internet) on real-life document image retrieval for conceptual understanding of principles, theories, approaches and algorithms.
- To study the special features of the writing style and noise distribution on real-life Amharic documents.
- To investigate and select techniques for noise removal from and binarization/thresholding of Amharic document images.
- To prepare Amharic document image corpus and queries that are used to measure the performance of the prototype system.
- To propose a suitable noise filtering and thresholding algorithm that improves the quality of degraded document images and integrate it to the previously developed Amharic Document Image Retrieval System (DIRS).
- To adopt IR model and improve text rendering and matching methods to accept multiple word query from the user.
- To evaluate the effectiveness of the system on real-life Amharic document images and report the findings of the research.

1.4. Scope and Limitation of the Study

Perpetuating and advancing from previous researches [11] [14] [19] [25] [35] on developing an Amharic document image retrieval system, this research endeavors to study, adopt and integrate image preprocessing that mainly focuses on noise removal and thresholding, and improve searching by adopting IR model and allowing users to query using multiple words. The other tasks in document image retrieval such as feature extraction, matching, indexing and searching were done by the previous researchers. Therefore, these modules were modified to fit the current study.

For performance measure, the document images are categorized into four levels of noise (low, medium, high and very high). The retrieval scheme considers only document images in Amharic language among the Ethio-Semitic languages with specific reference to real-life documents of computer printout formats. For testing, data corpus was collected from different sources such as books, magazines, newspapers, regulation documents.

The limitation of this study is that, among preprocessing tasks, skew correction was not implemented due to time limitation. Since there is no corpus available to evaluate the performance of the system, the dataset contains limited number of printed document images.

1.5. Methodology of the Study

The term methodology actually comes from an old Greek word, denoting the practice of analyzing different methods, implying a set or system of methods, principles and rules for regulating a given discipline [36]. According to Kothari [37], research methodology is a way to systematically solve the research problem. It may be understood as a science of studying how research is done scientifically. Therefore, the following methods, techniques and tools have been used with the aim of achieving the general objective of this research.

1.5.1. Literature Review

Extensive literatures from journal articles, books, conference papers and the Internet have been reviewed for understanding concepts related to document image retrieval and image preprocessing. Further review was also conducted to be acquainted with the previous researches

on Amharic document image retrieval system and real-life documents. Then, the best techniques and tools were analyzed, selected, adopted and/or modified with the intention of developing a system. As the system is developed for real-life Amharic documents, the history, characteristics and challenges of the language was also studied.

1.5.2. Dataset Collection

To evaluate the performance of the proposed system, real-life Amharic documents that contain different font types, styles, size and level of noise are digitized. While doing this research, the necessary Amharic document images with sufficient amount were collected from books, magazines, newspapers, regulations and religious documents in addition to dataset used in the previous work [19]. A total of 4,974 word images were extracted from the different sources.

1.5.3. Implementation Tools

To develop the prototype, MATLAB programming language was used. We implemented some algorithms using MATLAB because it provides easy access to images. The other reason we selected MATLAB is because it has rich implementations of the widely used image processing algorithms and its ease of use. Some algorithms we adopted in this research are already implemented in MATLAB. Therefore, there is no need of writing a new implementation.

MATLAB is a high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numeric computation. Using the MATLAB product, we can solve technical computing problems faster than C/C++ programming languages [38]. MATLAB is used in a wide range of applications, including signal and image processing, communications, control design, test and measurement, financial modeling and analysis, and computational biology.

Among MATLAB's toolbox, we used the *Image Processing Toolbox*. This toolbox software is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations including linear and nonlinear filtering and filter design, image analysis and enhancement, morphological operations, spatial image transformations and more [38].

All previous attempts to develop Amharic document image retrieval system [14] [19] [35] used *Java™* programming language for text rendering, segmentation, feature extraction, indexing, similarity measurement, retrieval and ranking the relevant documents retrieved. The reason for selecting Java™ is because it is a fully object-oriented programming language that is easy-to-use, robust, safe, supporting Unicode encoded text and lots of advanced built-in methods for image processing. The other reason is, the researcher has a good programming knowledge and skill in both MATLAB and Java™ programming languages. As a result, it is convincing to develop the prototype of Amharic document image retrieval system using MATLAB and Java™ programming languages.

MATLAB code can be integrated easily with other languages and applications, and distributed [38]. Accordingly, after developing the image preprocessing module in MATLAB, we used *MATLAB Builder JA* software to easily and successfully integrate the prototype of the proposed system with the previous works.

1.5.4. Testing Procedure

Different image preprocessing algorithms were explored in order to come up with a clean document. These are median, wiener2 and adaptive median filtering (AMF) for noise reduction and Otsu, Niblack and Sauvola thresholding algorithms. The performance of the algorithms is measured using the common image quality measurement of Peak Signal to Noise Ratio (PSNR) which is calculated based on the Mean Square Error (MSE). The MSE represents the cumulative squared error between the compressed and the original image, whereas PSNR represents a measure of the peak error. The lower the value of MSE, the lower the error. The PSNR block computes the peak signal-to-noise ratio, in decibel (dB), between two images. This ratio is often used as a quality measurement and the higher the PSNR, the better the quality of the reconstructed image [38] - [41].

Finally, the best performer image preprocessing technique is selected and integrated with the previous system where the retrieval effectiveness was measured using the three most widely used measures: recall, precision and F-measure [14].

Recall is the fraction of the relevant documents which has been retrieved from the corpus while precision is the fraction of the retrieved documents which are relevant. Recall and precision have been used extensively to evaluate the performance of retrieval algorithms. In many situations, the use of a single effectiveness measure which combines recall and precision could be more appropriate. One such measure is the harmonic mean (F-measure) of recall and precision [13].

1.6. Significance of the Study

Different documents articulated and printed in Ethiopic script are piled high in information centers, libraries, museums, and government and private organizations. Manually accessing these printed documents is tiresome and time consuming [14]. There is bulk of real-life and historical printed, typewritten, handwritten and special handwritten documents available that needs to be digitized and accessible via the Internet and digital libraries [11].

This research work attempts to integrate image preprocessing and multiple word querying modules to the previously developed Amharic document image retrieval system [11] [14] [19] [25] [35]. This is a major leap forward to realize the dream of developing full-fledged, effective and efficient document image retrieval system for Ethiopic script. The results of the study have significant contributions to preserve and properly utilize real-life materials and the knowledge embedded in the documents.

Several governmental, non-governmental and private institutions with large collection of real-life and historical documents such as libraries, museums, churches and other information centers benefit from using the output of this study to preserve history, to permit public access to the document images and create virtual offices and libraries by digitizing the documents they have. The ranked and retrieved documents provide vital information for historians, archeologists, religious and social researchers, politicians, academicians and other professionals.

This system can also be used to develop to Portable Document Format (PDF) readers with ‘find’ functionality if document images of books, magazines and/or newspapers are scanned and stored as PDF, specially to search from real-life document images.

All Ethio-Semitic languages (such as Amharic, Tigrigna, Ge’ez, Gurage, Hareri, Silti, etc) and other languages use Ethiopic script for codifying and preserving information and knowledge.

Therefore, the output of this research can be used to develop DIRS for these languages with little modification. This contributes a lot by minimizing the time and effort spent to develop a document image retrieval system from scratch.

In general, the output of this research can be used to develop information retrieval system and/or search engine for Amharic language especially for noisy real-life document images.

1.7. Organization of the Study

This thesis is organized into five chapters. The first chapter discusses the background of the study and statement of the problem. It also presents general and specific objectives of the study, methodology of the study, scope and limitation of the research and application of the investigated results.

In chapter two, literature review on information retrieval, document image processing and retrieval is presented. Moreover, a brief review of the history, development and characteristics of the Amharic writing system and the different type of documents, global and local related works on document image retrieval are discussed.

Chapter three describes the proposed image preprocessing techniques and algorithms. Three noise filtering and three thresholding techniques are briefly discussed with their equation and algorithms. The evaluation measures that are used for measuring the performance of each noise removal algorithms are presented.

Chapter four emphasizes the integration of noise reduction and thresholding techniques and experimental results that are used to confirm the validity of the proposed image preprocessing techniques to retrieve relevant information from noisy Amharic document images. Further experimentation on multiple word querying is also presented.

Finally, based on the findings of the study, conclusion and recommendations of the research are stated in chapter five.

CHAPTER TWO

LITERATURE REVIEW

With the rising popularity and importance of document images as an information source, information retrieval (IR) in document image corpus has become a growing and challenging problem [15]. IR principles enable us to design a search engine that is used for searching and retrieval of relevant documents from large collections [11].

2.1. Information Retrieval

According to Manning et al. [12], information retrieval (IR) is the technology for finding material (usually documents) of an unstructured nature that satisfies information need of users from within large collections. Many information retrieval tools have been developed for retrieving information from textual documents, but they are not applicable to digitized documents. This initiates the study of the strategies of retrieving information from document images. For example, a user facing a large number of imaged documents on the Internet has to download each one to see its contents before knowing whether the document is relevant to his/her interest, e.g., by looking for keywords. Obviously, it is of practical value to study a method that is capable of notifying the user, prior to downloading, whether a document image contains information of interest to him/her [15].

Research in multimedia information retrieval is also getting closer attention in recent years. This has paved the way for a large number of new techniques and systems for content-based image retrieval (CBIR), content-based video retrieval (CBVR) and text indexing and retrieval. These techniques are used to build many successful prototype systems for indexing and retrieval of multimedia data [11]. Most of the present day digital libraries aim at archiving printed books which are not available on-line. They scan pages from the book and make them accessible with some additional meta-data.

2.2. Retrieval from Document Images without Explicit Recognition

In recent years, there has been much interest in the area of Document Image Retrieval (DIR) because it solves the problems of recognition-based retrieval and it is a short-term process [11]. The success of DIR greatly depends on document image processing (DIP) [15].

The field of digital image processing refers to processing digital images by means of a digital computer [42]. An important line of research is Document Image Retrieval (DIR) that aims at finding relevant documents relying on image features only. Until today, the largest portion of documents belonging to libraries is made by printed books and journals. The electronic counterparts of these physical objects are scanned documents that are traditionally the main subject of DIR research [43].

The main question that a DIR system seeks to answer is whether a document image contains words which are of interest to the user, while paying no attention to unrelated words [15]. In the recognition-free retrieval approaches, the similarity computation between the indexed documents and the query is made at the feature level, avoiding the explicit recognition during the indexing [43].

2.3. Steps in Document Image Retrieval

In many businesses today, imaging systems are being used to store images of pages to make storage and retrieval more efficient [15]. As a result, document images have become a popular information source in our modern society, and information retrieval in document image corpus is an important topic in knowledge and data engineering research.

To make billions of volumes of traditional and legacy documents available and accessible on the Internet, they are scanned and converted to digital images using digitization equipment [15]. Thus, data capture of documents by optical scanning or by digital video yields a file of picture elements, or pixels that is the raw input to document analysis. These pixels are samples of intensity values taken in a grid pattern over the document page, where the intensity values may be: OFF (0) or ON (1) for binary images, 0-255 for gray-scale images, and 3 channels of 0-255 color values for color images [6].

Binary images are a special type of intensity image where pixels can only take on one of two values, black or white. Binarization also results in binary images. According to Burger and Burge [44], the image data in a grayscale image consist of a single channel that represents the intensity, brightness, or density of the image. The other type of image data, color images encode the primary colors red, green, and blue (RGB), typically making use of 8 bits per component. In these color images, each pixel requires 24 bits to encode all three components, and the range of each individual color component is (0 … 255).

The steps in a typical document image retrieval system are described by scholars differently [6] [4] [19]. Figure 2.1 summarizes the major steps involved in DIR.

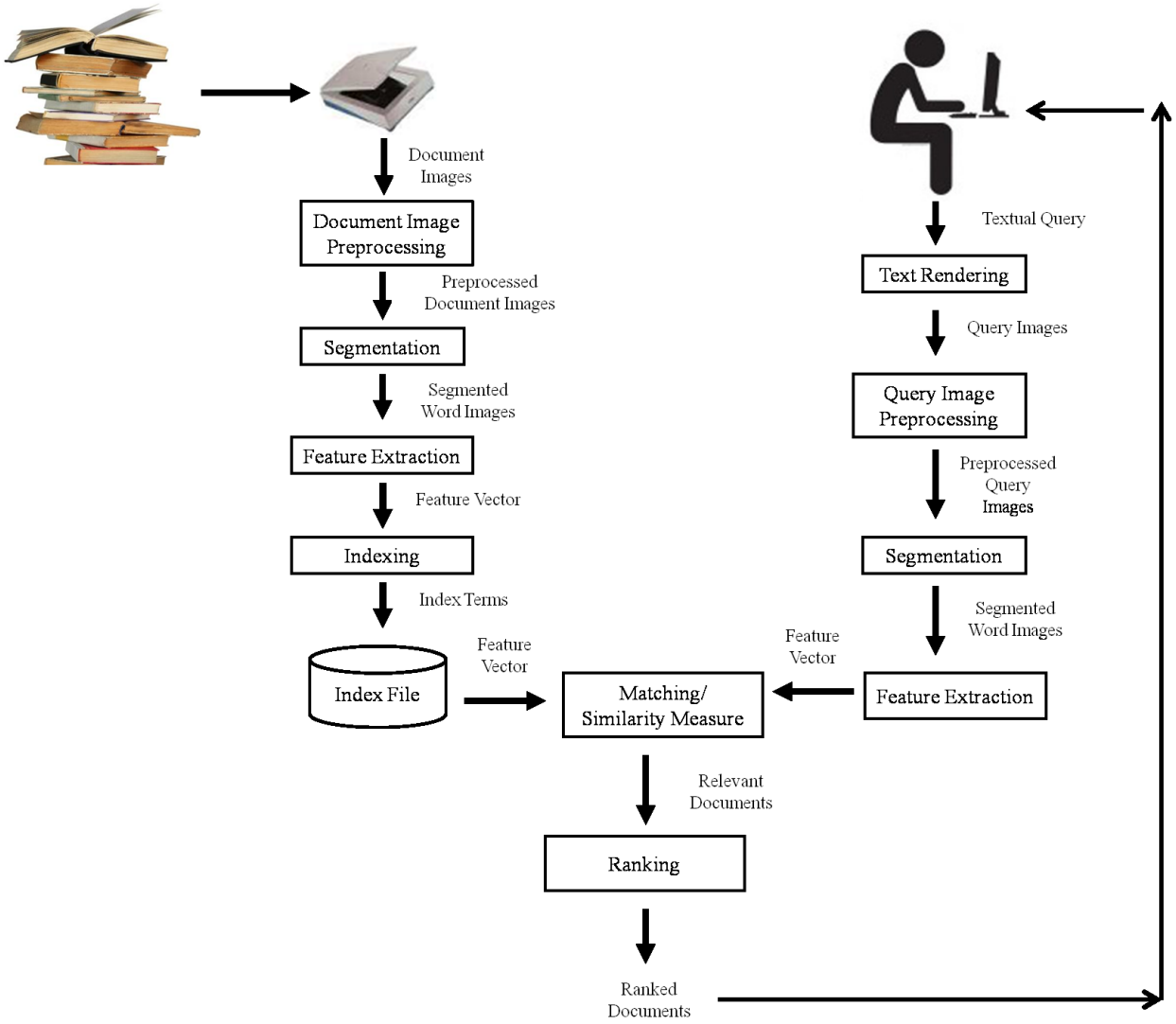


Figure 2.1: The overall architecture of the Document Image Retrieval System (DIRS)

The first task that must be done in order to manipulate and process images is digitization. Image acquisition is the process of acquiring or obtaining the image of document in color, gray level or binary format. A digitizer such as camera or scanner converts hardcopy documents to digital data [4]. After digitization, the next vital process in document analysis is to perform preprocessing on this image to prepare it for further analysis. This is discussed in section 2.4.

The retrieval tasks that should be conducted once images include pre-processed are segmentation, feature extraction, indexing, matching and ranking. Segmentation is the process of identifying the objects of our interest. Segmentation occurs on two levels. On the first level, if the document contains both text and graphics, these are separated for subsequent processing by different methods. On the second level, segmentation is performed on text by locating columns, lines and finally words [6].

Features are a representation of objects like word images. Features should also be distinct with respect to their neighborhood, stable with respect to noise, and complementary with respect to other features [11]. Feature extraction involves the identification of the meaningful information from the document image so that it reduces the storage required [4]. Feature extraction is the problem of gathering information from raw data, which is most relevant for a given application [11]. The features that are extracted from whole image are known as the global features and the features that are extracted from blocks identified during segmentation or from subdivision (sub-sectioning) of the document are known as local features [4].

To speed up searching from a document collection, it is necessary to organize documents using word images they contain [14]. A number of indexing structures are available including inverted file, signature file, suffix tree, suffix-trie [13]. For finding the similar instances of a word, the query word is matched with the other words in the dataset [45]. This is known as matching.

Matching in document images can identify the word of the documents that are more similar to the query word through the extracted feature vectors using different similarity measurements such as Euclidean, Cosine similarity, Cross Correlation, Dynamic time warping (DTW), and Manhattan distance [19] [20].

Ranking is used to sort the retrieved documents according to their degree of relevance to the query provided by the users. Accordingly, ranking algorithm is at the core of IR systems and

operates according to basic premises of document relevance in which different set of premises yield distinct document retrieval models [12] [13] [14].

2.4. Image Preprocessing

The images collected by different type of sensors are generally contaminated by different types of noises [41]. Paper documents are very sensitive to degradation of its integrity. Therefore, before manipulating the information in the image, preprocessing tasks must be conducted. Preprocessing is the first phase of document analysis [4] [6]. The purpose of preprocessing is to improve the quality of the image being processed. It makes the subsequent phases of image processing like segmentation, feature extraction and retrieval of documents easier [46]. The major preprocessing tasks while working with images are noise reduction, binarization or thresholding and skew correction [1].

2.4.1. Noise Reduction

One of the more important factors affecting digital image quality is image noise, which, if present in any significant quantity, can noticeably degrade image quality. The quality of digital images is affected by many factors, including the conditions under which the image was captured, the resolution of the optics used to capture the image and other physical characteristics of the image capture system, the characteristics of the solid state device used to capture the image and the methods used to digitize the electrical signals generated by the solid state device [39].

Karthikeya [47] define image noise as “the random variation of brightness or color information in images produced by the sensor and circuitry of a scanner or digital camera”. Another definition by Kavallieratou and Stamatatos [48] states that “noise is anything that is irrelevant with the textual information (i.e., foreground) of the document image”.

2.4.1.1. Types of Noise in Image

There are various approaches to classify image noise. In the classification proposed by [49], there are four kinds of noise: physical noise, digitization noise, filtering noise and storage/transmission noise.

- **The Physical Noise** – it is related to damages to the physical integrity and readability of the original information of a document. Moghaddam [50] classified physical noise in to internal and external noise. Internal noise includes paper aging, paper texture, carbon copy effect, scratches, cracks, and inadequate printing. External noise on the other hand, includes folding marks, filing and staple punching, stains, thorn-off regions, worm holes, reader´s annotations and highlighting, physical blur, and sunburn. Detail discussion of physical noise may be referred in [49].
- **The Digitalization Noise** – This is the noise introduced by the digitalization process. Several problems may be clustered in this group such as: inadequate digitalization resolution, unsuitable palette, framing noises, skew and orientation, lens distortion, geometrical warping, out-of-focus digitized images and motion noises.
- **The Filtering Noise** – Is unsuitable manipulation of the digital file may degrade the information that exists in the digital version of the document (instead of increasing it). The introduction of colors not originally present in the document due to arithmetic manipulation or overflow is an example of such a noise.
- **The Storage/Transmission noise** – refers to the noise that appears either from storage algorithms with losses or from network transmission.

The main type of noise that is prevalent in real-life and historical documents are presence of smear, strains, background of big variations and uneven illumination, seepage of ink are factors that impede (in many cases may disable) the legibility of the documents [48].

Based on Million [11] and Karthikeya [47], we classified categories of degradations or noise commonly observed in printed documents into eight. These artifacts are salt-and-pepper, cuts, blobs, erosion, shot, gaussian, quantization and non-isotropic noise. Figure 2.2 shows four of these artifacts i.e. salt-and-pepper, cuts, blobs and erosion.

I. **Salt-and-Pepper Noise**, also called impulse and speckle noise, or just dirt is a prevalent artifact in poorer quality document images (such as poorly thresholded faxes or poorly photocopied pages) [6]. Salt and pepper noise contains random occurrences of both black and white intensity values [51]. It is distributed all over the image flipping white pixels to black if it is a background and black pixels to white if it is a foreground. This appears as isolated pixels or pixel regions of ON noise in OFF backgrounds or OFF noise (holes)

within ON regions, and as rough edges on characters and graphics components. It is reduced by performing filtering on the image where the background is “grown" into the noise specks, thus filling these holes.

II. **Cuts** noise corrupts some part of an image by flipping black pixels into white. The occurrence of cuts in a document image breaks continuity of the shape of characters (or components). It happens due to print font quality, folding of paper, etc.

III. **Blobs** affect some part of the image by flipping pixel values to black. Blobs occur due to large ink drops within the document image during printing, faxing and photocopying. The existence of these noises merges separate characters or components.

IV. **Erosion** of pixels happens mostly at the boundary of the image due to the imperfections in scanning.

V. **Shot Noise**: The dominant noise in the lighter parts of an image from an image sensor is typically that caused by statistical quantum fluctuations, that is, variation in the number of photons sensed at a given exposure level; this noise is known as photon shot noise.

VI. In **Gaussian Noise**, each pixel in the image will be changed from its original value by a small amount. A histogram, a plot of the amount of distortion of a pixel value against the frequency with which it occurs, shows a normal distribution of noise. While other distributions are possible, the Gaussian (normal) distribution is usually a good model, due to the central limit theorem that says that the sum of different noises tends to approach a Gaussian distribution.

VII. **Quantization Noise** (or Uniform Noise) is the noise caused by quantizing the pixels of a sensed image to a number of discrete levels. It has an approximately uniform distribution, and can be signal dependent, though it will be signal independent if other noise sources are big enough to cause dithering, or if dithering is explicitly applied.

VIII. **Non-Isotropic Noise**: Some noise sources show up with a significant orientation in images. For example, image sensors are sometimes subject to row noise or column noise.

2.4.1.2. Importance of Noise Reduction

Noise reduction is the process of removing noise from a signal or image [47]. As we discussed in the above sections, noise is always present in digital images—it is captured at the same time

Figure 2.2: Sample Amharic word images with Cuts, Blobs, Salt-and-Pepper and Erosion of pixels (shown from top to bottom)

the image is captured and is part of the image data recorded by the imaging device. To the extent noise is present in an image, the true image is obscured and image detail is lost. The most important reason to reduce noise is to obtain easy way of recognition and retrieval of documents where extraneous features will otherwise cause subsequent errors in recognition [4].

In artistic, documentary and casual photography, noise has a negative impact on the aesthetics of digital images (they simply look worse), and in scientific and commercial applications of digital imaging (such as medical imaging, computer vision, and astronomy), noise can obscure important image details. In any case, it is worthwhile to attempt to reduce the noise inherent in digital images. Noise reduction can be accomplished at both the hardware level, by improving the design of image capturing hardware, and the software level, by attempting to detect and remove the noise from existing images [39].

It is common for libraries to provide public access to historical and ancient document image collections. It is common for such document images to require specialized processing in order to remove background noise and become more legible [48].

2.5. Image Filtering

In the field of digital image processing, the primary technique to remove noise from any type of image is filtering. When an image is acquired by a scanner, camera or other imaging system, it may be corrupted by random variations in intensity called noise, variations in illumination or poor contrast that must be dealt with the early stages of image processing [51].

Image restoration is the art and science of improving the quality of an image based on some absolute measure [52]. It usually involves some means of undoing a distortion that has been imposed.

Filtering consists in transforming an input image *i(k, l)* into an output image $i_o(k,l)$, which is also called the filtered version of *i(k,l)*. The output value $i_o(k,l)$ is usually a function of input values in a local neighborhood around position *(k,l)* [52]. The objective in the design of a filter to reduce noise is that it remove as much of the noise as possible while retaining the entire signal [6].

In some literatures such as [39] [42], filtering is classified as low-pass and high-pass.

I. **Low Pass Filter** is the basis for most smoothing methods. An image is smoothed by decreasing the disparity between pixel values by averaging nearby pixels [39]. Using a low pass filter tends to retain the low frequency information within an image while reducing the high frequency information. An example is an array of ones divided by the number of elements within the kernel, such as the following 3 by 3 kernel:

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

The above array is an example of one possible kernel for a low pass filter. Other filters may include more weighting for the center point, or have different smoothing in each dimension.

II. **High Pass Filter** is the basis for most sharpening methods. An image is sharpened when contrast is enhanced between adjoining areas with little variation in brightness or darkness [39].

A high pass filter tends to retain the high frequency information within an image while reducing the low frequency information. The kernel of the high pass filter is designed to increase the brightness of the center pixel relative to neighboring pixels. The kernel array usually contains a single positive value at its center, which is completely surrounded by negative values. The following array is an example of a 3 by 3 kernel for a high pass filter [39] [42]:

$-\frac{1}{9}$	$-\frac{1}{9}$	$-\frac{1}{9}$
$-\frac{1}{9}$	$\frac{8}{9}$	$-\frac{1}{9}$
$-\frac{1}{9}$	$-\frac{1}{9}$	$-\frac{1}{9}$

The traditional and proven classification into linear and nonlinear filters [42] [52] [53] is based on the mathematical properties of the filter function; i.e., whether the result is computed from the source pixels by a linear or a nonlinear expression.

2.5.1. Linear Filter

Linear filters combine the pixel values in the support region in a linear fashion; i. e., as a weighted summation [53]. Linear smoothing filters are good filters for removing Gaussian noise and in most cases, the other types of noise as well. A linear filter is implemented using the weighted sum of the pixels in successive windows. Typically, the same pattern of weights is used in each window, which means that linear filter is spatially invariant. Linear smoothing filters remove high-frequency components, and the sharp detail in the image is lost [51].

If different filter weights are used for different parts of the image, but the filter is still implemented as a weighted sum, then the linear filter is spatially varying [51]. Linear filters are usually used in conjunction with other nonlinear operations [52]. One of the widely used linear filter is mean filter.

2.5.1.1. Mean Filter

One of the simplest linear filters is implemented by a local averaging operation where the value of each pixel is replaced by the average of all the values in the local neighborhood:

$$f(i,j) = \frac{1}{M} \sum_{(i,j)\in N} g(i,j) \qquad (2.1)$$

, where M is the total number of pixels in the neighborhood N [42].

When designing linear smoothing filters, the filter weights should be chosen so that the filter has a single peak, called the main lobe, and symmetry in the vertical and horizontal directions. A typical pattern of weights for a *3 x 3* smoothing filter is:

$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$
$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{8}$
$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$

According to Gonzalez and Woods [42], four algorithms of mean filter are available. These are arithmetic, geometric, harmonic and contraharmonic mean filter.

2.5.1.2. Arithmetic Mean Filter

This is the simplest of the mean filters. Let S_{xy} represent the set of coordinates in a rectangular sub-image window (neighborhood) of size *m x n*, centered at point *(x, y)*. The arithmetic mean filter computes the average value of the corrupted image *g(i, j)* in the area defined by S_{xy}. The value of the restored image *f* at point *(i, j)* is simply the arithmetic mean computed using the pixels in the region defined by S_{xy}. In other words,

$$f(i,j) = \frac{1}{mn} \sum_{(i,j) \in S_{xy}} g(i,j) \qquad (2.2)$$

A mean filter smoothes local variations in an image, and noise is reduced as a result of blurring.

2.5.1.3. Geometric Mean Filter

An image restored using a geometric mean filter is given by the expression

$$f(x,y) = \left[\prod_{(s,t) \in Sxy} g(s,t) \right]^{\frac{1}{mn}} \qquad (2.3)$$

Here, each restored pixel is given by the product of the pixels in the subimage window, raised to the power *1/mn*. A geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to lose less image detail in the process.

2.5.1.4. Harmonic Mean Filter

The harmonic mean filtering operation is given by the expression

$$f(x,y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}} \qquad (2.4)$$

The harmonic mean filter works well for salt noise, but fails for pepper noise. It does well also with other types of noise like gaussian noise.

2.5.1.5. Contraharmonic Mean Filter

The Contraharmonic mean filter yields a restored image based on the expression

$$f(x,y) = \frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}} \qquad (2.5)$$

, where *Q* is called the *order* of the filter.

This filter is well suited for reducing or virtually eliminating the effects of salt-and-pepper noise. For positive values of *Q*, the filter eliminates pepper noise. For negative values of *Q* it eliminates salt noise. It cannot do both simultaneously. Note that the contraharmonic filter reduces to the arithmetic mean filter if *Q = 0*, and to harmonic mean filter if *Q = -1*.

In general, the arithmetic and geometric mean filers (particularly the latter) are well suited for random noise like gaussian and uniform noise. The contraharmonic filter is well suited for impulse noise, but it has the disadvantage that it must be known whether the noise is dark or light to select the proper sign for *Q*.

2.5.2. Nonlinear Filter

Linear filters have an important disadvantage when used for smoothing or removing noise: all image structures, including points, edges, and lines, are also blurred, and the quality of the whole image is evenly reduced (see Figure 2.3). This effect cannot be avoided, and thus the use of linear filters for noise removal is limited [44].

Like linear filters, nonlinear filters compute the result at some image position (u, v) from the pixels inside the moving region $R_{u,v}$ of the original image. The filters are called "nonlinear" because the source pixel values are combined by some nonlinear function [53].

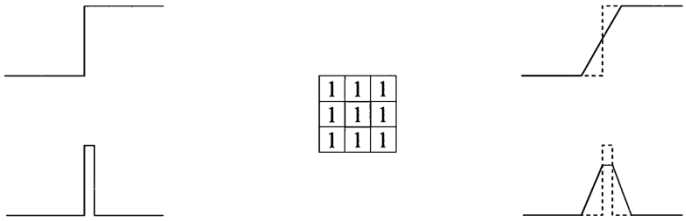


Figure 2.3: Nonlinear filters - Any image structure is blurred by a linear smoothing filter.

Important image structures such as step edges (top) or thin lines (bottom) are widened, and the local contrast is reduced. Any filter that is not weighted sum of pixels is a nonlinear filter. Nonlinear filters can be spatially invariant, meaning that the same calculation is performed regardless of the position in the image, or spatially varying. The median filter is a spatially invariant nonlinear filter [51].

2.5.2.1. Minimum and Maximum Filters

The simplest of all nonlinear filters are the minimum and maximum filters, defined as:

$$I'(u,v) \leftarrow \min\{I(u+i, v+j) \mid (i,j) \in R\}, \qquad (2.6)$$

$$I'(u,v) \leftarrow \max\{I(u+i, v+j) \mid (i,j) \in R\} \qquad (2.7)$$

The minimum filter removes the white (salt) dots because any single white pixel within the filter region is replaced by one of its surrounding pixels with a smaller value. In other words, minimum filter is useful for finding the darkest points in an image. However, the minimum filter

at the same time widens all the dark structures in the image. The reverse effects can be expected from the maximum filter. Any single bright pixel is a local maximum as soon as it is contained in the filter region *R*. White dots (and all other bright image structures) are thus widened to the size of the filter, while now the dark ("pepper") dots disappear [44].

2.5.2.2. Midpoint filter

The midpoint filter simply computes the midpoint between the maximum and minimum values in the area encompassed by the filter:

$$f(x,y) = \frac{1}{2}[\max\{g(s,t)\} + min\{g(s,t)\}] \qquad (2.8)$$

It works best for randomly distributed noise, like Gaussian or uniform noise.

2.5.2.3. Alpha-trimmed Mean Filter

Suppose that we delete the *d/2* lowest and the *d/2* highest intensity values of *g(s,t)* in the neighborhood S_{xy}. Let $g_r(s,t)$ represent the remaining *mn – d* pixels. A filter formed by averaging these remaining pixels is called an alpha-trimmed mean filter:

$$f(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t) \qquad (2.9)$$

where the value of *d* can range from *0* to *mn – 1*. When *d = 0*, the alpha-trimmed filter reduces to the arithmetic mean filter and if we choose *d = mn - 1*, the filter becomes a median filter. For other values of *d*, the alpha-trimmed filter is useful in situations involving multiple types of noise, such as a combination of salt-and-pepper and gaussian noise.

2.5.2.4. Median Filter

It is impossible to design a filter that removes any noise but the attempt to come up with a filter that keeps all the important image structures intact because no filter can discriminate which image content is important to the viewer and which is not [44]. The main problem with local averaging operations is that they tend to blur sharp discontinuities in intensity values in an image. An alternative approach is to replace each image pixel value with the median of the gray values

in the local neighborhood. Filters using this technique are called median filters [53]. Median filter works as follows [42].

$$I'(u,v) \leftarrow \text{median}\,\{I(u+i, v+j) \mid (i,j) \in R\} \qquad (2.10)$$

The median of *2K + 1* pixel values p_i is defined as:

$$median\,(p_0, p_1, \ldots, p_{K,\ldots,} p_{2K}) \triangleq p_K \qquad (2.11)$$

i.e., the center value p_k if the sequence (p_o, …, p_{2k}) is sorted ($p_i < p_{i+i}$).

The value of the pixel at *(u, v)* is included in the computation of the median. Median filters are quite popular because, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size [42]. In addition to this, median filters are very effective in removing salt and pepper and impulse noise while retaining image details because they do not depend on values which are significantly different from typical values in the neighborhood. Median filters work in successive image windows in a fashion similar to linear filters. However, the process is no longer a weighted sum.

For example, take a *3 x 3* window and compute the median of the pixels in each window centered around *(i, j)*:

1) Sort the pixels into ascending order by gray level.
2) Select the value of the middle pixel as the new value for pixel (*i, j)*.

Equation 2.10 defines the median of an odd-sized set of values, and if the side length of the rectangular filters is odd (which is usually the case), then the number of elements in the filter region is odd as well. In this case, the median filter does not create any new pixel values that did not exist in the image before. If, however, the number of elements is even *(2K for some K > 0)*, then the median of the sorted sequence *(p_0, …, p_{2k-i})* is defined as the arithmetic mean of the two middle values,

$$median\,(p_0, \ldots, p_{K-1,\ldots,} p_{K,\ldots,} p_{2K-1}) \triangleq (p_{K-1} + p_K) \,/\, 2\, p_K \qquad (2.12)$$

In general, an odd-sized neighborhood is used for calculating the median. However, if the number of pixels is even, the median is taken as the average of the middle two pixels after sorting.

2.5.2.5. Adaptive Filters

Once selected, the filters discussed thus far are applied to an image without regard for how image characteristics vary from one point to another. In this section, we take a look at two adaptive filters whose behavior changes based on statistical characteristics of the image inside the filter region defined by the *m x n* rectangular window S_{ij} [42].

2.5.2.5.1. Adaptive, local noise reduction filter

The simplest statistical measures of a random variable and reasonable parameters on which to base an adaptive filter are its mean and variance. The filter is to operate on a local region, S_{ij}. The response of the filter at any point *(i, j)* on which the region is centered is to be based on four quantities [42]: (a) *g(i, j)*, the value of the noisy image at *(i, j)* (b) δ_η^2, the variance of the noise corrupted *f(i, j)* to form *g(i, j)* (c) m_L, the local mean of the pixels in S_{ij} and (d) δ_L^2, the local variance of the pixels in S_{ij}.

According to Gonzalez and Woods [42], the filter shows one of the following behavior.

- If δ_η^2 is zero, the filter should return simply the value of *g(i, j)* . This is the trivial, zero-noise case in which *g(i, j)* is equal to *f(i, j)*.
- If the local variance is high relative to δ_η^2, the filter should return a value close to *g(i, j)*. A high local variance typically is associated with edges, and these should be preserved.
- If the two variances are equal, the filter should return the arithmetic mean value of the pixels in S_{xy}. This condition occurs when the local area has the same properties as the overall image, and local noise is reduced simply by averaging.

An adaptive expression for obtaining $f(i,j)$ based on these assumptions may be written as [42]:

$$f(i,j) = g(i,j) - \frac{\delta_\eta^2}{\delta_L^2}[g(i,j) - m_L] \qquad (2.13)$$

The only quantity that needs to be known or estimated is the variance of the overall noise, δ_η^2. The other parameters are computed from the pixels in S_{ij} at each location *(i, j)* on which the filter window is centered.

2.5.2.5.2. Adaptive Median Filter (AMF)

The median filter discussed previously performs well if the spatial density of the impulse noise is not large. In contrary to this, adaptive median filtering can handle impulse noise with large probabilities. An additional benefit is that it seeks to preserve details while smoothing non-impulse noise, something that the "traditional" median filer does not do. Like other filters, AMF works in a rectangular window area S_{ij}. Unlike those filters, the AMF changes (increases) the size of S_{ij} during filter operation, depending on certain conditions. The output of the filter is a single value used to replace the value of the pixel at *(i, j)*, the point on which the window S_{ij} is centered at a given time. The purpose of this algorithm are to remove impulse noise, smoothing of other noise and reduce distortion, like excessive thinning or thickening of object boundaries [41].

The advantages of AMF over standard median filter includes the standard median filter does not perform well when impulse noise is greater than 0.2, while the adaptive median filter can better handle these noises. The other advantage is the adaptive median filter preserves detail and smooth non-impulsive noise, while the standard median filter does not [41].

2.6. Binarization or Thresholding

Binarization converts the acquired image to binary format. The objective in methods for binarization is to automatically choose a threshold that separates the foreground and background information. Selection of a good threshold is often a trial and error process [4] [6].

Image analysis systems use binarization as a standard procedure to convert a gray-scale image to binary form. An ideal binarization algorithm would be able to perfectly discriminate foreground from background, thus, removing any kind of noise that obstructs the legibility of the document image (see Figure 2.4). The binary image is ideal for further processing. After the removal of background noise, it is possible for the document images to remain in gray-scale form [48].

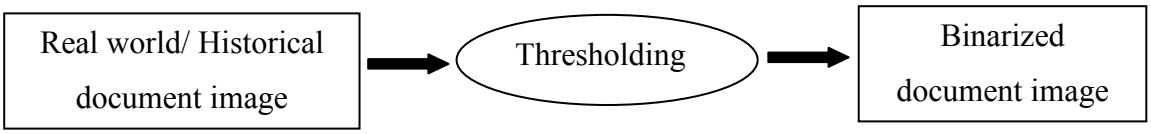


Figure 2.4: The process of thresholding along with its inputs and outputs

2.6.1. Binarization Approaches

Kavallieratou and Stamatatos [48] divided binarization approaches into general-purpose and document image-specific methods.

General-purpose methods are able to deal with any image. Therefore, they do not take into account specific characteristics of document images. On the other hand, document image-specific methods attempt to take advantage of document image characteristics (e.g., background pixels is the majority, foreground pixels are in similar gray-scale tones etc). In many cases, such methods are variations of general-purpose approaches. Although it is reasonable the latter approaches should be more effective when dealing with historical document images, recent results show that general-purpose methods can be more reliable under certain conditions.

Accordingly, O'Gorman and Kasturi [6] and Kavallieratou and Stamatatos [48] classified the type of thresholding methods into two. These are global and local/adaptive thresholding.

2.6.1.1. Global Thresholding

If the pixel values of the components and those of the background are fairly consistent in their respective values over the entire image, then a single threshold value can be found for the image. This use of a single threshold for all image pixels is called global thresholding. The pixels of the image are classified into text or background according to a global threshold. Usually, such methods are simple and fast.

There are a number of drawbacks to global threshold selection based on the shape of the intensity distribution. The first is that images do not always contain well-differentiated foreground and background intensities due to poor contrast and noise. They cannot be easily adapted in case the background noise is unevenly distributed in the entire image (e.g., smear or strains). Secondly, especially for an image of sparse foreground components, such as for most graphics images, the

peak representing the foreground will be much smaller than the peak of the background intensities. This often makes it difficult to find the valley between the two peaks. In addition, reliable peak and valley detection are separate problems unto themselves.

2.6.1.2. Local or Adaptive Thresholding

Since a single global threshold value cannot be used even for a single image due to non-uniformities within foreground and background regions, different threshold values are required for different local areas. This is known as adaptive thresholding. The pixels of the image are classified into text or background according to a local threshold determined by their neighboring pixels.

A common way to perform adaptive thresholding is by analyzing gray-level intensities within local windows across the image to determine local thresholds. The threshold is continuously changed through the image by estimating the background level as a two-dimensional running-average of local pixel values taken for all pixels in the image. Such methods are more adaptive and can deal with different kinds of noise existing in one image. On the other hand, they are significantly more time-consuming and computationally expensive.

The main problem with any adaptive binarization technique is the choice of window size. The chosen window size should be large enough to guarantee that a large enough number of background pixels are included to obtain a good estimate of average value, but not so large as to average over non-uniform background intensities.

O'Gorman and Kasturi [6] also discuss about how to choose a thresholding method. They state that whether global or adaptive thresholding methods are used for binarization, one can never expect perfect results. Depending on the quality of the original, there may be gaps in lines, ragged edges on region boundaries, and extraneous pixel regions of ON and OFF values.

2.7. Skew Detection and Correction (De-Skewing)

A text line is a group of characters, symbols, and words that are adjacent relatively close to each other, and through which a straight line can be drawn (usually with horizontal or vertical orientation) [6]. Deviation of the baseline of the text from horizontal direction is called skew.

The dominant orientation of the text lines in a document page determines the skew angle of that page. Skew is result of improper paper feeding into the scanner. During the document scanning process, the whole document or a portion of it can be fed through the loose-leaf page scanner. Some pages may not be fed straight into the scanner, however, causing skewing of the bitmapped images of these pages. So, document skew often occurs during document scanning or copying [4].

A document originally has zero skew, where horizontally or vertically printed text lines are parallel to the respective edges of the paper, however when a page is manually scanned or photocopied, non-zero skew may be introduced. This effect visually appears as a slope of the text lines with respect to the x-axis, and it mainly concerns the orientation of the text lines; Since such analysis steps as OCR, retrieval and page layout analysis most often depend on an input page with zero skew, it is important to perform skew estimation and correction before these steps. Also, since a reader expects a page displayed on a computer screen to be upright in normal reading orientation, skew correction is normally done before displaying scanned pages [4] [6]. Skew detection is one of the primary tasks to be solved in document image analysis system, and it is necessary for analyzing a document before further processing. The document, if not placed properly on the scan surface, can introduce skew in the resultant document image. Therefore, the document image may also need de-skewing. After skew detection, the image is usually rotated to zero skew angles [15]. An example of skewed document image is displayed in Figure 2.5 below.

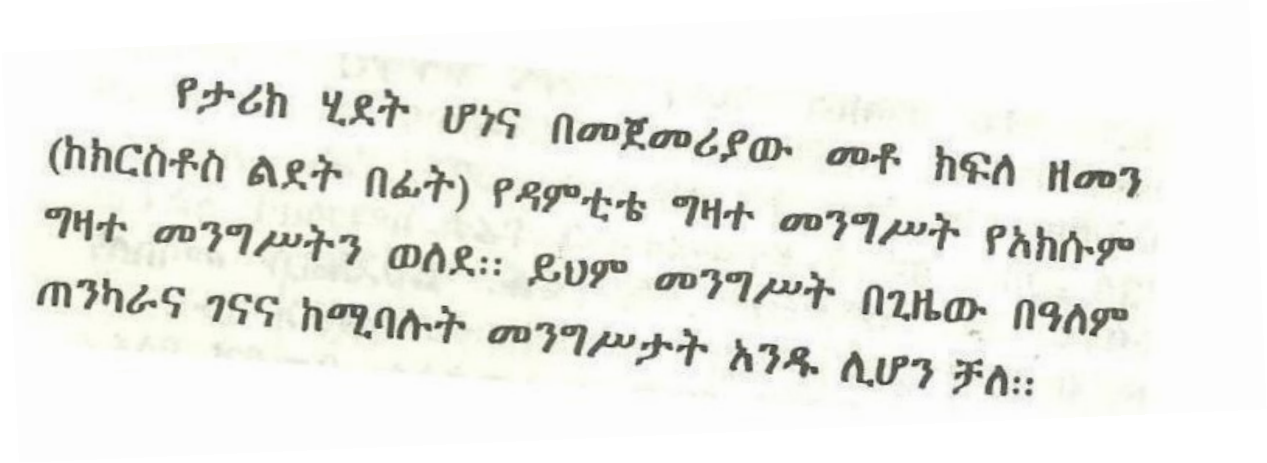


Figure 2.5: A sample skewed real-life printed Amharic document

2.8. The Amharic Language and the Ethiopic Script

Language is one of the fundamental aspects of human behavior and it constitutes a crucial component of our live [25]. Every language is built by using set of sounds that are used in the society. These sounds are represented by different combination of strokes so that people could put their ideas and thoughts on hard materials like paper and animal skin using these symbols [8].

Spoken language as a purely phonetic medium of communication, no longer suffice, where its limitation in space and time are not comparable to the demands of the progressive development of civilization. Therefore, people devised writing as a remedy [19]. This process of putting ideas and events using symbols makes it possible to preserve knowledge, historical events, different findings, and various kinds of information. Human knowledge is being transferred from one generation to another by means of writing which uses those symbols of the language that represent either the alphabet or the phoneme [8].

Writing is a means by which people record, objectify, and organize their activities and thoughts through images and graphs [35]. According to Lawrence [54], writing provides a way of extending human memory by imprinting information into media which is less changeable than the human brain. In addition, writing uses to hand cultural development down to the future generations as a secure possession for further cultivation [19].

2.8.1. Writing Systems

Writing systems are components of knowledge systems which assist in synthesizing ideas, thoughts, and activities through the use of signs, symbols or other pictorial renderings [26]. Writing systems were preceded by proto-writing, systems of ideographic and/or early mnemonic symbols which are used as means of communication. The best known examples are [11]: Jiahu Script (6600 BC), Vinca script (4500 BC) and early Indus script (3500 BC). The first true alphabetic writing appeared around 2000 BC, as a representation of language developed in Egypt.

Writing systems (or scripts) can be categorized into the following [11]: (i) Logographic script (a character is used to represent one concept, e.g. Chinese characters), (ii) Syllabic or Abugida script (consonants are written as the main letters, and then special symbols are used to indicate which vowels follow the consonant, e.g. Amharic characters, Indian Devanagari characters), (iii)

Alphabetic (the sounds in a language can be represented by an appropriate consonant and vowel alphabet, e.g. Latin alphabet), and (iv) Abjad (a consonantal alphabet in which vowels are not written, e.g. Arabic alphabet).

In Ethiopia, there are more than 200 different dialects spoken [11]. Amharic language is mainly spoken in Ethiopia and Eritrea. Figure 2.6 depicts the genetic structure of the Amharic language.

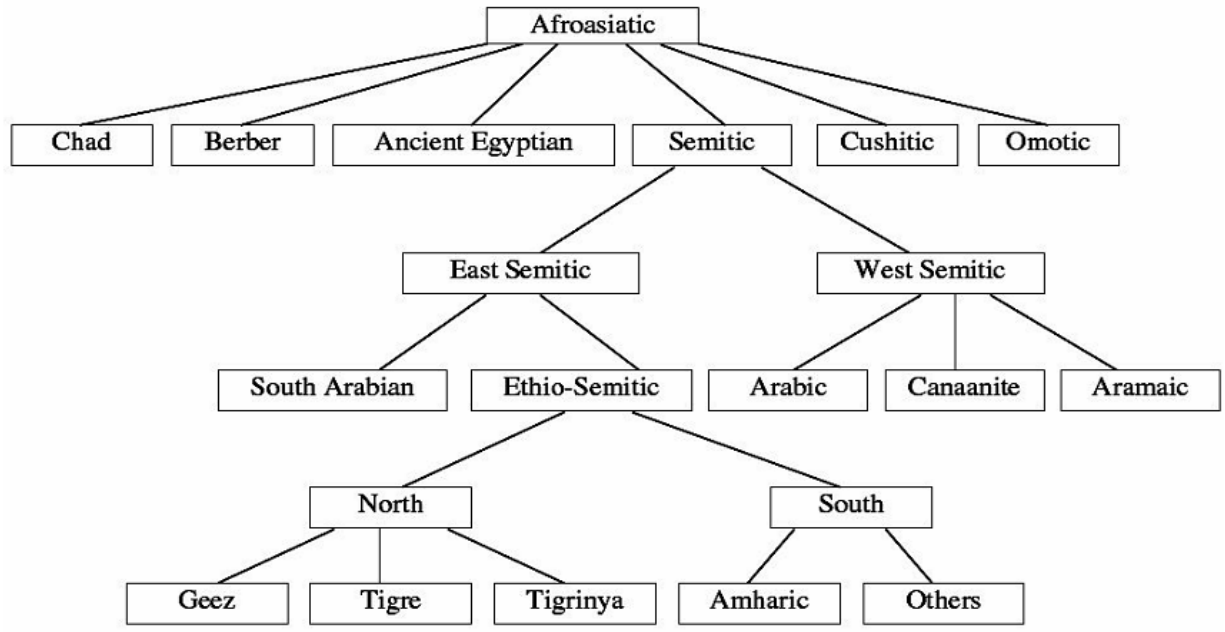


Figure 2.6: The genetic structure of Amharic language

The Ethiopic script originated from the Ge'ez alphabet around 300 A.D. and is used for writing in the various languages in Ethiopia and Eritrea, including Amharic, Tigre and Tigrigna. Ethiopic script is designed as a meaningful and graphic representation of knowledge. Ethiopic script is a component of the African knowledge systems and one of the signal contributions made by Africans to the world history and cultures. It is created to holistically symbolize and locate the cultural and historical parameters of the Ethiopian people [55].

The Ethiopian languages are divided into four major language groups, such as Cushitic, Omotic, Nilo-Saharan and Semitic. Semitic languages are spoken in northern, central and eastern Ethiopia (mainly in Tigray, Amhara, Harrari, and northern part of SNNP). The Cushitic languages are mostly spoken in central, southern and eastern Ethiopia (mainly in Afar, Oromia and Somali regions). The Omotic languages are predominantly spoken between the lakes of Southern

Rift Valley and the Omo River. The Nilo-Saharan languages are largely spoken in western parts of the country along the border with Sudan (mainly in Gambella and Benishangul - Gumuz regions) [55]. Amharic belongs to the Semitic family of languages [7] [11] [24].

2.8.2. The Amharic Language

According to Encyclopedia Aethiopica [24], it is difficult to be precise about the origin of Amharic, arising as it must have done during the centuries preceding the 13th century and in a region on the south fringes of the old Aksumite heartlands. However, some scholars such as [10] argue that the origins of the Amharic language are traced back to the 1st millennium B.C. It is rumored that they are the descendants of King Solomon and the Queen of Sheba. Immigrants from southwestern Arabia crossed the Red Sea into present-day Eritrea and mixed with the Cushitic population. New languages formed as a result of this union, e.g., Ge'ez. Ge'ez was the classical language of the Axum Empire of Northern Ethiopia. The Ge'ez or Ethiopic script was possibly developed from the Sabaean/Minean script. The earliest known inscriptions in the Ge'ez script dates back to the 5th century B.C. [35]. When the power base of Ethiopia shifted from Axum to Amhara between the 10th Century A.D. and the 12th Century A.D., the use of the Amharic language spread its influence, hence becoming the national language [10].

Aethiopica encyclopedia [24] states that, as a South Ethiopic language, Amharic does not descend directly from Ge'ez, showing as it does a small but significant number of both morphological and lexical features different from the latter, but from a presumed sister dialect or group of dialects. The fact that Amharic's closest relatives are now spread in isolated pockets from Harar in the east to Zway in the south (Argobba and the remaining East Gurage languages Silti, Ennaqor, Wallane) further suggests that these represent the fragments of what was likely to have been a more continuous band of closely related Semitic languages and dialects. The original home of Amharic is obviously to be found in the lands covered by the name Amhara. Today, Amharic is the sole language in much of Bagemedir (Gondar), western Wollo and northern Shewa and is the majority language in Gojjam.

Modern Amharic shows some dialectical variation, through perhaps less than might be expected for a language with such a wide distribution. The dialect areas that are generally recognized are geographically defined within the regions where Amharic either originated or has been spoken

the longest: Shewa, Begemeder, western Wollo and Gojjam. The dialect of Shewa and, in particular, Addis Ababa has become the prestige dialect forming a de facto standard. This is the form of Amharic that is used in the media as well as in the areas of administration and education. Some dialect features do, however, appear in written literature, even from the early days of Amharic writing. Conscious use of dialect forms and regionally restricted vocabulary occasionally also appear in novels as markers of local color [24].

2.8.3. Linguistic Features of Amharic

As stated by Bender [55] and Abreham [14], Amharic writing system took all the symbols from Ge'ez writing system and it adds some new symbols to the writing system. The Sabaean alphabet has twenty nine symbols, out of which Ge'ez took twenty four. In Ge'ez, two new symbols where created to represent sounds of Greek and Latin loan words. When Ge'ez was abandoned as the spoken language, other languages like Tigrignya and Amharic came into the society by adding new symbols in addition to the existing Ge'ez alphabets. The new symbols added in the Amharic script are ሸ(š),ዠ(z') ቸ(c) ኘ(n') ፈ(c') ቨ(v) ኸ(h£) ጀ (j). Sabaean is not used currently, where as Ge'ez is still used especially as a language in liturgy of the Ethiopian Orthodox Tewahedo and in church literature.

Unlike Arabic, Hebrew or Syrian, Amharic language is written from left to right and there is no CAPITAL or lower case distinction between characters [14] [35] [55]. Amharic writing system consists of thirty three characters (called "fidel"/ "ፊደል") as a core characters. The thirty three core characters occur in seven orders, each of which represent syllable combinations consisting of a consonant and following vowel. In addition to this, in the Amharic syllabic writing system, each character stands for a syllable rather than a single sound. Thus, the Amharic writing system is often called a syllabic rather than an alphabetic which allows anyone to write Amharic texts if s/he can speak Amharic and has knowledge of the Amharic alphabet. The non-basic forms are derived from the basic forms by more-or-less regular modifications. Other symbols representing labialization, numerals, and punctuation marks are also available. These bring the total number of characters in the script to 349. Table 2.1 shows summary of the number of characters in each group [8] [11] [14] [25] [35] [56].

Type of Amharic Characters	Total Characters
Basic characters	231
Labialized characters	89
Punctuation marks	9
Numbers	20
Total	**349**

Table 2.1: Total Number of Characters in Amharic Alphabet - ፊደል (Fidel)

In most languages, there is a small number of basic distinctions of person, number, and often gender that play a role within the grammar of the language. We see these distinctions within the basic set of independent personal pronouns, for example, English 'I', Amharic 'እኔ' 'əne'; English 'she', Amharic 'እሷ' 'əsswa'. In Amharic, as in other Semitic languages, the same distinctions appear in three other places within the grammar of the languages [57].

All Amharic verbs agree with their subjects; that is, the person, number, and (second- and third-person singular) gender of the subject of the verb are marked by suffixes or prefixes on the verb. Because the affixes that signal subject agreement vary greatly with the particular verb tense/aspect/mood, they are normally not considered to be pronouns [48].

Amharic has a further set of morphemes that are suffixed to nouns, signaling possession: ቤት "bet" 'house', ቤቴ "bete", 'my house', ቤቷ "betwa", 'her house' [55].

Amharic nouns can be primary or derived. A noun like 'እግር' "əgər" 'foot, leg' is primary, and a noun like 'እግረኛ' "əgr-äññə" 'pedestrian' is a derived noun. Amharic nouns can have a masculine or feminine gender. There are several ways to express gender. An example is the old suffix *-t* for femininity. This suffix is no longer productive and is limited to certain patterns and some isolated nouns. Nouns and adjectives ending in "-አዊ" "-awi" usually take the suffix *-t* to form the feminine form, e.g. ኢትዮጵያዊ "ityop':ya-(a)wi" 'Ethiopian (male)' vs. ኢትዮጵያዊት "ityop':ya-wi-t" 'Ethiopian (female)'; ሰማያዊ "sämay-awi" 'heavenly (male)' vs. ሰማያዊት "sämay-awi-t" 'heavenly (female)'. This suffix also occurs in nouns and adjective based on the pattern *qət(t)ul*, e.g. ንጉስ "nəgus" 'king' vs. ንግስት "nəgəs-t" 'queen' and ቅዱስ "qəddus" 'holy (male)' vs. ቅድስት "qəddəs-t" 'holy (female)' [40].

The feminine gender is not only used to indicate biological gender, but may also be used to express smallness, e.g. ቤቲቱ “bet-it-u” 'the little house' (little house-FEM-DEF). The feminine marker can also serve to express tenderness or sympathy [48].

As in other Semitic languages, Amharic verbs use a combination of prefixes and suffixes to indicate the subject, distinguishing three persons, two numbers and (in all persons except first-person and "honorific" pronouns) two genders. Adjectives are words or constructions used to qualify nouns. Adjectives in Amharic can be formed in several ways: they can be based on nominal patterns, or derived from nouns, verbs and other parts of speech. Amharic has few primary adjectives. Some examples are ደግ “dägg” 'kind, generous', ዲዳ “dəda” 'mute, dumb, silent', ቢጫ “bič'a” 'yellow' [55].

2.8.4. Morphology of Amharic

Amharic is an example of a language with a very rich morphology, which means that systems for searching Amharic text databases can be effective in operation only if full account is taken of the many word variants that may occur [11]. Like other Semitic languages, Amharic exhibits a root-pattern morphological phenomenon. A root is a set of consonants (also called radicals) which has a basic lexical meaning. A pattern consists of a set of vowels which are inserted among the consonants of a root to form a stem. In addition to this non-concatenative morphological feature, Amharic uses different affixes to create inflectional and derivational word forms [11] [55] [58].

Some adverbs can be derived from adjectives. Nouns are derived from other basic nouns, adjectives, stems, roots, and the infinitive form of a verb by affixation and intercalation. Case, number, definiteness, and gender marker affixes inflect nouns. Adjectives are derived from nouns, stems or verbal roots by adding a prefix or a suffix [58].

Moreover, adjectives can also be formed through compounding. Like nouns, adjectives are inflected for gender, number, and case. Amharic verbs are derived from roots. The conversion of a root to a basic verb stem requires both intercalation and affixation. Other verb forms are also derived from roots in a similar fashion. Verbs are inflected for person, gender, number, aspect, tense and mood. Other elements like negative markers also inflect verbs in Amharic [58].

2.9. Documents Written in Ethiopic Script

About one quarter of Ethiopia's population is literate in the written form of Amharic [23]. Accordingly, a pile of documents are published and ready for use. These documents can be categorized into typewritten, printed and handwritten.

2.9.1. Typewritten Documents

The first Amharic typewriter was made in 1950 when an Italian typewriter company called Olivetti, in cooperation with Addis Ababa College of Commerce (AACC), currently Addis Ababa University School of Commerce (AAUSC), developed an Amharic typewriter called Olivetti Lexicon 80 [32]. Since then, a number of documents are produced in the form of books, magazines, correspondence letters, etc.

Dereje [32] clearly described the characteristics of typewritten Amharic text as follows. Although height and width of the individual characters in a typewritten document are not constant, the space that is used to type a single character is proportional. This results in connected characters if two consecutive characters (especially the second, third and forth forms) take up all the space in between.

The main challenge of using Amharic typewriter is because of dust filled print heads and other scrapes of ink from the ribbon, the loop appendages of some characters and words appear as solid black circular image in most typewritten documents as depicted in Figure 2.7).

(a) (b) (c)

Figure 2.7: Scanned image of noisy typewritten words: (a) መንግሥት, (b) የተለያዩ, and (c) በመማር

From Figure 2.7, we observe that the holes of the words are filled with ink which makes matching less accurate because the feature of a clean word does not match with the noisy word image. The printed outputs from the mechanical Amharic typewriters that have been in use are

mostly of poor quality. It is very common to see touching characters, loops filled in black (caused by dust filling the hole of the print head of the machine), and broken characters in degraded background paper.

2.9.2. Printed Documents

There are a number of Amharic computer fonts available these days for Amharic document production. Some of the commonly used fonts in Amharic printed documents include 'Power Geez', 'Visual Geez', and ‘Nyala’. This shows that there is a pressing need that retrieval systems are enabled to search and read any machine printed character styles. However, since these systems rely on scripts, it is vital to understand features of each character and word such as shapes, curves, loops, directions, terminal points (up, down, left, and right) [11] [28]. To clearly understand the features of printed Amharic text, below we present Amharic characters written in different fonts.

ኢትዮጵያ ሀገራችን ለዘለዓለም ትኑር	Nyala Font
ኢትዮጵያ ሀገራችን ለዘለዓለም ትኑር	Geez-1 Font
ኢትዮጵያ ሀገሬችን ለዘለዓለም ትኑር	Geez-3 Font
ኢትዮጵያ ሀገራችን ለዘለዓለም ትኑር	Visual Geez 2000 Main Font

From the above example, we can see that words belonging to the same class but printed using different typefaces greatly vary both in shape, width, line thickness, etc. We also see that there are many possible representation of the same letter among various fonts. Some fonts produce a character big in shape or small size.

Printed documents also put a great challenge regarding indexing and retrieval of image documents. The various challenges from are divided into three major groups [11].

i. **Degradation Variations:** Printed documents are often poor in quality, in which degradations like salt and pepper, cuts, blobs and erosion, could be observed. This degradation in printed text mostly occurs due to the poor quality of paper and ink drops caused by foreign material like printers, photocopy or fax machines. The existence of large ink-blobs in the text joins disjoint words or components. There may be cuts due to folding of the paper, paper quality or foreign material. The occurrence of cuts discontinues word continuity such that one word may be split into two or more words.

ii. **Printing/Font Variations:** Printed documents are also written using various fonts, styles and sizes. As a result of which the shape and appearance of words varies substantially. For printing in a specific script, there are different fonts, sizes and styles available for use. Some of the commonly used fonts in Amharic printed documents include 'Power Geez', 'Visual Geez', 'Alphas' and 'Feedel'. Each of these fonts offer several stylistic variants such as normal, bold, and italic and font sizes, including 10, 12 and 14. These fonts, styles and sizes produce texts that greatly vary in their appearances (i.e. in size, shape, quality, etc.) in printed documents.

iii. **Word Morphology:** Each language has its own language rules, depending on which different morphological variants of a word are generated. This is briefly discussed in section 2.7.4.

2.9.3. Handwritten Documents

Handwriting was, and still is, the most dominant means of written communication. It also brings difficulty to automation of handwritten documents [8]. The handwriting aims to scribe (print) the characters in accordance with the shape of their printed counterparts. In Ethiopia, handwriting is extensively used among the society, public institutions and public officials for many purposes. Normally, there is no clear rule that abandons cursive handwriting. However, people often write in a disconnected and non-uniform manner [33]. An example of a handwritten text is shown in Figure 2.8.

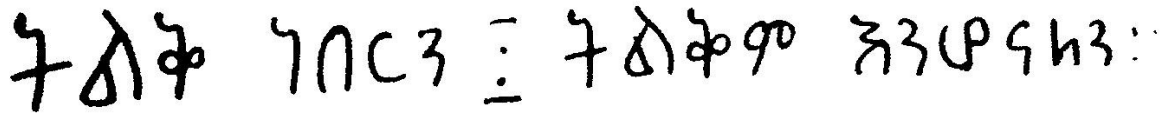


Figure 2.8: Handwritten Amharic text meaning “We were big; We will be big”.

As stated by Wondwossen [8], in handwritten documents, the major challenge is the existence and possibility of having connected characters, overlapping strokes, skewed and slant letters along with variation on the size of different as well as same characters at different instances. Furthermore, in handwriting, variations in spacing are generated from difference in writing style.

The problem is even evident for same characters written by one user at different instances. Since handwritten text lacks uniform spacing, stroke layout and alignment, preprocessing steps are very challenging.

According to Abreham [14] and Million [11], the main problems in handwritten document image retrieval are writing variations (between various writers, digitizers and writing conditions), which is not the main issue in the case of printed document image retrieval, and the large size of the vocabulary.

The present Amharic script does not serve the requirements of fast handwriting due to the following factors. Nigussie [33] lists these problems.

- Considerable amount of letters are written with many strokes, many turns and breaks.
- Some letters are not of a simple and a cursive shape.
- In many letters, one line is repeated unnecessarily.
- In few cases, the same letter is started at different "corners" depending upon the vowel that is sound with the letter. In other words, a letter might start differently in each of the seven forms.
- The same letter is "produced" differently by different people, i.e., different people start writing a letter from different "corners" and draws that letter with a variety of strokes, turns, breaks and directions.

In Ethiopia, most of historical information is written in a special handwriting style called 'Qum Tsehfet' [8]. This means of written communication used to be painted and handwritten on animal skin. The art of producing parchment and handwritten books and documents using this technique is a very significant feature of Ethiopian cultural heritage, and is as well a very old one.

The dominant writing materials for the purpose of "Qum Tsehfet" is vellum or parchment. Parchment is animal skin prepared from any domesticated mammals such as the ox, cow, calf, sheep, goat, horse and wild animals such as lion, tiger, gazelle, antelope, etc. but goat skin is more preferred due to the fact that it is a fine white one and can take ink very well. In addition to parchments, in the third and forth centuries A.D., a variety of writing materials like stone, metal, clay, and also wood were used.

Among the materials used for writing on hard materials, ink is the major one. For the case of parchments, two kinds of inks were mostly used. These are black and red, which were and still are used in Ethiopia for writing on parchments. The black ink is of a slightly different variety. Its variety colors are charcoal black and greenish black. The former is prepared particularly for magical prayer document writing and the latter on for daily uses.

Kefyalew [59] and Wondwossen [8] discusses about the major characteristics/features of Qum Tsehfet/ "ቁም ጽሀፈት" writing style. One of the challenges that is claimed to make retrieval of the special handwritten ("Qum Tsehfet") documents difficult is learning how to write using the style, designing and painting the symbols. Concerning the use of punctuations, in this special type of Amharic handwriting system, every word in a sentence is separated from the other using special word separator punctuations. These word separating punctuations are: "Nekut" or colon (፡) to separate words in a sentence, "Netela Serez" (፣) which acts as a comma in Latin scripts, "Dereb Serez" (፤) which acts as a semicolon in Latin script and "Arat Netib" (።) which acts as a sentence terminator or period in Latin script. Based on these rules, every word in this special writing system is separated from the other using one of these separators.

The writing material, the animal skin, must be preprocessed in order to make it useful. The preparation of the animal skin will help to guide the layout of the characters and the whole structure of the document [8]. This is done by drawing sequences of horizontal and vertical lines on the animal skin using sharp metal so that each character falls in one block. This feature of the writing style permits character as well as lines and words not to touch each other unless and otherwise a major noise is present.

The other peculiar feature of "Qum Tsehfet" is the weight of stroke colors. The text is either written using red or black ink or the combination of two and each stroke is written in bold format. This feature of the writing style will help to identify the information bearing strokes from the text, which is flooded by noise due to the writing material and/or the animal skin [8].

2.10. Challenges of Amharic Language, Script and Documents

Many challenges are identified by different scholars related to the Amharic language, the Ethiopic script and also the documents. The challenges in the language include the following [11] [25].

a) Existence of large number of Amharic vocabulary in the writing system is a great challenge in the development of retrieval system for the language.
b) The Amharic writing system uses multitudes of ways to denote compound words and there is no agreed upon spelling standard for compounds. These words can be written as two separate words or as a single word using 'hulet netib' (:) in between. For instance, the word "library" can be written as "ቤተመጻሕፍት" or "ቤተ፡መጻሕፍት" or "ቤተ መጻሕፍት".
c) The rule to Amharic language users generally follow in their writing system is that if the alphabet in a word sounds right when read aloud, then it is written. There are different ways of writing a single word due to various reasons such as regional dialects and various ways of writing loan words. Regional dialects have their own impact in word formation in the basic level where the words are more likely to be written by following their spoken form. For example, "ስራ" vs. "ሥራ", "ዓፄ" vs. "ዓጤ", "እነዚህ" vs. "እነኚህ", etc. The absence of restricted rules lead to a number of problems to develop efficient Amharic information retrieval system.
d) As we have seen in section 2.7.4, Amharic has a very rich morphology which forwards a challenge by itself specially during stemming.
e) In Amharic orthography, there are additional letters taken from Ge'ez that would take on the phonemic value of its nearest neighbor. The result being two syllabic series for 'se' ('ሰ' and 'ሠ'), two series for 'tse' ('ጸ' and 'ፀ'), two for 'a' ('አ' and 'ዐ') and four for 'ha' ('ሀ', 'ሐ', 'ኀ' and 'ኸ'). These redundancies in Amharic become a source of confusion and the letters are treated as interchangeable by a person. Common Amharic spelling then becomes highly flexible and correctness is not a matter of precision but one of acceptable proximity. For example, the word "sun" can be written as "ፀሀይ", "ፀሃይ", "ፀሐይ", "ፀኀይ", "ጸሀይ", "ጸሃይ", "ጸሐይ", "ጸኀይ", etc. all mean the same, although they are written differently.

f) It is common to write some words in shorter form by using '/' (forward slash) or '.' (dot). The short form of words which are linked by either of these symbols can be expanded as single or a combination of words. For instance, "አ.አ." is expanded as combination of two words አዲስ አበባ (means "Addis Ababa"). "መ/ር" is a short form of the single word መምህር (means teacher) or another word መዝሙር (means song).

g) Amharic loan (foreign) words can be written in different ways. For instance, the word "computer" can be written as "ኮምፒዩተር", "ኮምፒውተር", "ኮምፒዉተር", "ኮምፒተር", "ኮምፕዩተር", etc.

2.11. Related Research Works

Due to limitations of OCR systems a number of attempts are made by researchers to avoid the use of character recognition for various document image retrieval applications. Consequently, the research direction towards retrieving information without explicit recognition from document images considers printed or handwritten documents which are encoded in different languages. In this study, we attempt to review global and local researches that are done to search in document images.

2.11.1. Global Research Works

A research that aims to facilitate information retrieval from low quality historical documents by spotting different instances of a given query word in the text was conducted by Khurshid [45]. The overall retrieval system has been divided into two major parts: indexing and searching. The first step is the preprocessing stage involving an optimized document binarization algorithm NICK to crisply distinguish the foreground from the background. Then, a word/graphic segmentation is performed using a combination of horizontal smoothing and connected component area analysis. Nevertheless, the quality of the images does not permit a word segmentation free of errors.

For word spotting, the researcher proposed relative position correspondence (RPC), edit distance, merge-split edit distance and linear displacement matching methods. Major problem was to define a word matching method which not sensitive to segmentation errors caused by the low quality of the document images. From the results analysis, the best matching rate has been

achieved by the merge-split edit distance method which correctly detected 427 query word instances along with other 39 relevant words and 4 false positive. Next to that, linear displacement matching method detects one less false positive but is unable to detect 15 query instances, 7 more than the merge-split method. Merge-split edit distance and the linear displacement matching methods outperform others because of their capability to overcome segmentation problems. When tested on very old poor document images, merge-split edit distance achieved good results (422 perfectly detected words and 13 missed). The researcher recommended the integration of skew correction, better segmentation and feature extraction techniques.

Ataer and Duygulu [60], proposed a novel approach for searching Ottoman documents based on word matching. Ottoman script is a connected script, which is difficult to segment and recognize. The researchers used printed type of Ottoman scripts, which is more regular and justified than the other styles. These scanned documents are passed from a binarization process. Since the experiments are carried out on printed, relatively clean documents, a binarization step based on simple thresholding can produce acceptable results for further processing.

The proposed method is composed of two main stages: segmentation and matching. first successfully segments the documents into lines and then into words. Line extraction is performed by finding the baselines using horizontal projection profiles and then by specifying the upper and lower limits for the characters relative to the baselines. Then, words are extracted using the vertical projection profiles and then uses a hierarchical matching technique to find the similar instances of the word images. The proposed line extraction method achieved 100% accuracy and the word extraction works with 82% accuracy. The performance dropped because of the different noise levels in documents. In the matching stage, segmented words are queried and retrieval is performed with the use of four distinctive features: word length, quantized vertical projection profile and quantized vertical projection profiles of ascenders and descenders. These four features are used in four consecutive tests and each test discards the dissimilar word images for the next test. An average mean precision value of 85.24% is achieved for all the words in the data set. To improve the retrieving performance of the system, the researchers recommended the improvement of set of features by including some shape features and compute the occurrence of each word image in the dataset.

Kokare and Shirdhonkar [21] provide detailed survey and technical achievements of document image retrieval including methods developed, applications, challenges and future directions. The various steps involved in document image retrieval i.e. noise removal, feature extraction, and matching algorithm are shortly discussed. Noise removal is carried out to get rid of any noise introduced due to excessive dusty noise, large ink-blobs joining disjoint characters or components, the poor quality of paper and ink, text overlapping the signature. Image enhancement procedures can be applied for this purpose. To extract the features, methods such as Gradient, Structural and Concavity (GSC) features and others are identified by the researchers. Then, the document image retrieval is performed using a matching algorithm, for example DTW, to compare the query image with image database.

The survey includes papers covering the current state of art on the research in DIR retrieval based on images such as signature, logo, machine-print, different fonts etc. Among the various applications of DIR recognized by the researchers, word searching, document similarity measurement, DIR using signature as queries, automatic document logo detection and retrieving imaged documents in digital libraries are the major ones.

According to them, the great challenge in the field of document recognition and retrieval is search and retrieval from large collection of document images. To implement a successful search engine in image domain, issues such as computational speed, degradation of documents, appropriate representation of scheme and matching algorithms and cross-lingual retrieval should be addressed. A strategy for evaluation involves determining an appropriate database for evaluation and an appropriate metric and criteria for evaluating competing approaches such as precision and recall.

2.11.2. Local Research Works

An innovative research on document image retrieval without explicit recognition for Amharic language in addition to English and Indian languages was conducted by Million [11]. The motivations were the lack of robust OCR system for Amharic language and designing OCR is also a long term process for retrieving information from document images. The study proposed, on the other hand, an effective word image matching called dynamic time warping (DTW), on the other hand, combining features of transition profile, lower profile with order moment and

upper profile scheme that achieves high performance in presence of script variability, printing variations, degradations and word-form variations. Datasets containing a total of more than 800, 000 word images in English, Hindi and Amharic languages were prepared to conduct extensive experiments. The dataset contains basic words with their morphological variants generated using degradation models such as salt and pepper, cuts, blobs and erosion of pixels which are printed using various fonts, styles and sizes.

Test result on degraded text images shows that performance varies depending on the degradation type. On average, 92.52%, 95.49%, 89.51% and 93.61% F-scores are obtained on cuts, salt and pepper, blobs and erosion, respectively. The average performance of the proposed scheme on the various fonts, styles and sizes is 91.62% F-score. The researcher clearly stated the need to apply advanced image preprocessing techniques (noise removal and skew correction) for document analysis since printed document images are more of historical and poor in quality and document image processing algorithms for document image collections are missing.

Mesfin [35] also attempt to design a retrieval system that can search for relevant document images from scanned Amharic document image corpus by accepting a query from the user depending on image features only. Amharic document images were scanned in grayscale with 300 dpi intensity and fixed threshold method was used. Two phased segmentation algorithm i.e. line segmentation followed by word segmentation was implemented. Line and words in a document are identified using horizontal and vertical boundary segmentation.

Experiment is carried out on 121 scanned Amharic documents containing 483 pages and 109,238 words that are selected from printed legal documents and news items, among which 28 word queries were selected for testing. All documents in the dataset have a font size of 12, a font type of Power geez Unicode 1 and a plain style. One of the limitations of this research is, it does not consider ancient and handwritten documents. The retrieval effectiveness of the system was measured using retrieval measures such as precision, recall and F-measure. Based on performance analysis, the highest average F-measure of 57.08% was achieved using parallel bar feature extraction method and Euclidean similarity measure.

In order to increase the performance of the proposed retrieval system, preprocessing techniques like noise removal, font resizing and font conversion are recommended. In addition to this, the need for efficient indexing structure and searching using multiple word queries is stressed.

Based on Mesfin's [35] recommendation, Abreham [14] attempted to design an effective and efficient document retrieval system that searches relevant documents from Amharic image corpus and displays documents in ranked order as per users' query.

To extract features, word shape analysis of vertical bar pattern is performed. The extracted feature uniquely identifies each of the word in the document collection. To measure the similarity between images, Cosine similarity measurement was used which was also used to detect suffix and prefix of word images. The suffix and prefix detection at the time of comparing two image terms shows 85.3% average accuracy. Inverse document frequency (IDF) was computed to remove common word images or stop words which occurred in more than 80% of the documents.

An inverted index structure was used to build the index file, which contains the index term, cumulative frequency, term frequency and document frequency of each word image. Documents were ranked based on their Term frequency and inverse document frequency (TF*IDF) weight.

The efficiency and effectiveness of the system is tested using test cases. The efficiency is checked by measuring the time taken to search relevant documents from the index file. Further effectiveness of the system is measured using recall, precision and F-measure. According to the result, effectiveness of the system shows F-measure value of 41.59%. The efficiency, which was measured using the average search time required before and after indexing, showed improvement by more than 26.6%.

The performance of the system is greatly affected by noise that exist within real-life document images. Therefore, pre-processing techniques like noise removal, skew correction and normalization are recommended to be integrated to increase the effectiveness and efficiency of the system. Furthermore, analysis of better feature extraction and matching techniques and algorithms were also recommended.

Continuing Abreham's [14] work, Adane [19] endeavored to design and integrate effective and efficient feature extraction and matching schemes which are insensitive to the artifacts in real life word images, such as noise, word variations and difference due to font sizes, styles and types in order to enhance the performance of Amharic document image retrieval system.

Primarily, computer printed Amharic documents with various font styles, sizes and types are collected from books, magazines, and newspapers which are then converted to gray scale digital images using a flatbed scanner at 300 DPI. The document images are then binarized into digital and manageable representations. This is followed by line and word segmentation respectively. Then, features are extracted at word level using combination of three feature extraction techniques Vertical distance, Vertical projection and Lower word profile.

Based on DTW matching algorithm, the best performance of 95.52% is registered. DTW achieved good matching performance on low and medium level noisy document images. The selected matching, feature extraction and stemming techniques are integrated to the previous Amharic document image retrieval system and tested on noisy real life document images which are representative of low, medium and high level noisy document images. Accordingly, the maximum F-measure of 80.46% is achieved on low level noisy document images. On the other hand, 66.66% and 55.82% F-measures are registered on medium and high level noisy document images, respectively. The overall performance of the currently available Amharic DIRS is 67.64% F-measure.

Analysis of the performance of the system shows that it is significantly affected by the existence of noise and different levels of degradations in real-life document images. Hence, integrating noise reduction technique improves the performance of the document image retrieval system. The researcher highly stressed the need for the development and integration of noise detection and removal tool and skew detection technique that enhances effectiveness of the Amharic document image retrieval system.

The recent research on Amharic DIRS by Tilahun [25] attempted to develop Amharic Document Image Retrieval System using Natural Language Processing (NLP) concepts without explicit recognition. The proposed document images retrieval techniques depend heavily on the

performance of the Amharic words synthesizer and also presents a new approach for indexing document images.

The main contributions of this study includes integration of Amharic words synthesizer to tackle problems in index file creation and query processing for morphological rich languages like Amharic, handling variations in font types, sizes and styles by resizing to candidate word size and it also enables users searching using multiple words which in turn increases the system performance.

The dataset contains document images in printout format from magazines, newspapers and books written in different font type (VG 2000 main, Visual Geez Unicode, Geez-1), style (normal, bold), size (8, 10, 12, 16, 20) and image qualities (clean and noisy). A total of 32,020 template word images, most of it was built using the Amharic word synthesizer, are presented in the lexicon for indexing and searching purposes. The system was tested on those Amharic words database and scanned documents. Based on the experiment conducted, lower word profile registered the highest F-measure (92.5%) followed by vertical projection (82.9%) and upper bound profile (76.9%).

Taking into account the low quality of the documents considered in this experiment, the researcher states that the proposed approach is very promising for historical document image indexing and retrieval if advanced noise removal and image restoration techniques are applied. Moreover, the researcher also recommended integrating a tool that can produce synonyms of a given Amharic word, computational linguistic tool such as Part-Of-Speech tagger for Amharic to be used, application of documents with diverse contents and removal of punctuation marks.

Based on the recommendation of previous researchers [14] [19] [25] [35], the focus of this research is mainly integrating noise detection and removal, and thresholding tools as well as multiple query searching to the previous attempts to develop Amharic DIR system that works in real-life document images.

CHAPTER THREE

IMAGE PREPROCESSING TECHNIQUES AND ALGORITHMS

Preprocessing is one of the main tasks that prepare a document image for efficient retrieval by cleaning noise, correcting the slant and converting the color or gray level image to binary. Previously developed Amharic document image retrieval systems (ADIRS) by [14] [19] [35] lack noise removal and use a fixed thresholding system. In this research, image preprocessing techniques are explored and the best noise reduction and thresholding algorithms are integrated to enhance the performance of the Amharic document image retrieval system.

3.1. Architecture of the Amharic Document Image Retrieval System

The proposed architecture for the ADIRS is shown in Figure 3.1. Two main processes are performed: offline and online. The offline process is the indexing process that starts by scanning real life documents in order to come up with a digitized document images. Then, this document images are preprocessed. The main image preprocessing tasks are noise detection and removal and binarization/thresholding. After preprocessing, the document images are segmented into words and their feature values are extracted. This feature values are finally stored in index file using inverted file indexing structure.

Searching is the online process where a user enters query words to retrieve relevant documents that fulfils his/her information need. The multiple input query words are rendered to convert the text into an image. This multiple query words are split and each word is rendered as an image. Then, query image preprocessing mainly binarization is performed using a fixed threshold. Since the query is rendered as black foreground and white background, a fixed threshold does the trick of converting the gray scale image to binary image. Noise filtering task is not applied on the query image since it does not contain noise. Features are extracted from the processed query image and similarity measure is calculated to search for and retrieve documents that match or contain the query word. The similarity score calculated for each query word is collected and ranked using vector space IR model. Finally, the ranked list of retrieved documents are displayed back to the user.

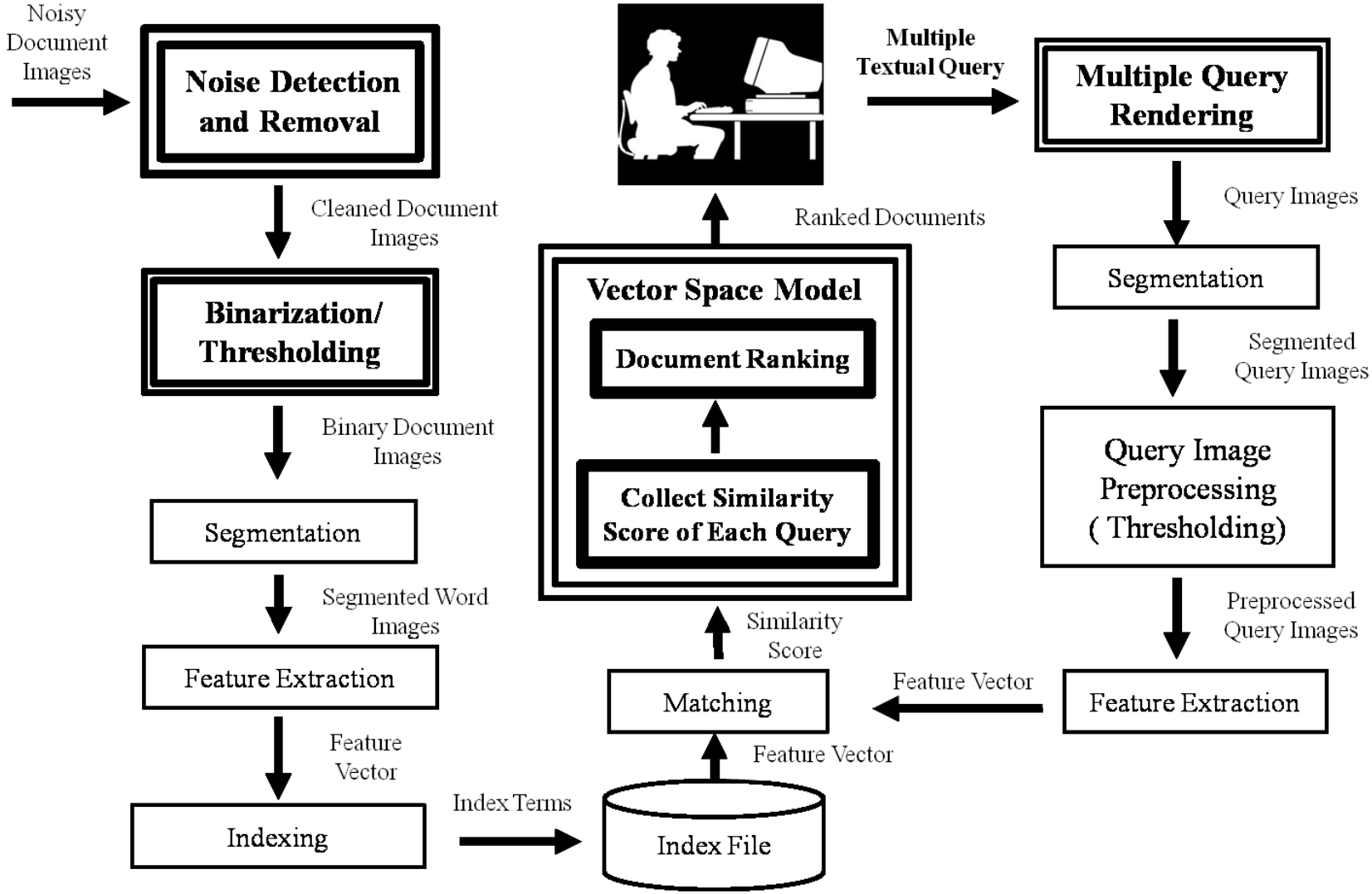


Figure 3.1: Architecture of the proposed Amharic Document Image Retrieval System (ADIRS)

The focus of this research is represented by double rectangle and **bold**.

The effectiveness of previously developed Amharic document image retrieval system (ADIRS) by [14] [19] [35] is highly affected by noise prevalent in real world documents. This was because no image preprocessing module especially noise detection and removal were integrated except fixed thresholding method of binarization. Therefore, image preprocessing module must be integrated to the system to enhance retrieval performance. Since this research work is an extension of Mesfin [35], Abreham [14] and Adane [19], some functions are taken from their work and different image preprocessing techniques are developed and integrated.

3.2. Noise Filtering Techniques

In this study, three noise filtering techniques (one linear and two nonlinear) are explored to see their effect on Amharic document images.

3.2.1. Median Filtering

Compared to other non-linear filters, median filtering is both simple and efficient. Just like a linear low-pass filtering, it smoothes the image and can therefore eliminate certain of the image's imperfections. However, unlike a linear low-pass filter, which inevitably adds a blur around contours, it better preserves the sharp variations of the image [61].

Definition of Median filter as stated in [61] is presented below. Let {I(i, j)} be an image. The median filter associates the mean value m(k, l) with the point with coordinates (i, j), in the (M × N) rectangular window, created on (i, j). If we assume M and N to be odd, and if u(n) denotes the sorted sequence (u(n) ≥ u(n-1)) obtained from the array [I(i, j)] where $i \in \{(k - (M-1)/2), \dots, k + (M-1)/2\}$ and $j \in \{(l - (N-1)/2), \dots, l + (N-1)/2\}$, we have:

$$m(k,l) = u\big((MN+1)/2\big)p_K \qquad (3.1)$$

In median filtering, the input pixel is replaced by the median of the pixels contained in the neighborhood [62]. Symbolically, this can be represented as:

$$u(m,n) = median\{y(m-k, n-1), (k,1) \in W\} \qquad (3.2)$$

where W is suitably chosen neighborhood. The algorithm for median filtering requires arranging the pixel gray values in the neighborhood in increasing or decreasing order and picking up the

value at the center of the array. Generally the size of the neighborhood is chosen as odd number so that a well-defined center value exists. If, however, the size of the neighborhood is even the median is taken as the arithmetic mean of the two values at the center.

According to Acharya and Ray [62], some of the properties of median filter are the following. First, it is a nonlinear filter. Second, it is useful in removing isolated lines or pixels while preserving spatial resolution. It is found that median filter works well on binary noise but not so well when the noise is gaussian. Finally, its performance is poor when the number of noise pixels is greater than or equal to half the number of pixels in the neighborhood.

The median filter deals with each pixel and assures it fits with the pixels around it. Therefore, it is very useful in filtering out missing or damaged pixels. It is especially effective for pictures with salt and pepper noises, which are often results of electronic noise during transmission. Because of the sheer volume of data that normally needs to be filtered, the main problem in designing the median filter is efficiency and time consumption.

Median filtering is done by replacing the value of each element by the median found in a window around the element. Thus, the median will in general replace a noisy value with one closer to its surroundings. An algorithm for a simple 2D median filter is presented in algorithm 3.1.

3.2.2. Wiener Filter

The wiener filter is a filter proposed by a scholar named Norbert Wiener. Wiener is a type of linear filter that filters an image adaptively, tailoring itself to the local image variance. This approach often produces better results than linear filtering. The adaptive filter is more selective than a comparable linear filter, preserving edges and other high-frequency parts of an image [63].

Wiener filter is a discrete time linear finite impulse response (FIR) filter [62]. Wiener filter applied locally on each frame of a sequence allows to obtain good results in terms of edges [63].

Given a degraded image I' of some original image I and a restored version R, the measure we use to say whether the restoration was a good job or not is if R is as close as possible to the "correct" image, I. One way of measuring the closeness of R to I is by adding the squares of all differences.

Algorithm 3.1: Algorithm for 2-Dimentional Median Filter

```
allocate outputPixelValue[image width][image height]
  edgex := (window width / 2) rounded down
  edgey := (window height / 2) rounded down
  for x from edgex to image width - edgex
    for y from edgey to image height - edgey
      allocate colorArray[window width][window height]
      for fx from 0 to window width
        for fy from 0 to window height
          colorArray[fx][fy] := inputPixelValue[x + fx - edgex][y + fy - edgey]
      sort all entries in colorArray[][]
      outputPixelValue[x][y] := colorArray[window width / 2][window height / 2]
```

This has been widely used in reconstruction of one-dimensional signals and two-dimensional images. Although Wiener filter is sensitive to noise, yet it can be used for good restoration of the original image. The elegance of Wiener filter lies in the fact that it incorporates the prior knowledge about the noise embedded in the signal and also the spectral density of the object being imaged. As a result, Wiener filter provides a better and improved restoration of original signal since it takes care of the noise process involved in the filtering [62].

Wiener filter estimates the local mean and variance around each pixel.

$$\mu = \frac{1}{MN} \sum_{i,j \in n} I(i,j) \qquad (3.3)$$

and

$$\sigma^2 = \frac{1}{MN} \sum_{i,j \in n} I^2(i,j) - \mu^2 \qquad (3.4)$$

where n is the M-by-N local neighborhood of each pixel in the image I. Wiener filter then creates a pixel-wise filter using these estimates,

$$I'(i,j) = \mu + \frac{\sigma^2 - v^2}{\sigma^2} (I(i,j) - \mu) \qquad (3.5)$$

where v^2 is the noise variance.

3.2.3. Adaptive Median Filtering (AMF)

The adaptive median filtering (AMF) has been applied widely as an advanced method compared with standard median filtering [41]. The AMF performs spatial processing to determine which pixels in an image have been affected by impulse noise. The AMF classifies pixels as noise by comparing each pixel in the image to its surrounding neighbor pixels. The size of the neighborhood is adjustable, as well as the threshold for the comparison. A pixel that is different from a majority of its neighbors, as well as being not structurally aligned with those pixels to which it is similar, is labeled as impulse noise. These noise pixels are then replaced by the median pixel value of the pixels in the neighborhood that have passed the noise labeling test. The algorithm works by changing the size of S_{ij} (the size of the neighborhood) during operation [42].

Given: S_{ij} = processing window size, Z_{min} = minimum intensity value in S_{ij}, Z_{max} = maximum intensity value in S_{ij}, Z_{med} = median of intensity value in S_{ij}, Z_{xy} = intensity value in at coordinates (i, j), and S_{max} = maximum allowed size of S_{ij}.

The adaptive median-filtering algorithm works in two stages or levels, denoted as level A and level B. This is explained in algorithm 3.2.

Algorithm 3.2: Algorithm for Adaptive Median Filter

Level A: IF Zmin < Zmed < Zmax, then

// Zmed is not an impulse

(1) go to level B to test if Z_{ij} is an impulse

ELSE

// Zmed is an impulse

(1) the size of the window is increased and

(2) level A is repeated until

(a) Zmed is not an impulse and go to level B or

(b) Smax reached: output is Z_{ij}

Level B: IF Zmin < Z_{ij} < Zmax, then

// Z_{ij} is not an impulse

(1) output is Z_{ij} (distortion reduced)

ELSE

// either Z_{ij} = Zmin or Z_{ij} = Zmax

(2) output is Zmed (standard median filter)

// Zmed is not an impulse (from level A)

The key to understanding the mechanics of this algorithm (algorithm 3.2) is to keep in mind that it has three main purposes: to remove salt-and-pepper (impulse) noise, to provide smoothing of other noise that may not be impulsive, and to reduce distortion, such as excessive thinning or

thickening of object boundaries. The values Z_{min} and Z_{max} are considered statistically by the algorithm to be "impulse-like" noise components, even if these are not the lowest and highest possible pixel values in the image.

3.3. Thresholding Techniques

In thresholding, the gray-scale image is reduced or converted to a binary image. For a thresholding algorithm to be really effective, it should preserve logical and semantic content [46].

As stated by [44], thresholding an image is a special type of quantization that separates the pixel values in two classes, depending upon a given threshold value $I_{th.}$ The threshold function $f_{threshold}(I)$ maps all pixels to one of two fixed intensity values I_0 or I_1; i.e.,

$$f_{threshold}(I) = \begin{cases} I_0 \; for \; I < I_{th} \\ I_1 \; for \; I \geq I_{th} \end{cases} \qquad (3.6)$$

with $0 < I_{th} < ft_{max}$. A common application is binarizing an intensity image with the values $I_o = 0$ and $I_1 = 1$.

In this research, both global and local thresholding techniques are experimented, that is, Otsu from global thresholding and Niblack and Sauvola from local thresholding.

3.3.1. Otsu Global Thresholding Algorithm

In global thresholding, a single threshold for all the image pixels is used. When the pixel values of the components and that of background are fairly consistent in their respective values over the entire image, global thresholding could be used. In adaptive thresholding, different threshold values for different local areas are used [46].

One of the earliest and famous thresholding algorithms was suggested by Otsu [64], which is based on the principle that the gray-level for which the inter-class variance is maximum is selected as the threshold.

Otsu's thresholding method [64] is a global binarization method which involves iterating through all the possible threshold values and calculating a measure of spread for the pixel levels on each side of the threshold, i.e. the pixels that either treated as foreground or background. The aim is to

find the threshold value where the sum of foreground and background spreads is at its minimum. This method gives satisfactory results when the numbers of pixels in each class are close to each other. The Otsu method still remains the reference for comparing different thresholding and binarization methods in general [39].

For a gray level k all the gray-value $\leq$ k will form a class (C_0) and all the others will form a different class (C_1). Select that k as threshold for which the between class variance $V(k)$ is maximum. The criterion proposed by Otsu maximizes the between-class variance of pixel intensity. This method involves in more computational complexity of the between-class variance [62]. The step-by-step formulas or algorithm for Otsu thresholding [64] is presented in algorithm 3.3.

3.3.2. Niblack's Local Thresholding Algorithm

While global methods apply one threshold to the entire image, local thresholding methods apply different threshold values to different regions of the image. The value is determined by the neighborhood of the pixel to which the thresholding is being applied [65].

One of the most popular local threshold binarization methods is proposed by W. Niblack [66]. This method involves calculating for each image pixel the mean and the standard deviation of the gray level value of the neighboring pixels that are found in a window of a predefined size. This size influences the quality of the output and it is recommended to be small enough to conserve local details and large enough to suppress noise [67] [68].

Niblack's algorithm [66] calculates a pixel-wise threshold by sliding a rectangular window over the gray-level image. The threshold T is computed by using the mean $m(x, y)$ and standard deviation $s(x, y)$, of all the pixels in the window, and is denoted as:

$$T(x,y) = m(x,y) + w \times s(x,y) \qquad (3.7)$$

where w is a constant, which determines how much of the total print object edge is retained, and has a value between 0 and 1. The value of w and the size of the sliding window (SW) define the quality of binarization. Niblack's thresholding algorithm gives thick and unclear strokes with a

small *w* value, and slim and broken strokes with a large *w* value, while with a small *SW* value noise is closer to texture.

Algorithm 3.3: Algorithm for Otsu's Thresholding Method

1. The pixels are divided into two classes:

   C1 with gray level [1 ... t] and

   C2 with gray level [t+1 ... L]

2. The probability distribution for the two classes is:

   2.1. Define the within-class variance

$$C1 = \frac{P_1}{W_1(t)}, \dots, \frac{P_t}{W_1(t)} \text{ and}$$

$$C2 = \frac{P_{t+1}}{W_2(t)}, \dots, \frac{P_L}{W_2(t)}$$

$$where, W_1(t) = \sum_{i=1}^{t} P_i \; and \quad W_1(t) = \sum_{i=t+1}^{L} P_i$$

   The mean of the two classes are:

$$\mu_1 = \sum_{i=1}^{t} \frac{iP_i}{W_1(t)} \; and \; \mu_2 = \sum_{i=t+1}^{L} \frac{iP_i}{W_2(t)}$$

   2.2. Define the between-class variance

   Using discriminant analysis, Otsu defined the between-class variance of the threshold image as:

$$\sigma^2_{Between} = W_1(\mu_1 - \mu_t)^2 + W_2(\mu_1 - \mu_t)^2 \qquad (3.8)$$

Advantage of Niblack's method is that it always identifies the text regions correctly as foreground but on the other hand tends to produce a large amount of binarization noise in non-text regions and text boundaries [45].

3.3.3. Sauvola's Local Thresholding Algorithm

Sauvola's algorithm [15] is a modification of Niblack's which is claimed to give improved performance on documents in which the background contains light texture, big variations and uneven illumination. In this algorithm, a threshold is computed with the dynamic range of the standard deviation, *R*, using the equation:

$$T(x,y) = m(x,y) * \left(1 + k * \frac{s(x,y)}{R} - 1\right) \qquad (3.9)$$

where *m(x, y)* and *s(x, y)* are the mean and standard deviation of the whole window and *k* is a fixed value. An optimal combination of *k* and window size (SW) will produce a good binary image.

3.4. Vector Space Model

Ricardo and Ribeiro-Neto [13] clearly presented the definition and equation of vector space model. The vector space model is accomplished by assigning non-binary weights to index terms in queries and in documents. These term weights are ultimately used to compute the degree of similarity between each document stored in the system and the user query. By sorting the retrieved documents in decreasing order of this degree of similarity, the vector model takes into consideration documents which match the query terms only partially. The main resultant effect is that the ranked document answer set is a lot more precise (in the sense that it better matches the user information need) than the document answer set retrieved by the Boolean model.

Definition:

For the vector model, the weight $w_{i,j}$ *associated with a pair (*k_i *,* d_j*) is positive and non-binary. Further, the index terms in the query are also weighted. Let* $w_{i,q}$ *be the weight associated with the pair [ki , q], where* $w_{i,q} \geq 0$*. Then, the query vector* $\vec{q}$ *is defined as* $\vec{q} = (w_{1,q}, w_{2,q}, \ldots, w_{t,q})$

where t is the total number of index terms in the system. As before, the vector for a document dj is represented by $\vec{d} = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$.

Therefore, a document *dj* and a user query *q* are represented as *t-dimensional* vectors as shown in Figure 3.2. The vector space model proposes to evaluate the degree of similarity of the document *dj* with regard to the query *q* as the correlation between the vectors $\overrightarrow{d_j}$ and $\vec{q}$. This correlation can be quantified, for instance, by the cosine of the angle between these two vectors. That is,

$$sim(d_j, q) = \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \times |\overrightarrow{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \qquad (3.10)$$

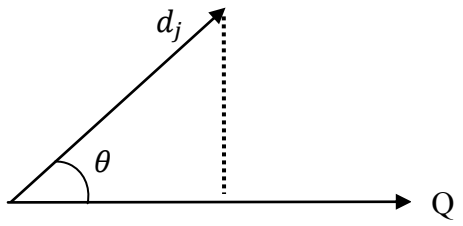


Figure 3.2: The cosine of θ is adopted as *sim(dj,q)*

where $|\vec{d}_j|$ and $|\overrightarrow{q}|$ are the norms of the document and query vectors. The factor $|\overrightarrow{q}|$ does not affect the ranking (i.e., the ordering of the documents) because it is the same for all documents. The factor $|\vec{d}_j|$ provides a normalization in the space of the documents. Since $w_{i,j} \geq 0$ and $w_{i,q} \geq 0$, *sim(q, dj)* varies from 0 to +1. Thus, instead of attempting to predict whether a document is relevant or not, the vector space model ranks the documents according to their degree of similarity to the query.

3.5. Performance Measures

The main goals of this research are to integrate image preprocessing method and adopt IR model to the already developed Amharic DIRS. Therefore, to quantitatively assess the strength and quality of the restored images and to judge the performance of image denoising and thresholding techniques, Mean Squared Error (MSE) and Peak Signal to Noise Ratio (PSNR) are used. According to [41], these measurements are the automatic choice for researchers.

The Mean Square Error (MSE) and the Peak Signal to Noise Ratio (PSNR) are the two error metrics used to compare image compression quality [38]. The MSE is the cumulative squared error between the denoised image and the original image (see equation 3.10) [39].

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[X(i,j) - Y(i,j)]^2 \qquad (3.11)$$

Peak Signal to Noise Ratio (PSNR) is the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation [40].

Saba et. al. [57] defined PSNR as the ration of the variance of the noise-free signal to the mean-squared error between the noise-free signal and the denoising signal.

PSNR is a measure of the quality of reconstruction; a higher PSNR would normally indicate that the reconstruction is of higher quality. PSNR is usually expressed in terms of the logarithmic decibel scale, 'dB' for short. PSNR a transformation of mean square error (MSE) and is computed as shown in equation 3.11 [38] [39] [57].

$$PSNR = 10 * \log_{10}\left(\frac{R^2}{MSE}\right) \qquad (3.12)$$

In the previous equation, *R* is the maximum fluctuation in the input image data type. For example, if the input image has a double-precision floating-point data type, then *R* is *1*. If it has an 8-bit unsigned integer data type, *R* is *255*, etc [38].

An information retrieval system returns relevant documents that satisfies the information need of users' query. In order to present a set of ranked documents, the performance in terms of effectiveness and efficiency should be measured. In this Amharic DIRS, we measure the effectiveness of the system before and after integrating image preprocessing module to the existing system. According to Manning et. al. [12], the two most frequent and basic measures for information retrieval effectiveness are precision and recall.

While Precision is the fraction of retrieved documents that are relevant, Recall is the fraction of relevant documents that is retrieved from the total number of relevant document in the collection [13]. Delalandre [69] presents a clearly understandable equation of precision and recall as follow.

$$Precision = \frac{|\{relavant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|} \quad (3.13)$$

$$Recall = \frac{|\{relavant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|} \quad (3.14)$$

The advantage of having the precision and recall value is that one is more important than the other in many circumstances. Recall is a non-decreasing function of the number of documents retrieved. On the other hand, in a good system, precision usually decreases as the number of documents retrieved increase. In general, we want to get some amount of recall while tolerating only a certain percentage of false positives [69].

A single measure that trades-off precision versus recall is the F-measure, which is the weighted harmonic mean of precision and recall [19]:

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (3.15)$$

Using the various filtering and thresholding techniques, an extensive experimentation is made in this study in order to select the best combination of algorithms working for Amharic real-life document images.

CHAPTER FOUR

EXPERIMENTATION AND DISCUSSION

This study undertake experiments to integrate image preprocessing module to the Amharic Document Image Retrieval System (ADIRS) developed by previous researchers [14] [19] [35].

The proposed architecture (as shown in Figure 3.1) starts by converting the real-life documents to a digital form using scanner. Once the documents are digitized, the main objective of this research work i.e. image preprocessing tasks such as image denoising and binarization are carried out. Then, line and word segmentation module developed by Mesfin [35] and feature extraction module by Adane [19] are performed. After extracting features from the documents, the next step is to identify index terms and store them in a file. This is accomplished by the module developed by Abreham [14].

When a user enters a query word, it is rendered in order to convert the text to an image. The query does not contain any noise since it has been rendered from an input text. The text is written in black foreground and white background. Therefore, a fixed thresholding algorithm can do the trick of binarizing the rendered image. Thresholding is done as if the intensity of a pixel at (i, j) is greater than 128, then one *(1)* is assigned, else zero *(0)* is assigned. Then, segmentation and feature extraction is accomplished. Finally, the features of the query image is matched with the features of index terms in the index file and documents that are similar (i.e. scored higher performance) are returned to the user in ranked order.

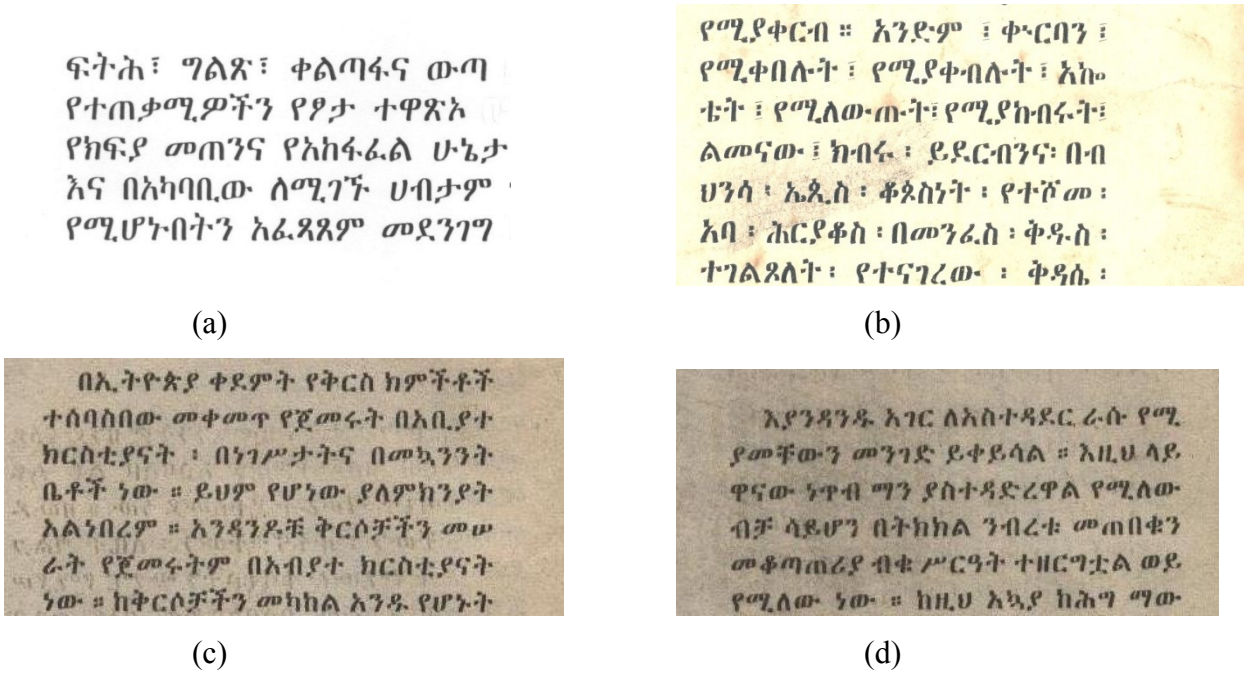
4.1. Dataset Preparation

In this research, different printed real-life documents from book called 'Metsehafe Qidase', an old magazine named 'Meskerem' printed in 1989 G.C., a regulation document from Oromia auditor office and different newspapers are scanned. A flatbed scanner is used with a resolution of 300 depth-per-inch (DPI). 300 DPI is the preferred and efficient level for real-life documents because it does not tend to break thin lines or fill gaps [30].

To evaluate the performance of the system, 4,974 word images are collected from real-life books, magazines, newspapers, and regulations with varying font styles (plain, **bold**, *italic*), types

(Power Ge'ez, Visual Ge'ez, Nyala, etc), size (10, 12, 14). Based on the level of noise prevalent in the document images and the pixel intensity, we classified them into low level, medium level, high level and very high level using the following criteria, as depicted in Figure 4.1.

- **Criteria 1:** If a document image have less intensity and a little background noise behind words, then it is *low level*.
- **Criteria 2:** If a document image affected by blob noise that connects different words together, blurring and higher background noise than low levels, it class is *medium level*.
- **Criteria 3:** If a document image characterized by considerable background and show-through noise from the back of the paper, it is classified as *high noise level*.
- **Criteria 4:** If a document image is plagued by blurring, aging and much more background and show-through noise, and blob noise than all other levels of noise, then it is *very high level* noisy document image.



(a) (b)

(c) (d)

Figure 4.1: Noisy document images

(a) low level noisy document from Oromia Auditor Office, (b) medium level noisy document from 'Metsehafe Qidase', (c) high level document from 'Meskerem' magazine, and (d) very high level document from 'Meskerem' magazine.

The number of word images collected from different sources and their category is summarized in Table 4.1.

Data Source	Number of Word Images	Category of Noise Level
Book	1,786	478 medium, 700 high and 608 very high level
Magazine	508	All magazines we used have medium-level noise
Newspaper	690	514 high level and 176 very high level
Regulation	1,990	All regulation documents were low-level
Total	**4,974**	From all levels of noise

Table 4.1: Source and number of word images used

To conduct the experimentation, a SAMSUNG laptop computer with specification Intel(R) Core(TM) i5-2450M CPU @ 2.50 GHz (4 CPUs), 6 Giga Bytes RAM and Windows 7 Ultimate edition 64-bit Service Pack 1 operating system was used. The prototype was developed using MATLAB™ Image Processing Toolbox 7.0 and integrated with the application developed in Java™ programming language using NetBeans IDE 7.1.2.

4.2. Preprocessing Amharic Real-life Document Images

We implemented and also used built-in methods in MATLAB Image Processing Toolbox to reduce noise and convert images to binary. The methods used and code fragments of implemented functions are presented below.

In order to read the images, *imfinfo(filename)*, which is a built-in method in MATLAB Image Processing Toolbox was used as shown in listing 4.1. *imfinfo(filename)* returns a structure whose fields contain information about an image in a graphics file. Since the images were scanned as color (RGB - Red Green Blue) because we want to keep and evaluate every detail (content or noise) of the document image and we converted the image file from RGB to gray-level using *rgb2gray(image)* built-in method in MATLAB to make processing efficient and easy. To identify the color type of the image, *ColorType* property of the *imfinfo()* method is used. *ColorType* is a string indicating the type of image; this includes 'truecolor' for a truecolor (RGB) image, 'grayscale' for a grayscale intensity image.

Once the document image becomes ready for manipulation, the first image preprocessing method implemented in this study is noise filtering or reduction.

Listing 4.1: Reading image data and converting to gray-level image

```
% This program opens image file and converts it to gray scale image
% It takes a filename and returns the original image as a gray level image
function GrayImage = ImageReader(filename)
    % Get the information about graphics file using imfinfo()
    info = imfinfo(filename);
    % Get information about the color type of image
    colorType = [info.ColorType];
    % If the color type of the image is true color or RGB
    if strcmp(colorType,'truecolor')
        % Convert the noisy image to graylevel image using rgb2gray()
        GrayImage = rgb2gray(OriginalImage);
    else
        % The image is a gray scale image. Then, save the image as it is
        GrayImage = OriginalImage;
end;
```

In this study, the final result of image preprocessing is a filtered document image which is evaluated with the original gray scale image using peak signal-to-noise ratio (PSNR) via the method as shown in listing 4.2. We implemented equation (3.11) using MATLAB.

4.2.1. Noise Filtering

To clean the noise in real life printed document images, we evaluate three noise removal techniques explicitly Median filter, Weiner filter and Adaptive Median filtering (AMF). The reason we selected median filtering is because it is very widely used in digital image processing

since, under certain conditions, it preserves edges while removing noise and for certain types of random noise, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size. In addition to this reasons, it has good capability of removing impulsive (salt-and-pepper) noise [80] [82].

The first filtering technique that was used in this research is median filter. The median of a set is the middle value when they are sorted. If there are an even number of values, the median is the mean of the middle two. Median filtering is more effective than convolution when the goal is to simultaneously reduce noise and preserve edges [42].

Listing 4.2: PSNR Measurement Function

```
% MSE  – Mean Square Error
% PSNR – Peak Signal to Noise Ratio
function [MSE, PSNR] = calc_MSE_PSNR(original, preprocessed)
        preprocessed = im2double(preprocessed);
        original = im2double(original);
        [M N] = size(original);
        acc = 0.0;
        for i=1:M
                for j=1:N
                        acc = acc + (original(i,j) - preprocessed(i,j))^2;
                end
        end
        MSE = (1/(M * N)) * acc;
        PSNR = 10*(log10((255^2)/MSE));
```

Median filter replaces the value of a pixel by the median of the gray levels in the neighborhood of that pixel (the original value of the pixel is included in the computation of the median). The median, *j*, of a set of values is such that half the values in the set are less than or equal to *j*, and half are greater than or equal to *j*. In order to perform median filtering at a point in an image, we first sort the values of the pixel in question and its neighbors, determine their median, and assign this value to that pixel [42] [70].

There are two implementations of median filter *"medfilt2()"* in MATLAB. This differ by the number and type of arguments/parameters. The first one *B = medfilt2(I, [m n])* performs median filtering of an image *I*, with each output pixel containing the median value in the m-by-n neighborhood (window size). *medfilt2* pads the image with 0s on the edges, so the median values for the points within *[m n]/2* of the edges might appear distorted. On the other hand, *B = medfilt2(I)* performs median filtering of an image *I* using the default 3-by-3 neighborhood.

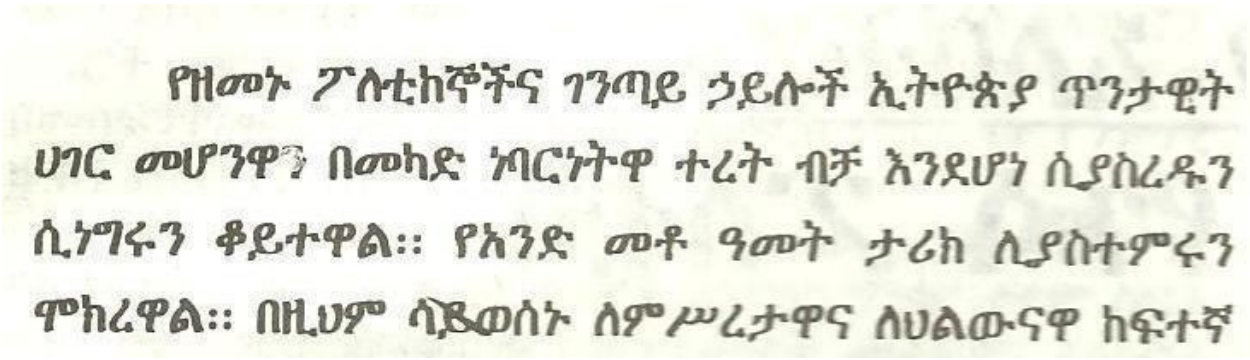
We used the first implementation to create a function called *MedianFilter*. The function takes a gray-level image and the dimension or window size and finally returns a filtered image using the median value. The challenge we faced is while we wrote our function to call the built-in method and choosing window size. This is shown in Listing 4.3.

Listing 4.3: Implementation of Median Filter

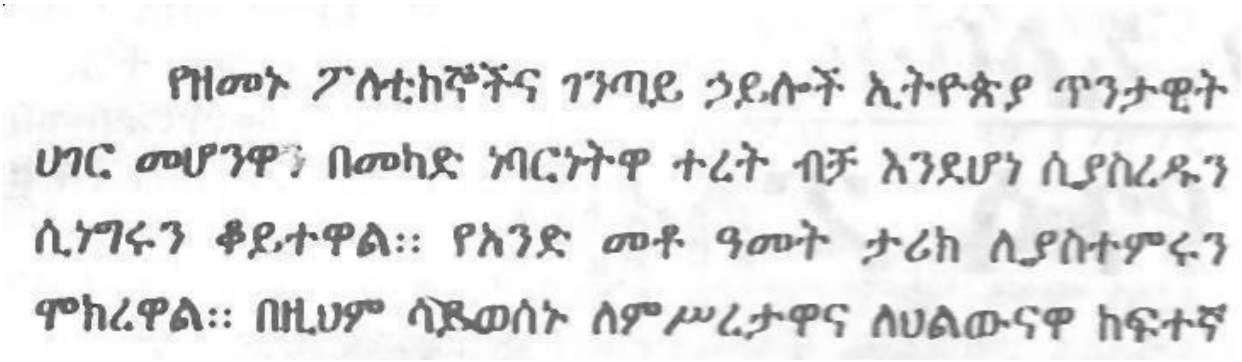
```
function FilteredImage = MedianFilter(GrayImage, size)

    % Apply median filter to the image using medfilt2()

    FilteredImage = medfilt2(GrayImage, [size size]);
```

As we can see from Figure 4.2, the original noisy document image (a) is cleaned using median filter. The PSNR registered is 28.1002. The resulting image (b) is blurred and sharp corners or edges are difficult to identify than the original. Al-Khaffaf [71] also proves this experimentation result as "median filter has the disadvantage of removing fine details and sharp corners in the image". Therefore, we evaluated an adaptive median filtering noise reduction method.

As discussed in section 3.2.3, adaptive median filtering (AMF) is an improvement of median filter which is tested before this. We wrote our own implementation code for AMF because MATLAB does not include this technique in the Image Processing Toolbox. The implementation code in MATLAB is presented in listing 4.4. The function takes the gray-level image as first argument and the maximum window size as second argument. We implemented this function according to algorithm 3.2. Understanding the concept of moving window size and the condition where we move from level A to level B was difficult while writing the code.



(a)



(b)

Figure 4.2: A noisy document image before (a) and after (b) Median filter is applied

Figure 4.3 shows the result of applying adaptive median filter to an image displayed in Figure 4.2 (a). The PSNR improved to 32.9151 and the filtered document image looks better than median filtered (Figure 4.2(b)). But, little blurring is still there. Therefore, we evaluated an adaptive noise removal method, called wiener to come up with a document image that is not blurred.

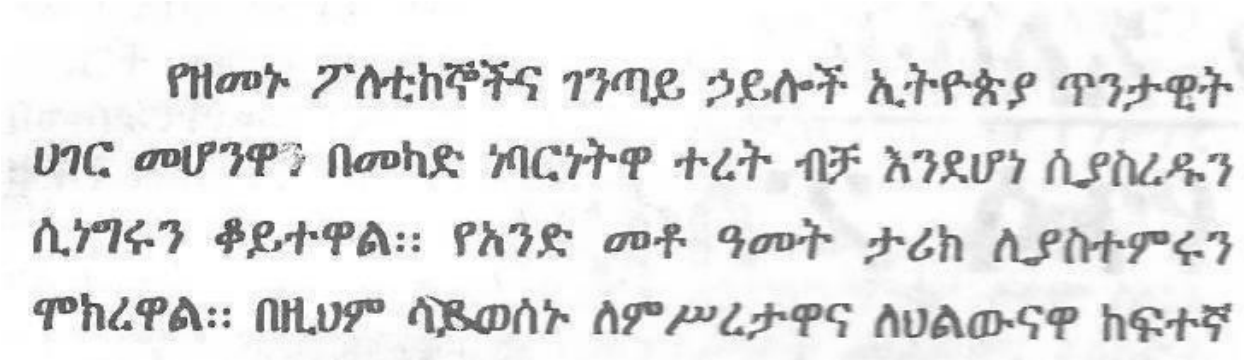


Figure 4.3: Adaptive median filtered noisy document image

The third noise reduction technique used was a low-pass wiener adaptive filtering technique. *"wiener2"* is the MATLAB implementation, which filters a grayscale image that has been degraded by constant power additive noise. *wiener2* uses a pixel-wise adaptive wiener method based on statistics estimated from a local neighborhood of each pixel. We wrote a function, shown in listing 4.5, using one of the varieties which takes an image and the dimension size as a parameter, i.e. *[J, noise] = wiener2(I, [m n])*, estimates the additive noise power before doing the filtering and returns this estimate in noise. Our *WienerFilter* function is similar to the above *MedianFilter* except the built-in MATLAB function here is *wiener2* (see listing 4.5).

Applying wiener filter (33.7229 dB) registers better PSNR than median filter (28.1002 dB) and adaptive median filter (32.9151 dB). This result and visual comparison of Figure 4.2 (b), Figure 4.3 and Figure 4.4 shows that the wiener filtered document image has higher quality in terms of reduction of noise and no blurring when compared with median and adaptive median filtering techniques.

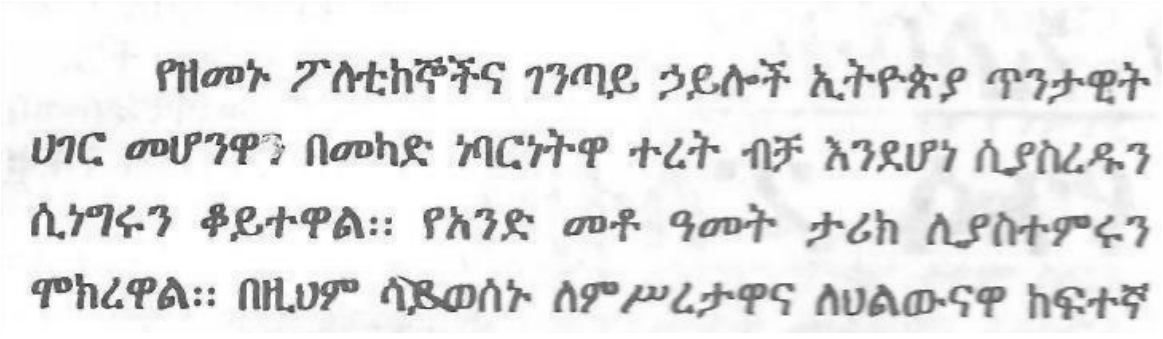


Figure 4.4: A noisy document image after Wiener filter is applied

Listing 4.4: Implementation of Adaptive Median Filtering

```
function FilteredImage = AdaptiveMedianFilter(GrayImage, Smax)
    % get the Width and Height of image GrayImage
    [M, N] = size(GrayImage);
    % Initialization
        FilteredImage = GrayImage;
        FilteredImage(:) = 0;
        ProcessedImage = false(size(GrayImage));

    % Begin filtering - Increase the size of filter window from 3 to Smax by 2
    for i = 3:2:Smax
    % Minimum filtering, maximum filtering and median filtering
        zmin = ordfilt2(GrayImage, 1, ones(i, i), 'symmetric');
        zmax = ordfilt2(GrayImage, i * i, ones(i, i), 'symmetric');
        zmed = medfilt2(GrayImage, [i i], 'symmetric');
    % Check if Zmed is the impulse noise:
    % If no, LevelB = true,and go to Level B; vice versa
        LevelB = (zmed > zmin) & (zmax > zmed) & ~ProcessedImage;
    % If no, zB = true, and go to Level B; vice versa
        zB = (GrayImage > zmin) & (zmax > GrayImage);
    % If LevelB and zB are both true, Zxy is output
        outputZxy  = LevelB & zB;
    % If LevelB is true and zB is false, Zmed is output
        outputZmed = LevelB & ~zB;
    % Set Zxy to output
        FilteredImage(outputZxy) = GrayImage(outputZxy);
    % Set Zmed to output
        FilteredImage(outputZmed) = zmed(outputZmed);
    % Set true pixel ProcessedImage, namely relative output is set
        ProcessedImage = ProcessedImage | LevelB;
    % If all pixel are processed & outputs are obtained --> Completed
        if all(ProcessedImage(:))
            break;
        end
    end

    % Output zmed for any remaining unprocessed pixels.
        FilteredImage(~ProcessedImage) = zmed(~ProcessedImage);
```

Listing 4.5: Implementation of Wiener Filter

```
function FilteredImage = WienerFilter(GrayImage, size)

    % Apply wiener filter to the image using wiener2()

    FilteredImage = wiener2(GrayImage, [size size]);
```

4.2.2. Thresholding / Binarization

The next task after image noise removal is thresholding or binarization which is converting gray-level images to binary (0s and 1s). According to Cheoi [72], we should apply filtering algorithms that enhance the quality of the image before binarization for improved performance. We verified this statement and found that it is correct. We binarize a document image and then tested median, adaptive median and wiener noise reduction filters. But, the result is the same as the binarized image which means no filtering is performed because the image is already in binary format (0s and 1s). Filtering should be applied primarily to gray scale or color images because this type of images contain different levels of intensity. Therefore, we concluded that noise removal techniques should be applied before binarization or thresholding.

A grayscale image is turned into a binary (black and white) image by first choosing a gray level in the original image, and then turning every pixel black or white according to whether its gray value is greater than or less than *T*:

$$\text{A pixel becomes } \begin{cases} white \text{ if its gray level is } > T \\ black \text{ if its gray level is } \leq \text{T} \end{cases}$$

In this study, we implement and evaluate one global and two local thresholding techniques. These are Otsu global thresholding and Niblack and Sauvola local thresholding.

The justification why we selected Niblack's method is because it is one of the local thresholding techniques for segmentation and various scholars [45] [67] [73] [74] evaluated that the output of the method is significant and has most acceptable result out of all thresholding techniques in segmenting text documents. Otsu thresholding can be simply done in *MATLAB Image Processing Toolbox* and we wrote our implementation of Niblack and Sauvola methods.

Otsu's global threshold method [64] finds the global threshold t that minimizes the intra-class variance of the resulting black and white pixels. For all the pixels inside l, Otsu's threshold T is calculated to divide the pixels into two clusters. If the two estimated cluster means $\hat{\mu}_1$ and $\hat{\mu}_2$ are further apart than a user-specified limit, $\|\hat{\mu}_1 - \hat{\mu}_2\| \geq l$, then the pixels inside S are binarized using the threshold value T. When $\|\hat{\mu}_1 - \hat{\mu}_2\| < l$, all the pixels inside S are assigned to the class with the closest updated mean value [67]. This is a standard binarization technique and was implemented using the built-in MATLAB function "graythresh" [38]. Then, the binarization is formed by setting $b_i = 1$ if $x_i >= t$ and $b_i = 0$ if $x_i < t$ [75].

The default binarization technique used in MATLAB Image Processing Toolbox is Otsu thresholding. The function *BW = im2bw(I, level)* converts the grayscale image *I* to a binary image. The output image *BW* replaces all pixels in the input image with luminance greater than level with the value 1 (white) and replaces all other pixels with the value 0 (black) [38]. The function *graythresh* can be used to compute the level which is a value between 0 and 1.

We wrote a function that calculates the level using *graythresh* function and supply the result to *im2bw* Otsu binarization method as shown in listing 4.6.

Listing 4.6: Otsu Thresholding Function

```
function ThresholdedImage = OtsuThresholding(FilteredImage)

    % Apply Otsu thresholding to the filtered image using graythresh and im2bw()

    level = graythresh(FilteredImage);

    % Use the variable 'level' as an argument in im2bw()

    ThresholdedImage = im2bw(FilteredImage, level);
```

According to Otsu's [64] model, binarization can be considered as a two-class discrimination problem for determining a global threshold. However, complicated document images require adaptive binarization, in which the local threshold is calculated for each pixel. Therefore, we

implemented two local thresholding techniques (Niblack and Sauvola) and compare the results of evaluation with Otsu.

Niblack's thresholding is a simple and efficient method for adaptive thresholding developed by Niblack [66]. Trier et al. [73] evaluated eleven different local adaptive binarization methods for gray scale images with low contrast, variable background intensity and noise. Their evaluation showed that Niblack's method performed better than other local thresholding methods. The implementation code for Niblack thresholding algorithm in MATLAB is presented in Listing 4.7.

As shown in Listing 4.7, the code for *Niblack Thresholding* works as follows. First, a filter size *N* should be determined. According to literature [38] and experimental trial-and-error, filter size $N = 60$ is found to register best performance. For other values of *N*, the content of the document image specially at the border is faded. Following this, the local mean, variance and standard deviation are calculated. Then, weight *w* is also determined. In this study and other literature [73], $w = -0.8$ showed better performance. The weight is multiplied by the local standard deviation and summed with the mean. Finally, all pixel values greater than the threshold are replaced with black color.

The third thresholding algorithm we implemented is Sauvola's method which is presented in listing 4.8. The implementation of Sauvola's algorithm (equation 3.9) first accepts a gray-scale image and calculates the mean and standard deviation. Then, we set the value of the parameter, *k* to be 0.5 because this is the preferred value by the developer of the algorithm named J. Sauvola [15] and works fine for this researches purpose. Accordingly, the value of the range of standard deviation, *R* is set to 128 for same reason. Following this, we code the equation that finds the optimal threshold value (equation 3.9) and iterate through the pixel values. If the pixel value is greater than the optimal threshold, then it is black. Otherwise, it is white, i.e. the background of the image.

Listing 4.7: Niblack's Thresholding Function

```
function ThresholdedImage = NiblackThresholding(FilteredImage)

    % Apply Niblack thresholding to the filtered image

    % Select filter size, N = 60 works well

    N = 60;

    % Calculate local mean, variance and standard deviation

    localMean = filter2(ones(N), FilteredImage) / (N*N);

    localVariance = filter2(ones(N), FilteredImage.^2) /(N*N) - localMean.^2;

    localStandardDeviation = sqrt(localVariance);

    % Select the weight and calculate the threshold value

    weight = -0.8;

    threshold = localMean + weight * localStandardDeviation;

    %  Pixel values greater than the threshold are replaced with black color.
    % Otherwise, white color.
    % Finally, return the thresholded image

    ThresholdedImage = FilteredImage > threshold;
```

Listing 4.8: Sauvola's Thresholding Function

```
function ThresholdedImage = SauvolaThresholding(GrayImage)
    % Find the mean & standard deviation required for Sauvola's algorithm
        m = mean(GrayImage(:));
        s = std(im2double(GrayImage(:)));
    % k is a parameter and R is dynamic range of std
        k = 0.5;
        R = 128; % for 8-bit gray scale image
    % obtain width and height of image GrayImage
        [M, N] = size(GrayImage);
    % Implementing Sauvola thresholding algorithm:
    % Implement the equation (3.9)
        value = m *(1 + k *((s/R) - 1))
    % Assign the GrayImage to FilteredImage so we can manipulate and store
    % the output on the ThresholdedImage
        ThresholdedImage = GrayImage > value;
```

4.3. Experimentations

In this study, three experiments are conducted to evaluate the best noise reduction and thresholding technique for real-life Amharic document images. We already evaluated that wiener filter performs better than median and adaptive median filters. But, we want to evaluate and see the effect of the combination of thresholding techniques. Therefore, in all the experimentations, first we read the image file and then preprocessing is conducted. Finally, the filtered document image is written to file.

4.3.1. Experiment 1

In the first experimentation, noise reduction technique of median filter is tested with Niblack's, Otsu's and Sauvola's thresholding algorithms. Printed real-life documents containing low, medium, high and very high level noise are supplied to the module for testing. The quality of the preprocessed image was measured by the widely used image quality measurement method called Peak Signal-to-Noise Ratio (PSNR). Table 4.2 summarizes the result of applying different filter size, *N* for high level noisy document image. The code fragment is presented in Appendix II.

In Niblack's algorithm, selecting the filter size is vital because it determines the quality of the image. According to the different tests conducted on a high noisy document image (see Figure 4.1) excerpted from a newspaper, filter size of *60* cleans the noise better than any other value. The result is depicted in Table 4.2. When we apply filter size of *N = 60*, the PSNR is 57.7938 dB. Therefore, we applied *N =60* in all experiments that evaluate Niblack's thresholding algorithm.

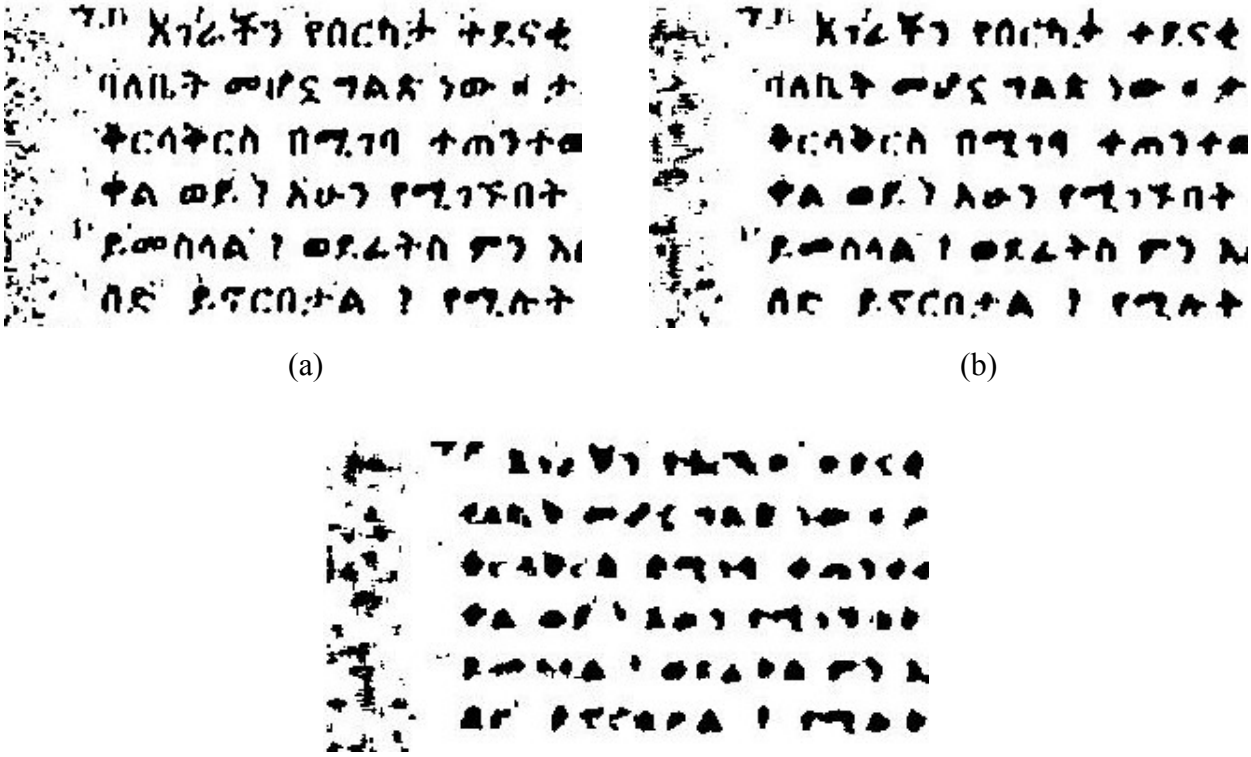
Value of N	**MSE**	**PSNR**
N = 40	0.1110	57.6777
N = 50	0.1093	57.7457
N = 60	**0.1081**	**57.7938**
N = 70	0.1087	57.6502
N = 80	0.1118	57.5939

Table 4.2: Performance registered for different filter size *N*.

The window size or dimension used was 3 x 3. As we increase the size, the performance (measured in PSNR) decreases. According to Marti et al. [63], a 3 x 3 median filter is very appropriate for filtering task. As a result of this, we used dimension 3 x 3 in all experiments conducted in this study. For instance, the sample document image shown in Table 4.3 that contains high level of noise registered decreasing the PSNR value as the window size increased and the visibility of each word decreases as depicted in Figure 4.5. The document image becomes blurred and the words are unreadable or not-recognizable.

Window Size	MSE	PSNR
3 x 3	0.1081	57.7938
5 x 5	0.1149	57.5271
7 x 7	0.1279	57.0623

Table 4.3: Performance registered for different window sizes



(a) (b)

(c)

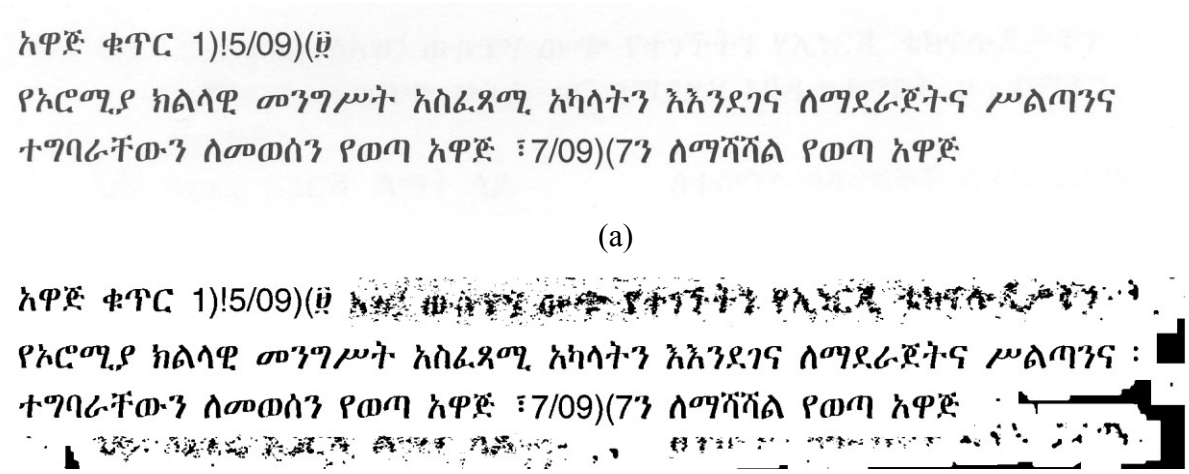
Figure 4.5: Sample document image with different window size (a) 3 x 3 (b) 5 x 5 (c) 7x7

Using $N = 60$ filter size for Niblack's thresholding and 3 x 3 window size for Median filter, the MSE and the PSNR are presented in Table 4.4 for low, medium, high and very high noisy document images. The table shows the performance of the three thresholding techniques (Niblack, Otsu and Sauvola) with Median filtering.

Noise Level	Median and Niblack		Median and Otsu		Median and Sauvola	
	Avg. MSE	Avg. PSNR	Avg. MSE	Avg. PSNR	Avg. MSE	Avg. PSNR
Low	0.1216	57.5625	0.0107	67.8783	0.0096	**68.3356**
Medium	0.0548	*60.9310*	0.0209	65.0486	0.0179	**65.7184**
High	0.0821	59.3304	0.0657	**60.2943**	0.0708	59.9815
Very High	0.0991	58.3313	0.0887	**58.8169**	0.0929	58.4477

Table 4.4: Performance registered in experiment 1

According to Table 4.4, mostly, as the noise level increases from low to medium to high to very high, the MSE result increases and PSNR result decreases. This is false in the case of medium level noise filtering with Median and Niblack. This is because, as shown in Figure 4.6 (b), Niblack's thresholding reveals the show-through noise at the top and bottom of the image and it creates more noise than the original image. At the right corner of the document image, there are black rectangles which are examples of noise created by Niblack's thresholding algorithm. The main reason for the introduction of the noise is mean is biased towards the highest intensity value since Niblack's method uses mean and standard deviation of some window size of the image and



(a)

(b)

Figure 4.6: Sample result of experiment 1

(a) Original low-level noisy image (b) Median filtered and Niblack's thresholding image with PSNR = 58.5187 dB

Based on the experiment conducted, the step-wise combination of median filtering and Sauvola thresholding performs better for low and medium noise level document images. On the other hand, for high and very high level noise, Median and Otsu combination performs better.

4.3.2. Experiment 2

This experiment was conducted to apply and test adaptive median filtering (AMF) with the three thresholding techniques. The code fragment and sample data is presented in Appendix III. As shown in Table 4.5, which shows the result obtained in experiment two, better performance is registered for all noise levels using adaptive median filtering (AMF) while the thresholding techniques performed similarly as experiment 1.

Noise Level	AMF and Niblack		AMF and Otsu		AMF and Sauvola	
	Avg. MSE	Avg. PSNR	Avg. MSE	Avg. PSNR	Avg. MSE	Avg. PSNR
Low	0.1291	57.2878	0.0100	68.1520	0.0089	**68.6837**
Medium	0.0466	61.5829	0.0198	65.2665	0.0164	**66.0584**
High	0.0801	59.5467	0.0634	**60.5549**	0.0673	60.3215
Very High	0.0974	58.4418	0.0869	**58.9359**	0.0935	58.6323

Table 4.5: Performance registered in experiment 2

4.3.3. Experiment 3

In experiment 3, wiener2 filter is evaluated with Niblack, Otsu and Sauvola's thresholding for binarization. The code that is written for this experiment is shown in appendix IV. Table 4.6 shows the performance registered by using wiener filter for noise removal then the three thresholding techniques for binarization.

Noise Level	Wiener and Niblack		Wiener and Otsu		Wiener and Sauvola	
	Avg. MSE	Avg. PSNR	Avg. MSE	Avg. PSNR	Avg. MSE	Avg. PSNR
Low	0.1183	57.6827	0.0099	68.2106	0.0085	**68.8869**
Medium	0.0514	61.2200	0.0198	65.2637	0.0162	**66.1142**
High	0.0788	59.6636	0.0629	**60.6395**	0.0669	60.4232
Very High	0.0968	58.4755	0.0868	**58.9490**	0.0944	58.5937

Table 4.6: Performance registered in experiment 3

Based on the experiment conducted, one can observe that as the noise level increases from low to medium to high to very high, the MSE result increases and PSNR result decreases. As a result,

low noise level documents have the highest signal to noise ratio (68.8869 dB using Wiener and Sauvola). To the contrary, documents with very high level of noise have the lowest PSNR i.e. 58.949 dB tested by Wiener and Sauvola.

Table 4.8 summarizes all the results from the best results from the three experiments conducted in this study.

Noise Level	***Experiment 1***		***Experiment 2***		***Experiment 3***	
	i **Median & Otsu**	*ii* **Median & Sauvola**	*iii* **AMF & Otsu**	*iv* **AMF & Sauvola**	*v* **Wiener & Otsu**	*vi* **Wiener & Sauvola**
Low	67.8783	**68.3356**	68.1520	**68.6837**	68.2106	**68.8869**
Medium	65.0486	**65.7184**	65.2665	**66.0584**	65.2637	**66.1142**
High	**60.2943**	59.9815	**60.5549**	60.3215	**60.6395**	60.4232
Very High	**58.8169**	58.4477	**58.9359**	58.6323	**58.9490**	58.5937

Table 4.8: Summary of best result obtained in all experiments

As clearly shown in Table 4.8, for low and medium levels of noise, the step wise Wiener filtering then Sauvola thresholding registered best performance in terms of PSNR when compared with the other experiments. This result is followed by AMF and Sauvola with only 0.2032 dB difference. In both cases, Sauvola's thresholding is applied which shows that it is a good binarization technique for low and medium noise levels.

However, for high and very high noise levels, the best technique found are step-wise combination of wiener filter and Otsu thresholding with slight increase from adaptive median filter and same thresholding technique i.e. (0.0846 dB for high and 0.0131 dB for very high level noise).

By mathematically comparing the result of noise reduction and thresholding techniques in Table 4.8, wiener filter is better than AMF because using both thresholding techniques, the resulting value of wiener is greater than AMF's result. That is 60.6395 dB is greater than 60.5549 (sub-columns labeled *vi* and *iv*) and 60.4232 dB is greater than 60.3215 dB (sub-columns labeled *v* and *iii*). The same way, Otsu thresholding technique's result is greater than Sauvola's (comparison of sub-columns labeled *vi and iv,* and *v and iii* verifies this finding). In addition to

this, Sauvola's method performs poor for document images that have low brightness compared to Otsu's. It tends to create a white image while there is content in it. Sauvola's method is an improvement to Niblack's and both use mean value which is biased by large values. Figure 4.7 clearly shows an original high-level noisy document image (a), result of Sauvola's method (b), and result of Otsu's method (c).

(a) (b) (c)

Figure 4.7: Comparison between Sauvola's and Otsu's method
(a) an original high-level noisy document image, (b) result of Sauvola's method, (c) result of Otsu's method

Therefore, based on the experiments conducted, the best performance and quality document image is the one that is first filtered using the 2-Dimensional adaptive noise-removal filtering (known as wiener filter) and binarized using Otsu thresholding.

4.4. Integrating Noise Filtering and Thresholding Algorithms to the Amharic DIRS

By conducting the above three experiments, we found the best combination of filtering and thresholding algorithms to be wiener and Otsu respectively. In this experiment, we integrated the preprocessing module we developed in MATLAB with the previous attempts to develop Amharic DIRS developed in Java. To integrate the MATLAB code with Java, we used MATLAB Builder JA software. MATLAB® Builder™ JA enables to create Java™ classes from MATLAB® programs. These Java classes can be integrated into Java programs and deployed royalty-free to desktop computers or web servers that do not have MATLAB installed [71].

The following Java program (see listing 4.8) was developed by the researchers to call the MATLAB file that is compiled as Java package. After integrating the image preprocessing module developed in MATLAB, we used low-level, medium-level, high-level and very high-

level noisy document images to evaluate the performance of the system which is measured by precision, recall and F-measure. Integrating the two programming languages was an exigent task for the researchers because of the different parameters and procedures that must be followed strictly. This has consumed the time of the researchers.

The number of words detected differs when document images are noisy and cleaned. As shown in Table 4.9, when we use noisy document images, the total number of word images using the low, medium, high and very high noise level dataset was 4,622. After preprocessing, the detected word images increased by 160 i.e. there is a 3.34% increase in the number of word images detected.

Noise level	**Number of word images detected**	
	Before integration	**After integration**
Low	1,990	1960
Medium	886	883
High	1,214	1155
Very high	532	784
Total	**4,622**	**4,782**

Table 4.9: Number of word images detected before and after integration of the proposed module using different dataset

We selected query words and tested the performance using the different noisy documents. Then, we applied noise reduction and thresholding to the same noisy document images and produced clean images. This clean document images are tested using same query words and the performance is measured again. Experimental results showing before and after integration is shown in tables 4.10 for low-level, 4.11 for medium-level, 4.12 for high-level and 4.13 for very high-level noise.

Listing 4.8: Integrating the MATLAB Implementation with Java

```
/** ImagePreProcessing.java
 * Author: Biniam Asnake
 */

/* Necessary package imports */
import ADIRsImagePreprocessing.ImagePreprocessingClass;
import com.mathworks.toolbox.javabuilder.MWArray;
import com.mathworks.toolbox.javabuilder.MWNumericArray;

public class ImagePreProcessing
{
        public void ImagePreProcessing()     // Default Constructor
        {
                MWNumericArray n = null;   // Stores input value
                Object[] result = null;    // Stores the result
                ImagePreprocessingClass WienerOtsu = null;
                        // Stores ImagePreProcessing instance
                try
                {
                        /* Create new ImagePreprocessingClass object */
                        WienerOtsu = new ImagePreprocessingClass();

                        /* Call the ImagePreprocessingClass matlab file */
                        WienerOtsu.NoiseFilteringAndThresholding();
                        System.out.println("Done Image Preprocessing.");
                }
                catch (Exception e)
                {
                        System.out.println("Exception: " + e.toString());
                }
                finally
                {
                        /* Free native resources */
                        MWArray.disposeArray(n);
                        MWArray.disposeArray(result);
                        if (WienerOtsu != null)
                        {
                                WienerOtsu.dispose();
                        }
                }
        }
}
```

Query word	Before integration of proposed module			After integration of proposed module		
	Recall	**Precision**	**F-measure**	**Recall**	**Precision**	**F-measure**
መንግስት	100	100	100	100	100	100
ሥራ	100	100	100	100	100	100
አገልግሎት	100	67	80.24	100	67	80.24
ርዕስ	75	100	85.71	100	100	100
አበል	100	50	66.67	100	50	66.67
Average	**95**	**83.4**	**86.52**	**100**	**83.4**	**89.38**

Table 4.10: System performance for low-level noisy document images before and after integration of the proposed module

For document images that contain small amount of noise, the performance of the system without the integration of the module developed in this study was 86.32% F-measure. After cleaning the noise using wiener filter and binarizing using Otsu, the performance of the system increased to 89.38% F-measure which showed a 2.86% improvement from earlier system. Only small improvement is registered because the previous system can, to some extent, clean low level noisy document images since fixed thresholding algorithm is implemented by Mesfin [35].

Query word	Before integration of proposed module			After integration of proposed module		
	Recall	**Precision**	**F-measure**	**Recall**	**Precision**	**F-measure**
አገር	100	75	85.71	100	50	66.67
ቅርስ	100	50	66.67	100	100	100
ቤት	0	0	0	100	100	100
ክፍል	100	33.33	49.99	100	50	66.67
ምስለ	100	75	85.71	100	100	100
Average	**80**	**46.66**	**57.62**	**100**	**80.00**	**86.67**

Table 4.11: System performance for medium-level noisy document images before and after integration of the proposed module

As we can see in Table 4.11, the performance of the earlier system with no image noise detection and removal and fixed thresholding achieved 57.62% F-measure. After we integrate our module that performs wiener filtering and Otsu thresholding, the performance of the system increased to

86.67% F-measure. For medium level of noise, the retrieval system showed an improvement of 29.05% in F-measure.

Query word	Before integration of proposed module			After integration of proposed module		
	Recall	**Precision**	**F-measure**	**Recall**	**Precision**	**F-measure**
ቅዱስ	0	0	0	100	75	85.71
ዓለም	0	0	0	100	66.67	80.00
ምሥጋና	100	100	100	50	100	66.67
ኢትዮጵያ	100	33.33	49.99	100	75	85.71
አንድ	100	100	100	100	75	85.71
Average	**60**	**46.66**	**50.00**	**90**	**68.33**	**80.76**

Table 4.12: System performance for high-level noisy document images before and after integration of the proposed module

For high noisy document images, the performance increased by 50.00% F-measure where the new result is 80.76%. This is due to the fact that the filtering and binarization methods were effective in removing the noise that is prevalent in real-life document images.

Query word	Before integration of proposed module			After integration of proposed module		
	Recall	**Precision**	**F-measure**	**Recall**	**Precision**	**F-measure**
ቅዱስ	100	25	40	100	50	66.67
አንድ	100	66.67	80	100	100	100
ኢትዮጵያ	50	50	50	100	50	66.67
ዓለም	0	0	0	66.67	100	80
አብያተ	0	0	0	50	50	50
Average	**50**	**28.33**	**34.00**	**83.33**	**70.00**	**72.67**

Table 4.13: System performance for very high level noisy document images before and after integration of the proposed module

Document images that contain very high noise including background noise, cuts, folds, blur and low contrast are indexed and the retrieval performance of the Amharic DIR system was very low as 34% F-measure. But, after applying the selected filtering and thresholding techniques (wiener and Otsu respectively), the F-measure augmented to 72.67% which showed 38.67% improvement in the retrieval performance.

Generally, the integration of the system improved retrieval performance of the previous system. The overall performance of the system for all levels of noise is depicted in Table 4.14. Without any noise removal and only a simple fixed thresholding algorithm, Adane's [19] system outperforms the two previous attempts of Mesfin [35] and Abreham [14]. Moreover, the system that is developed and integrated in this research surpass Adane's [19] system.

For low level noisy document images, his system's F-measure was 80.46%. This system improved this result to 89.38%. Similarly, for medium and high noise level the system's performance was 66.66% and 55.82% respectively.

Our system enhanced this values to 86.67% and 80.76% for medium and high level respectively. His system was not evaluated on very high noisy document images which are affected by background noise, blur, cuts, folding and other degradations. Our system achieved 72.67% for very high level noise.

As mentioned in section 1.2, the overall performance of the last Amharic DIRS by Adane's [19] is 67.64% F-measure without considering very high level noisy document images while this proposed system achieved 85.6% F-measure. This result is 17.96% higher than the previous system's performance for document images that are highly affected by noise.

Level of Noise	**Recall**	**Precision**	**F-measure**
Low	100%	83.4%	89.38%
Medium	100%	80%	86.67%
High	90%	68.33%	80.76%
Very high	83.33%	70%	72.67%
Average*	**96.67%**	**77.24%**	**85.60%**
Average**	**93%**	**75%**	**82.37%**

Table 4.14: Summary of result achieved in this study for different noise levels

* without considering the performance of the system on very high level noisy documents
* considering the performance of the system on very high level noisy documents

4.5. Searching Using Multiple Query Terms

In addition to integrating image preprocessing to the previously developed Amharic DIRS, this study also aims to improve the searching facility by accepting multiple query words from the user. The previous system by [14] [19] [35] accepts only a single query.

As can be seen in the newly proposed architecture of the Amharic DIRS in this study (Figure 3.1), we used one of the widely used model: Vector Space IR Model. The user is able to enter multiple word query which is split or segmented using the space in between words as separator. Each word is rendered (converted to image) and stored individually. Feature extraction procedure is performed after the image is preprocessed using binarization or thresholding technique. Then, similarity between the feature vectors of the query image and the feature vector of the index file is measured using DTW (Dynamic Time Warping) matching technique by Adane [19]. Following, the list of similarity score is collected and documents are ranked based on the score. Finally, the relevant retrieved documents are presented to the user in ranked order.

We implemented (see Appendix VI) and experimented searching using multiple query word in two approaches: recall-oriented and precision-oriented systems. A recall-oriented system retrieves documents that contain at least one of the query words (working as ‘or’ logical operator) which increases the recall performance of the system. On the other hand, precision-oriented system works as ‘and’ logical operator where the documents retrieved must contain all the query terms.

We experimented on a low noise level dataset that has been cleaned by combination of the selected filtering and thresholding techniques (Wiener and Otsu). The dataset contains 823 word images excerpted from regulation documents. According to the result of the experimentation shown in Table 4.15, in the precision-oriented system, the average value of recall, precision and F-measure measures are 100%, 53.33% and 65.33% respectively. Highest performance is registered by the recall-oriented system (100% recall, 68.33% precision and 77.33% F-measure).

The precision-oriented system performs less precision value but with same recall performance because it only retrieves documents that contain both of the query terms. If these query words are

not found in a similar document, the precision is zero since no relevant document appears (zero divided by the number of retrieved documents is always zero).

Multiple Query Terms	Recall-Oriented System			Precision-Oriented System		
	Recall	**Precision**	**F-Measure**	**Recall**	**Precision**	**F-Measure**
ከፍተኛ አገልግሎት	100	66.67	80	100	66.67	80
መሠረተ ልማት	100	100	100	100	50	66.67
ዘመናዊ አዲተር	100	25	40	100	25	40
ጠቅላላ ሀብት	100	50	66.67	100	25	40
አጭር ርዕስ	100	100	100	100	100	100
Average	**100%**	**68.33%**	**77.33%**	**100%**	**53.33%**	**65.33%**

Table 4.15: Result of experimenting Multiple Query Word Retrieval

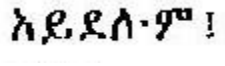
4.6. Findings and Challenges

In this research, we attempted to integrate noise filtering and thresholding techniques and improve searching by supplying multiple queries to the Amharic DIR system by [14] [19] and [35]. Experimental result shows that a wiener filter followed by Otsu thresholding registers better image preprocessing performance of cleaning all levels of noise prevalent in the document images. On the average, 82.37% F-measure is achieved for all noise levels. Searching with multiple query words in a recall-oriented system achieves better retrieval performance of 77.33% F-measure.

Although a better performance is registered than the previously developed system, the retrieval effectiveness is highly affected by segmentation errors. This can be clearly identified while counting the number of word images extracted. For instance, in one very high noise level document image, while there are 183 word images, only 131 are extracted by the old system and this result is improved to 170 by the proposed system. On the other hand, from a high noisy document image that contains 191 word images, only 166 are extracted by the proposed system and a better extraction performance is registered by the old system when supplied. This clearly shows that the system's performance is highly affected by segmentation errors. The current system either considers two or more separate words as one because of noise or a single word as

two or more words when the noise is removed and the space between characters of a single word is large enough to be a word (segmentation threshold value) by the segmentation algorithm. Examples of word images that show the two type of segmentation errors are displayed in Figure 4.8.

(a)



(b)

Figure 4.8: Examples of segmentation errors

(a) Multiple words might be considered as one because of noise, (b) Single word might be considered as multiple because of noise filtering

In addition to segmentation errors, the performance of the system is highly affected by change of shape of words because of over filtering. Word variants also affect the retrieval performance of the system. For instance, the word 'እየመጡላቸሁ' has a root word 'መጣ' which has a prefix 'እየ', an infix '-ኡ' (after መጥ-) and a suffix 'ላቸሁ'.

We also found out that Niblack's binarization method produces more background noise but it completely recovers text from severely degraded documents. The major drawback of Niblack's thresholding technique is for part of the image in which the window does not contain any objects, the method detects the noise as objects and elaborates them.

Moreover, we tried to develop a hybrid algorithm of Otsu and Niblack by finding the value of the threshold using Niblack and supplying that result to Otsu. However, the result is the same as Niblack's method in terms of PSNR measure and image quality.

Another finding is that Adaptive median filtering has issues related with efficiency. It takes much more time than all other filtering techniques tested in this paper because it has a *for loop* that iterates up to the maximum window size.

Therefore, an advanced segmentation algorithm that adjusts the threshold dynamically for various documents which have different spaces between words and characters and a better indexing structure should be integrated to come up with a practical document image retrieval system for Amharic document images. Furthermore, real-life document images that are skewed should be considered in further studies.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

Real-life documents such as books, magazines, newspapers and reports contain vital information regarding the social, political, cultural, economic and other aspects. To make this documents accessible and searchable by the general public, OCR (optical character recognition) systems were explored to convert document images into their textual equivalent to ease searching using the existing search engines. But, developing an OCR tool is a long term process since the recognition performance is low and it requires human correction. Therefore, a document image retrieval system without explicit recognition came into existence.

This research is a continuation of other attempts to develop a retrieval system from printed Amharic real-life document images. Here, we developed and integrated an image preprocessing specifically noise reduction and thresholding techniques in addition to searching with multiple queries to augment the performance of the system.

5.1. Conclusion

The main objective of this research is to integrate effective image preprocessing techniques of noise reduction and thresholding as well as multiple word rendering and querying to enhance the effectiveness of relevant document retrieval from printed real-life Amharic document images such as books, magazines and newspapers.

In this study, three noise reduction (or filtering) techniques, explicitly median, wiener, adaptive median filters are evaluated in combination with Otsu, Niblack and Sauvola thresholding algorithms. A series of experiments were conducted to select the optimal combination of noise removal and thresholding techniques on low, medium, high and very high noisy document images.

The performance results obtained show that the combination of wiener filtering and Otsu thresholding achieved the highest peak signal-to-noise ratio (PSNR) of 68.2106 dB for low level, 65.2637 dB for medium level, 60.6395 dB for high level and 58.949 dB for very high level noisy printed Amharic real-life document images.

Then, this documents are indexed by the previous retrieval system and the performance is measured before and after integration of the module separately using the four levels of noise. This showed a great improvement in all the noise levels. For low noise level document images, the performance increased from 80.46% to 89.38% F-measure. Correspondingly, for medium noise level, it increased from 66.66% to 80.67% and for high noise level from 55.82% to 80.76% F-measure. Moreover, the performance is evaluated on document images that contain very high level noise and registered 72.67% F-measure.

This study also attempts to improve the on-line searching procedure of the Amharic DIRS by permitting users to enter their information need with more than one query word and retrieve relevant documents in ranked order. In recall-oriented approach, 77.33% F-measure is achieved when tested on cleaned low level noisy document images.

Nevertheless, the system's effectiveness is highly affected by segmentation errors. Since the noise removal and thresholding techniques create gap between characters, the system considers a single word as two words. Further, skewed real-life documents should be considered.

5.2. Recommendations

The current research enhanced the effectiveness of the previous systems by integrating noise filtering and thresholding techniques. Yet, to improve the effectiveness and efficiency of the system, the following issues must be dealt with.

1. To improve the performance of the system, an advanced segmentation technique that dynamically adjusts the threshold value according to the document under consideration should be developed.
2. To enhance the retrieval performance, further studies should be conducted to design an effective stemmer for Amharic word variants in document images.
3. Real-life documents that are skewed are not considered in this research. Hence, in order to come up with a practical Amharic document image retrieval system, image preprocessing tasks such as skew detection and correction, image restoration and edge detection are vital.

4. This work improves the rendering and searching processes of the previous system to accept multiple queries. We recommend further work on adopting other information retrieval models to search from document images.
5. Much information and knowledge is kept in historical documents written in Amharic language using different formats such as typewritten, handwritten including special handwriting style called "Qum Tsehfet". Therefore, developing a retrieval system for historical document images is an open research topic.
6. No document image retrieval system is developed for other Ethio-Semitic languages. Since their writing system is similar with Amharic language, there is a need to extend the present work towards other Semitic languages document image retrieval.

REFERENCES

[1]. S. Marinai, "Introduction to Document Analysis and Recognition," *IEEE Transactions on PA MI*, vol. 27, no. 1, 2006, pp. 23–43.

[2]. C.V. Jawahar, Million Meshesha, and A. Balasubramanian, "Searching in Document Images," *in Proc. 4th Indian Conf. on Computer Vision, Graphics and Image Processing (ICVGIP),* India, 2004, pp. 622-627.

[3]. Million Meshesha and C.V. Jawahar, "Matching Word Images for Content-Based Retrieval from Printed Document Images," *International Journal on Document Analysis and Recognition*, vol. 11, no. 1, 2008, pp.29-38.

[4]. S. Akram, Dr. M.U. Dar, and A. Quyoum, "Document Image Processing - A Review," *International Journal of Computer Applications*, vol. 10, no.5, 2010, pp.234-243.

[5]. David Doermann, "The Indexing and Retrieval of Document Images: A Survey," *Computer Vision and Image understanding*, vol. 70, no. 3, 1998, pp. 287-298.

[6]. R. Kasturi, L. O'gorman and V. Govindaraju, "Document Image Analysis: A primer," *Sadhana*, vol. 27, no.1, 2002, pp. 3–22.

[7]. Encyclopædia Britannica, "*Ethiopia*," Encyclopaedia Britannica Ultimate Reference Suite, Chicago: Encyclopædia Britannica, 2010.

[8]. Wondwossen Mulugeta, "Optical Character Recognition for Special Type of Handwritten Amharic Text ("Yekum Tsifet"): Neural Network Approach," M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 2004.

[9]. Million Meshesha and C.V. Jawahar, "Optical Character Recognition of Amharic Documents*,"* African Journal of Information and Communication Technology (AJICT), vol. 3, no. 2, 2007, pp. 53-66.

[10]. "Ethiopia: Language & Culture," National African Language Resource Center (NALRC), University of Wisconsin, Madison, 2010, pp. 1-2.

[11]. Million Meshesha, "Recognition and Retrieval from Document Image Collections," PhD Dissertation, International Institute of Information Technology, India, 2008.

[12]. C.D. Manning, P. Raghavan and H. Schutze, *Introduction to information retrieval*, USA: Cambridge university press, 2008.

[13]. B. Ricardo and B. Ribeiro-Neto, *Modern Information Retrieval*. A Division of the Association for Computing Machinery: Addison-Wesley, ACM Press, 1999.

[14]. Abreham Gebretsadik, "Searching in Amharic Document Image Corpus," M.Sc. Thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2010.

[15]. J. Sauvola, T. Seppanen, S. Haapakoski and M. Pietikainen, "Adaptive Document Binarization," in Proc. 4th Int. Conf. On Document Analysis and Recognition, Ulm, Germany, August 1997, pp.147-152.

[16]. Y. Lu and C.L. Tan, "Information Retrieval in Document Image Databases," IEEE Transaction on Knowledge and Data Engineering, vol. 16, no. 11, 2004, pp. 42-49.

[17]. C. L. Tan, W. Huang, Z. Yu and Y. Xu, "Imaged Document Text Retrieval Without OCR," Institute of Electrical & Electronic Engineers (IEEE) Transaction on Pattern Analysis and Machine Intelligence, vol. 24, no. 6, 2004, pp. 838-844.

[18]. M.C. Motwani, M.C. Gadiya, R.C. Motwani, and F.C. Harris, Jr., "Survey of Image Denoising Techniques," *in Proc. of GSPx, Santa Clara Convention Center,* Santa Clara: CA, 2004, pp. 25-37.

[19]. Adane Letta, "Feature Extraction and Matching In Amharic Document Image Collections," M.Sc. Thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2011.

[20]. K. Zagoris, K. Ergina, N. Papamarkos, "A Document Image Retrieval System," *Engineering Applications of Artificial Intelligence*, vol. 3, no. 2, 2010, pp.872-879.

[21]. M.B. Kokare and M.S. Shirdhonkar, "Document Image Retrieval: An Overview," *International Journal of Computer Applications (0975 – 8887)*, vol. 1, no. 7, 2010, pp.114-119.

[22]. L. M. Paul, *Ethnologue: Languages of the World*, 16th ed. Dallas, Texas: SIL International, 2009.

[23]. A. Wimsatt and R. Wynn, "Amharic Language and Culture Manual," Texas State University, Texas, 2011.

[24]. S. Uhlig, *Encyclopedia Aethiopica,* 1st ed., Wiesbaden: Harrassowitz Verlag, 2003.

[25]. Tilahun Yeshambel, "Amharic Document Image Retrieval Using Lingustic Features," M.Sc. Thesis, Department of Computer Science, Addis Ababa University, Addis Ababa, Ethiopia, 2011.

[26]. Worku Alemu, “The Application of OCR Techniques to the Amharic Script,” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 1997.

[27]. Ermias Abebe, “Recognition of formatted Amharic text using OCR techniques,” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 1998.

[28]. Million Meshesha, “A Generalized Approach to Optical Character Recognition (OCR) of Amharic Texts,” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 2000.

[29]. Yaregal Assabie, “Optical Character Recognition of Amharic Text: An Integrated Approach,” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 2002.

[30]. Abay Teshager, “Amharic Character Recognition System for Printed Real-Life Documents,” M.Sc. Thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2010.

[31]. Sertse Abebe, “Bilingual Script Identification For Optical Character Recognition Of Mixed Amharic and English Printed Document,” M.Sc. Thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2011.

[32]. Dereje Teferi, “Optical Character Recognition of Typewritten Amharic Character” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 1999.

[33]. Nigussie Tadesse, “Handwritten Amharic Text Recognition Applied to the Processing of Bank Checks,” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 2000.

[34]. Mesay Hailemariam, “Line Fitting To Amharic OCR: The Case of Postal Address,” M.Sc Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 2003.

[35]. Mesfin Worku, “Amharic Document Image Retrieval without Explicit Recognition,” M.Sc. Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, 2009.

[36]. M. Berndtsson, J. Hansson, B. Olsson, B. Lundell, *Thesis Projects: A Guide for Students in Computer Science and Information Systems*, 2nded. London: Springer-Verlag Limited, 2008.

[37]. C.R. Kothari, *Research Methodology: Methods and Techniques*, 2nded. India: New Age International Publishers, 2004.

[38]. MathWorks, "MATLAB Image Processing Toolbox 7.0 User's Guide," The MATHWORKS Inc, 2012, Available at http://www.matlab.com, [Accessed on: 6/3/2012].

[39]. C. Torrence*, Image Processing IDL Version 7.1*, 1st ed., ITT Visual Information Solutions Inc., 2009.

[40]. A. R. Clive, *The Principles of Amharic,*Institute of African Studies, University of Ibadan, 1968.

[41]. L. J. Galbiati, *Machine Vision and Digital Image Processing Fundamentals,* New Jersey: Prentice Hall, 1990.

[42]. R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, 2nd ed., Upper Saddle River, New Jersey: Prentice-Hall, 2002.

[43]. S. Marinai, "A Survey of Document Image Retrieval in Digital Libraries," *in Proc. 9th Colloque International Francophone Sur l'Ecritet le Document (CIFED)*, 2006, pp. 193–198.

[44]. W. Burger, M. J. Burge, *Digital Image Processing - An Algorithmic Approach using Java*. New York: Springer-Verlag, 2008.

[45]. K. Khurshid, "Analysis and Retrieval of Historical Document Images," PhD Dissertation, Université Paris Descartes, France, 2009.

[46]. H. K. A. Devi, "Thresholding: A Pixel-Level Image Processing Methodology Preprocessing Technique for an OCR System for the Brahmi Script," *Asian Journal Image Processing*, vol. 1, no.3, 2006, pp. 161-165.

[47]. A. Karthikeya, "Image Noise Reduction Algorithms," *Digital Signal Processing*, ElectronicsBus Magazine, 2012.

[48]. M. L. Bender and Hailu Fulass, *Amharic Verb Morphology*, Committee on Ethiopian Studies, East Lansing: African Studies Center, Michigan State University, 1978.

[49]. R. D. Lins, M. Kamel and A. Campilho, "A Taxonomy for Noise in Images of Paper Documents - The Physical Noises,", i*n Proc. 7th International Conference on Image Analysis and Recognition (ICIAR)*, Springer, Heidelberg, vol. 56, no. 27, 2009, pp. 844–854.

[50]. M. Cheriet, R. F. Moghaddam, "DIAR: Advances in Degradation Modeling and Processing," *in Proc. 6th International Conference on Image Analysis and Recognition (ICIAR)*, Springer, Heidelberg, vol. 51, no. 12, 2008, pp. 1–10.

[51]. D. Vernon, *Machine Vision*, 1st ed., USA: Prentice-Hall, 1991.

[52]. T.M. Ha and H. Bunke, "Image Processing Methods for Document Image Analysis," *Handbook of Character Recognition and Document Image Analysis*, 1997, pp.1-47.

[53]. N. Efford, *Digital Image Processing: A Practical Introduction Using Java*, First Edition, Pearson Education Limited, 2000.

[54]. L. Lawrence. (n.d.), Ancient Scripts [Online]. Available: http://www.ancientscripts.com [Accessed Date: 2 /2/2012].

[55]. M. L. Bender, S. W. Head and R. Cowley, *Language in Ethiopia: The Ethiopian Writing System.* Oxford University Press, 1976.

[56]. Daniel Yacob, *Developments towards an Electronic Amharic corpus,* Geez Frontier Foundation, USA: Verginia, 2005.

[57]. L. Wolf, *Reference Grammar of Amharic*, Wiesbaden: Harrassowitz Verlag, 1995.

[58]. Martha Yifiru, "Morphology-Based Language Modeling for Amharic," PhD Dissertation, Department Informatik, Universität Hamburg, 2010.

[59]. Kesis Kefyalew Merahi, *The Contribution of Ethiopian Orthodox Church to Ethiopian Civilization,* Addis Ababa, 1987.

[60]. E. Ataer and P. Duygulu, "Retrieval of Ottoman Documents," *in Proc. 8th ACM SIGMM International Workshop on Multimedia Information Retrieval,* 2006, pp.155-162.

[61]. G. Blanchet and M. Charbit, *Digital Signal and Image Processing using MATLAB,* , John Wiley & Sons, 2003.

[62]. T. Acharya and A. K. Ray, *Image Processing - Principles and Applications*, John Wiley & Sons, 2005.

[63]. J. Martí, J. Miguel, B. A. Maria, M. J. Serrat, "Pattern Recognition and Image Analysis," *in Proc. 3rd Iberian Conference*, Girona, Spain, 2007, pp. 43-52.

[64]. N. Otsu, "A Threshold Selection Method from Gray Level Histograms," *IEEE Transaction on System*, Man and Cyber, vol. 9, no. 1, 1979, pp. 62-66.

[65]. G. Leedham, C. Yan, K. Takru, J. H. N. Tan and L. Mian, "Comparison of Some Thresholding Algorithms for Text/Background Segmentation in Difficult Document Images,"

in Proc. the 7th International Conference on Document Analysis and Recognition, 2003, pp.121-126.

[66]. W. Niblack, *An Introduction to Digital image processing*, Prentice Hall, 1986.

[67]. B. D. Trier and A. K. Jain, "Goal-Directed Evaluation of Binarization Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, 1995, pp. 11-21.

[68]. C. A. Boiangiu, A. Olteanu, A. V. Stefanescu, D. Rosner, "Local thresholding image binarization using variable-window standard deviation response," *in Proc. Annals of DAAAM*, 2010, pp. 204-208.

[69]. M. Delalandre. (2010). Performance Evaluation of Document Image Analysis Systems: A Primer [Online]. Available: http://mathieu.delalandre.free.fr/teachings/evaluation/, [Access Date: 4/1/2012].

[70]. B. M. Singh, R. Sharma, A. Mittal, D. Ghosh, "Parallel Implementation of Niblack's Binarization Approach on CUDA," *International Journal of Computer Applications*, vol. 32, no.2, Oct. 2011, pp. 22-27.

[71]. G. N. Sarage and S. S. Jambhorkar, "Noise Removal from Mammographic Image based on Mean and Median Filtering Techniques," *International Journal of Advanced Research in Computer Science*, vol. 2, no.4, 2011, pp. 498-500.

[72]. K. J. Cheoi, "A new Binarization Method of Text Images Using Image Enhancement Techniques," *Journal of the Research Institute for Computer and Information Communication*, vol. 15, no. 2, 2007, pp. 33-40.

[73]. O. D. Trier and T. Taxt, "Evaluation of binarization methods for document images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, no. 7, 1995, pp. 312–315.

[74]. S. Farid, "Local thresholding image binarization using variable-window standard deviation response," *in Proc. International Conference on Emerging Technologies (ICET)*, Peshawar, Pakistan, 2009, pp. 280 - 286.

[75]. M. R. Gupta, N. P. Jacobson, E. K. Garcia, "OCR binarization and image pre-processing for searching historical documents," *Pattern Recognition*, vol. 40 , 2007, pp.389–397.

APPENDIX I: Experiment 1 Source Codes and Sample Document Images

```
% This program opens image file and
% converts it to gray scale image
function [OriginalImage, GrayImage] = ImageReader(filename)
% Read the image using imread
    OriginalImage = imread(filename);
% Convert the noisy image to graylevel image using rgb2gray()
    GrayImage = rgb2gray(OriginalImage);
```

==========

```
% This is the First experimentation
% Medain filtering is used for noise reduction and
% Niblack, Otsu and Sauvola thresholdinsg is used for binarization
% MSE is Mean Square Error
% PSNR is used to measure the peak signal to noise ratio
function ExMedain
    % Clear the command window
    clc
    % Close all figures opened
    close all
    % Read the image by calling ImageReader function and passing the file name
    [OriginalImage, GrayImage] =
    ImageReader('G:\FinalImageCorpus\VeryHighLevelNoise\Newspaper3.jpg');
    % Apply median filter by calling MedianFilter function and passing the gray
    % image file and the dimension
    FilteredImage = MedianFilter(GrayImage, 3);
    % Call NiblackThresholding method by passing the Median Filtered Image
    ThresholdedImage = NiblackThresholding(FilteredImage);
```

*% Call **OtsuThresholding** method by passing the Median Filtered Image*

ThresholdedImage = OtsuThresholding(FilteredImage);

*% Call **SauvolaThresholding** method by passing the Median Filtered Image*

ThresholdedImage = SauvolaThresholding(FilteredImage);

% Display the images

ImageDisplayer(OriginalImage, 'Original Noisy Image');

ImageDisplayer(GrayImage, 'Gray Image');

ImageDisplayer(FilteredImage,'Filtered Image');

ImageDisplayer(ThresholdedImage,'Thresholded Image');

% Calculate PSNR (Peak Signal to Noise Ratio) by calling the function calcPSNR.

[MSE, PSNR] = calc_MSE_PSNR(GrayImage, ThresholdedImage)

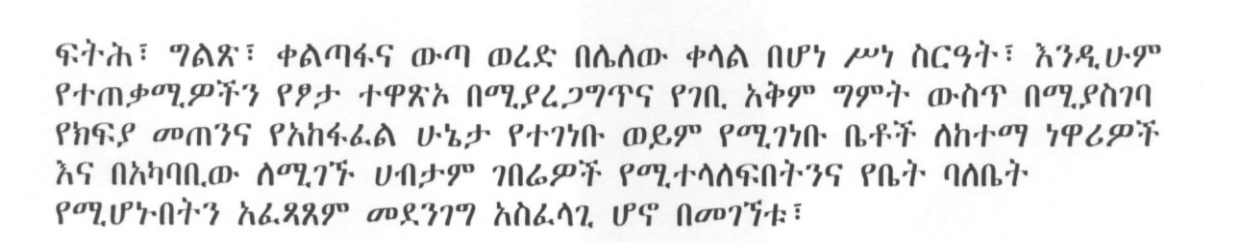
% Write the thresholded image to a jpg file

report = ImageWriter(ThresholdedImage,
'G:\FinalImageCorpus\VeryHighLevelNoise\EX_MN_Newspaper3.jpg');
disp(report)

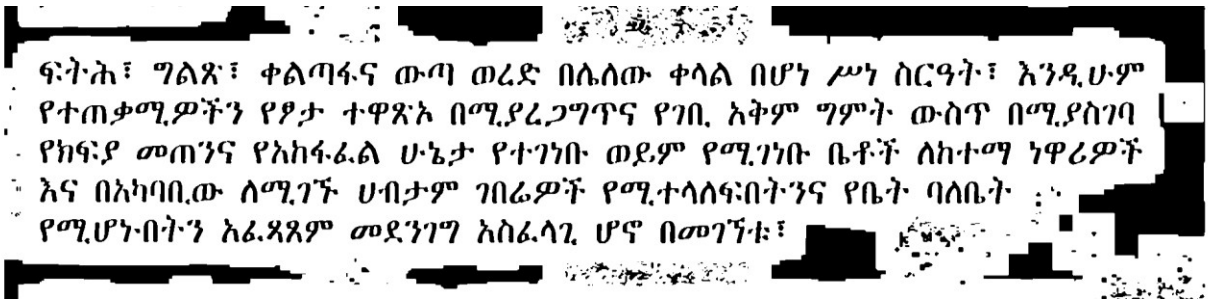
==========

Low Level Noise:

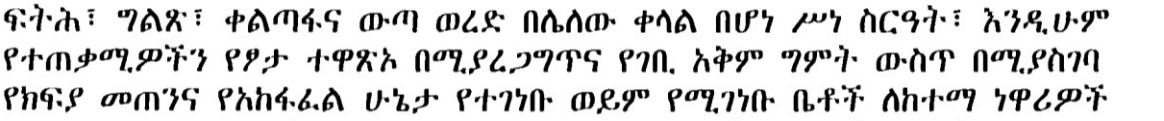
Original image



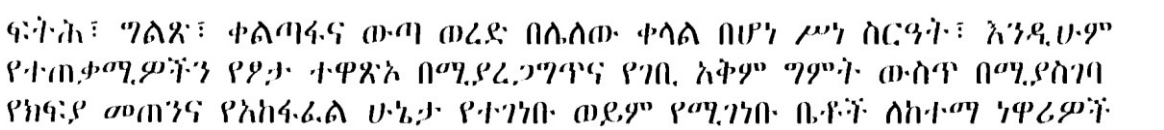
Median Filtered and Niblack Thresholded



Median Filtered and Otsu Thresholded

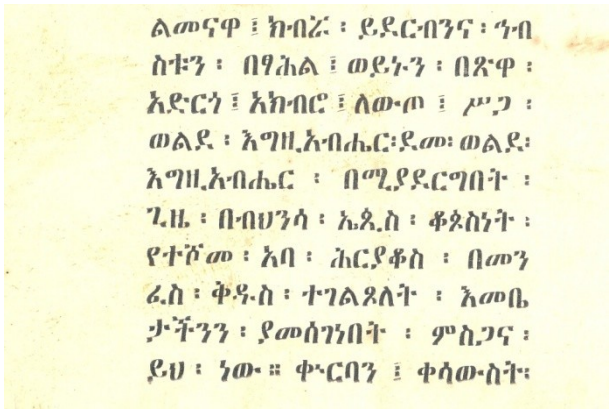


Median Filtered and Sauvola Thresholded

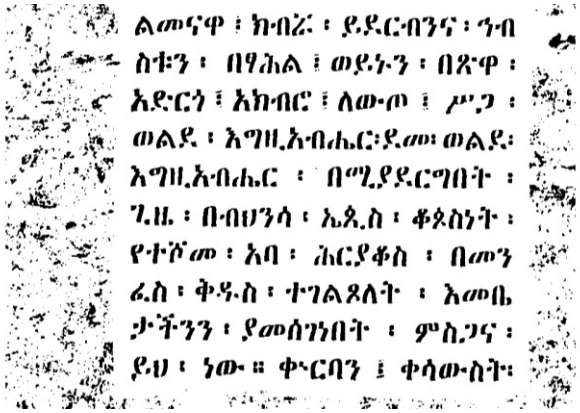


Medium Level Noise

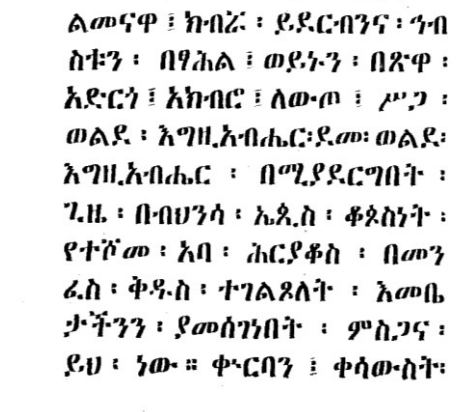
Original image



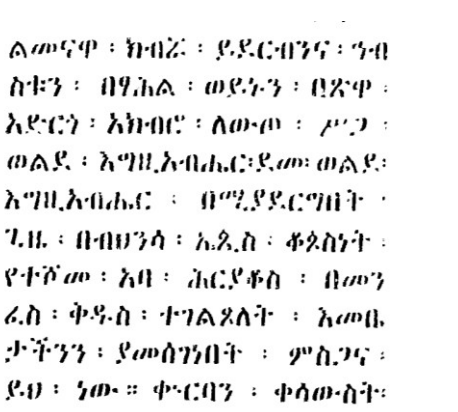
Median Filtered and Niblack Thresholded



Median Filtered and Otsu Thresholded

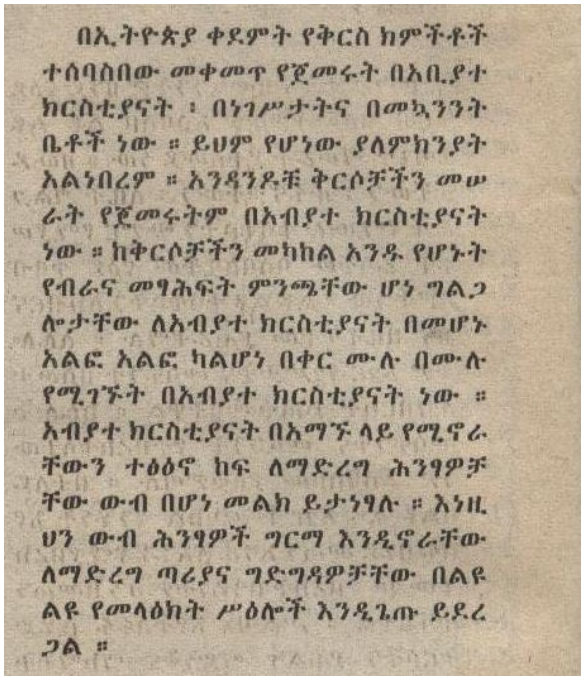


Median Filtered and Sauvola Thresholded



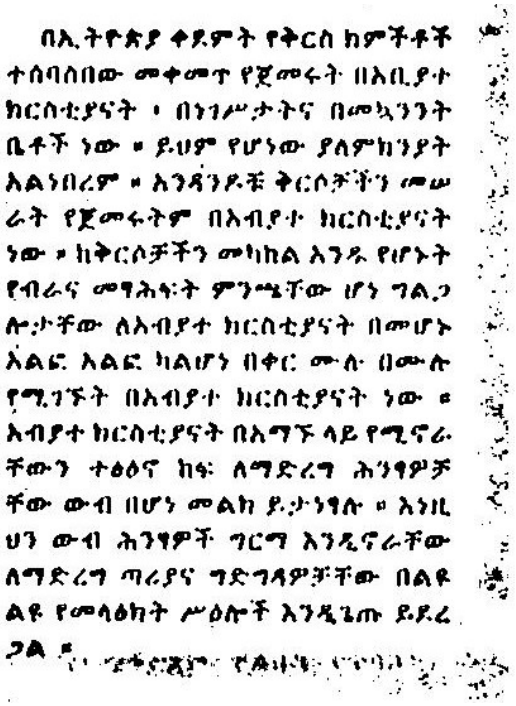
High Level Noise

Original image



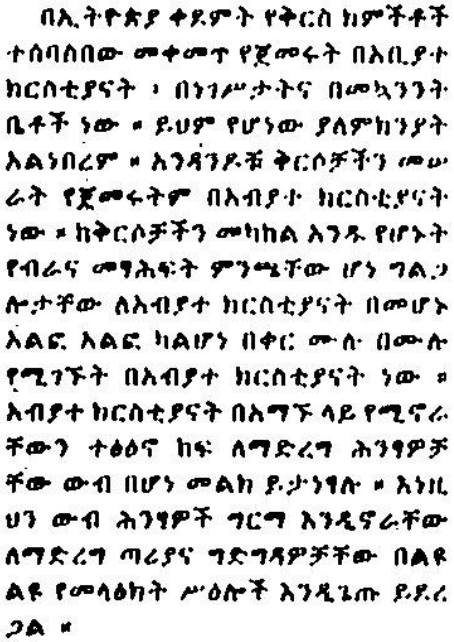
በኢትዮጵያ ቀደምት የቅርስ ክምችቶች ተሰባስበው መቀመጥ የጀመሩት በአቢያተ ክርስቲያናት ፡ በነገሥታትና በመኳንንት ቤቶች ነው ። ይህም የሆነው ያለምክንያት አልነበረም ። አንዳንዶቹ ቅርሶቻችን መሠራት የጀመሩትም በአብያተ ክርስቲያናት ነው ። ከቅርሶቻችን መካከል አንዱ የሆኑት የብራና መፃሕፍት ምንጫቸው ሆነ ግልጋሎታቸው ለአብያተ ክርስቲያናት በመሆኑ አልፎ አልፎ ካልሆነ በቀር ሙሉ በሙሉ የሚገኙት በአብያተ ክርስቲያናት ነው ። አብያተ ክርስቲያናት በአማኙ ላይ የሚኖራቸውን ተፅዕኖ ከፍ ለማድረግ ሕንፃዎቻቸው ውብ በሆነ መልክ ይታነፃሉ ። እነዚህን ውብ ሕንፃዎች ግርማ እንዲኖራቸው ለማድረግ ጣሪያና ግድግዳዎቻቸው በልዩ ልዩ የመላዕክት ሥዕሎች እንዲጌጡ ይደረጋል ።

Median Filtered and Niblack Thresholded



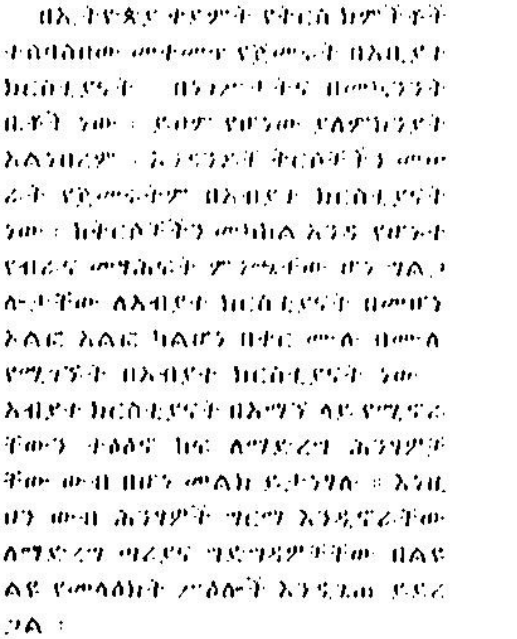
በኢትዮጵያ ቀደምት የቅርስ ክምችቶች ተሰባስበው መቀመጥ የጀመሩት በአቢያተ ክርስቲያናት ፡ በነገሥታትና በመኳንንት ቤቶች ነው ። ይህም የሆነው ያለምክንያት አልነበረም ። አንዳንዶቹ ቅርሶቻችን መሠራት የጀመሩትም በአብያተ ክርስቲያናት ነው ። ከቅርሶቻችን መካከል አንዱ የሆኑት የብራና መፃሕፍት ምንጫቸው ሆነ ግልጋሎታቸው ለአብያተ ክርስቲያናት በመሆኑ አልፎ አልፎ ካልሆነ በቀር ሙሉ በሙሉ የሚገኙት በአብያተ ክርስቲያናት ነው ። አብያተ ክርስቲያናት በአማኙ ላይ የሚኖራቸውን ተፅዕኖ ከፍ ለማድረግ ሕንፃዎቻቸው ውብ በሆነ መልክ ይታነፃሉ ። እነዚህን ውብ ሕንፃዎች ግርማ እንዲኖራቸው ለማድረግ ጣሪያና ግድግዳዎቻቸው በልዩ ልዩ የመላዕክት ሥዕሎች እንዲጌጡ ይደረጋል ።

Median Filtered and Otsu Thresholded



በኢትዮጵያ ቀደምት የቅርስ ክምችቶች ተሰባስበው መቀመጥ የጀመሩት በአቢያተ ክርስቲያናት ፡ በነገሥታትና በመኳንንት ቤቶች ነው ። ይህም የሆነው ያለምክንያት አልነበረም ። አንዳንዶቹ ቅርሶቻችን መሠራት የጀመሩትም በአብያተ ክርስቲያናት ነው ። ከቅርሶቻችን መካከል አንዱ የሆኑት የብራና መፃሕፍት ምንጫቸው ሆነ ግልጋሎታቸው ለአብያተ ክርስቲያናት በመሆኑ አልፎ አልፎ ካልሆነ በቀር ሙሉ በሙሉ የሚገኙት በአብያተ ክርስቲያናት ነው ። አብያተ ክርስቲያናት በአማኙ ላይ የሚኖራቸውን ተፅዕኖ ከፍ ለማድረግ ሕንፃዎቻቸው ውብ በሆነ መልክ ይታነፃሉ ። እነዚህን ውብ ሕንፃዎች ግርማ እንዲኖራቸው ለማድረግ ጣሪያና ግድግዳዎቻቸው በልዩ ልዩ የመላዕክት ሥዕሎች እንዲጌጡ ይደረጋል ።

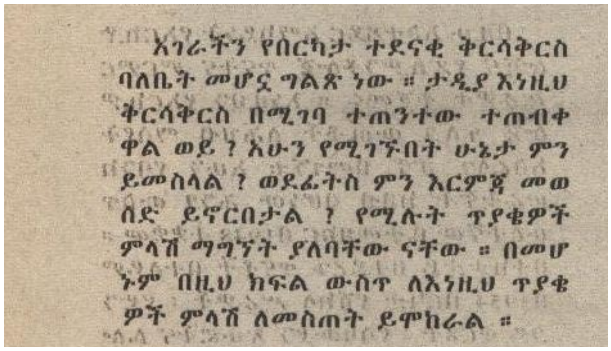
Median Filtered and Sauvola Thresholded



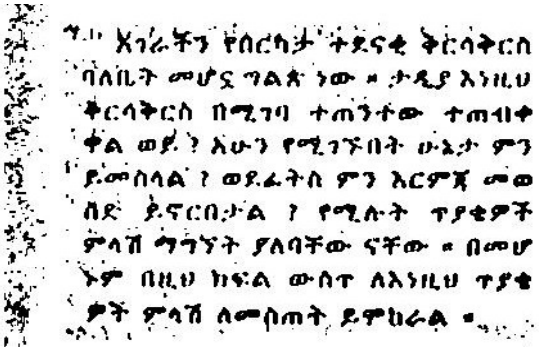
በኢትዮጵያ ቀደምት የቅርስ ክምችቶች ተሰባስበው መቀመጥ የጀመሩት በአቢያተ ክርስቲያናት ፡ በነገሥታትና በመኳንንት ቤቶች ነው ። ይህም የሆነው ያለምክንያት አልነበረም ። አንዳንዶቹ ቅርሶቻችን መሠራት የጀመሩትም በአብያተ ክርስቲያናት ነው ። ከቅርሶቻችን መካከል አንዱ የሆኑት የብራና መፃሕፍት ምንጫቸው ሆነ ግልጋሎታቸው ለአብያተ ክርስቲያናት በመሆኑ አልፎ አልፎ ካልሆነ በቀር ሙሉ በሙሉ የሚገኙት በአብያተ ክርስቲያናት ነው ። አብያተ ክርስቲያናት በአማኙ ላይ የሚኖራቸውን ተፅዕኖ ከፍ ለማድረግ ሕንፃዎቻቸው ውብ በሆነ መልክ ይታነፃሉ ። እነዚህን ውብ ሕንፃዎች ግርማ እንዲኖራቸው ለማድረግ ጣሪያና ግድግዳዎቻቸው በልዩ ልዩ የመላዕክት ሥዕሎች እንዲጌጡ ይደረጋል ።

Very High Level Noise

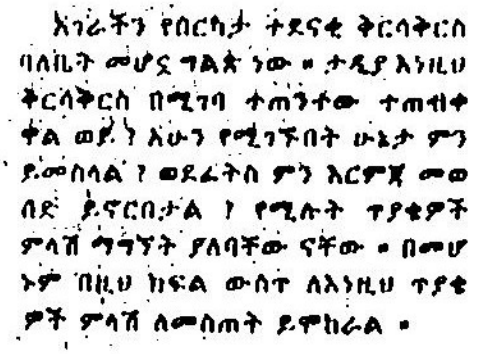
Original image



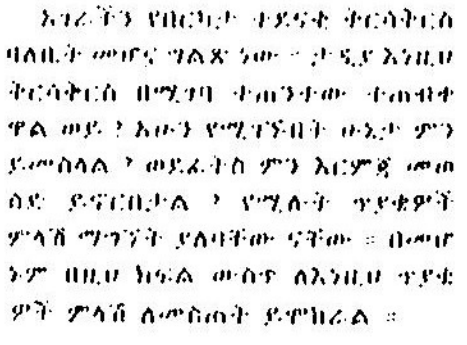
Median Filtered and Niblack Thresholded



Median Filtered and Otsu Thresholded



Median Filtered and Sauvola Thresholded



==========

APPENDIX II: Experiment 2 Source Codes and Sample Document Images

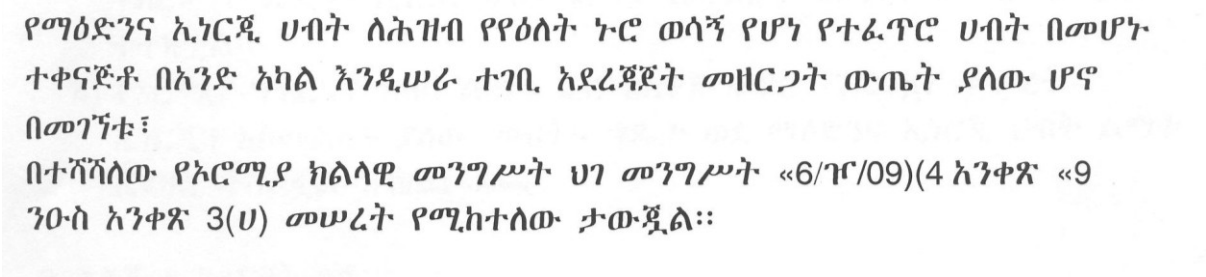
```
% This is the second experimentation
% AMF filtering is used for noise reduction and
% Niblack, Otsu and Sauvola thresholdings is used for binarization
% MSE is Mean Square Error
% PSNR is used to measure the peak signal to noise ratio
function ExAMF
    % Clear the command window
    clc
    % Close all figures opened
    close all
    % Read the image by calling ImageReader function and passing the file name
    [OriginalImage, GrayImage] =
ImageReader('G:\FinalImageCorpus\LowLevelNoise\Awaj4.jpg');
    % Apply wiener filter by calling WienerFilter function and passing the gray
    % image file and the dimension
    FilteredImage = AdaptiveMedianFilter(GrayImage, 7);
    % Call NiblackThresholding method by passing the Median Filtered Image
    ThresholdedImage = NiblackThresholding(FilteredImage);
    % Call OtsuThresholding method by passing the Median Filtered Image
    ThresholdedImage = OtsuThresholding(FilteredImage);
    % Call SauvolaThresholding method by passing the Median Filtered Image
    % Display the images
    ImageDisplayer(OriginalImage, 'Original Noisy Image');
    ImageDisplayer(GrayImage, 'Gray Image');
    ImageDisplayer(FilteredImage,'Filtered Image');
```

```
ImageDisplayer(ThresholdedImage,'Thresholded Image');
% Calculate MSE (Mean Square Error) and PSNR (Peak Signal to Noise Ratio)
% by calling the function calc_MSE_PSNR.
% The first value returned is MSE and trhe second is PSNR
 [MSE, PSNR] = calc_MSE_PSNR(GrayImage, ThresholdedImage)
% Display MSE
disp(['The MSE is: ', MSE]);
% Display PSNR
disp(['The PSNR is: ', PSNR]);
% Write the thresholded image to a jpg file
report = ImageWriter(ThresholdedImage,
'G:\FinalImageCorpus\LowLevelNoise\EX_AN_Awaj4.jpg');
disp(report);
```

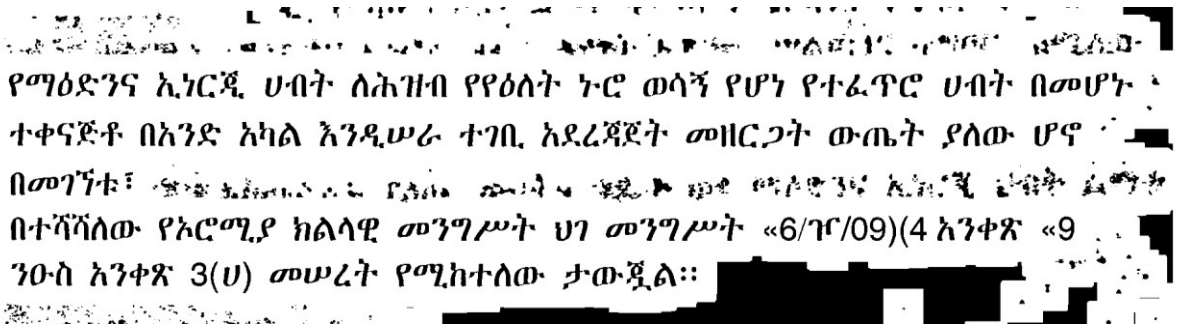
==========

Low Level Noise

Original image



AMF Filtered and Niblack Thresholded



AMF Filtered and Otsu Thresholded

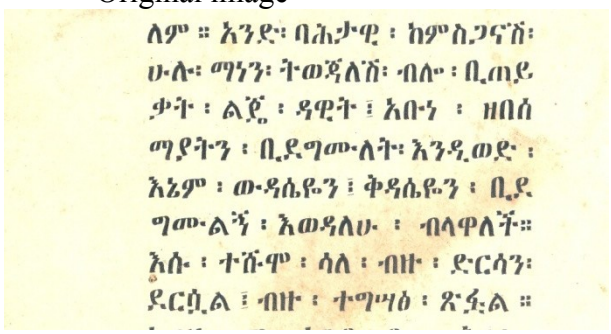
የማዕድንና ኢነርጂ ሀብት ለሕዝብ የየዕለት ኑሮ ወሳኝ የሆነ የተፈጥሮ ሀብት በመሆኑ ተቀናጅቶ በአንድ አካል እንዲሠራ ተገቢ አደረጃጀት መዘርጋት ውጤት ያለው ሆኖ በመገኘቱ፣
በተሻሻለው የኦሮሚያ ክልላዊ መንግሥት ህገ መንግሥት «6/ገ/09)(4 አንቀጽ «9 ንዑስ አንቀጽ 3(ሀ) መሠረት የሚከተለው ታውጇል።

AMF Filtered and Sauvola Thresholded

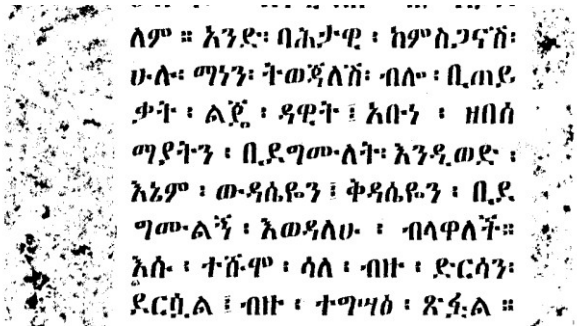
የማዕድንና ኢነርጂ ሀብት ለሕዝብ የየዕለት ኑሮ ወሳኝ የሆነ የተፈጥሮ ሀብት በመሆኑ ተቀናጅቶ በአንድ አካል እንዲሠራ ተገቢ አደረጃጀት መዘርጋት ውጤት ያለው ሆኖ በመገኘቱ፣
በተሻሻለው የኦሮሚያ ክልላዊ መንግሥት ህገ መንግሥት «6/ገ/09)(4 አንቀጽ «9 ንዑስ አንቀጽ 3(ሀ) መሠረት የሚከተለው ታውጇል።

Medium Level Noise

Original image



AMF Filtered and Niblack Thresholded



AMF Filtered and Otsu Thresholded

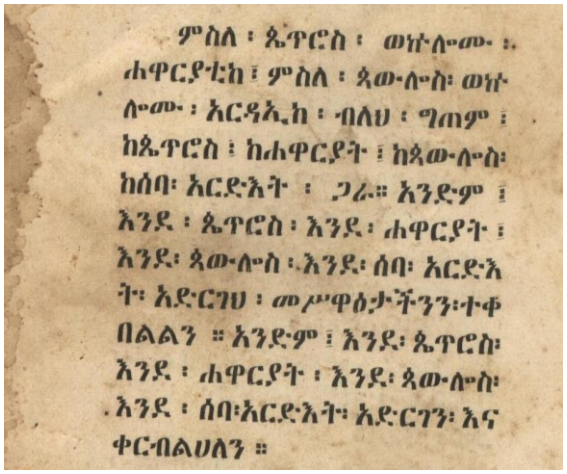
ለም ። አንድ፡ ባሕታዊ ፡ ከምስጋናሽ፡
ሁሉ፡ ማነን፡ ትወጃለሽ፡ ብሎ ፡ ቢጠይ
ቃት ፡ ልጇ ፡ ዳዊት ፡ አቡነ ፡ ዘበሰ
ማያትን ፡ ቢደግሙለት፡ እንዲወድ ፡
እኔም ፡ ውዳሴዬን ፡ ቅዳሴዬን ፡ ቢደ
ግሙልኝ ፡ እወዳለሁ ፡ ብላዋለች።
እሱ ፡ ተሹሞ ፡ ሳለ ፡ ብዙ ፡ ድርሳን፡
ደርሷል ፡ ብዙ ፡ ተግሣፅ ፡ ጽፏል ።

AMF Filtered and Sauvola Thresholded

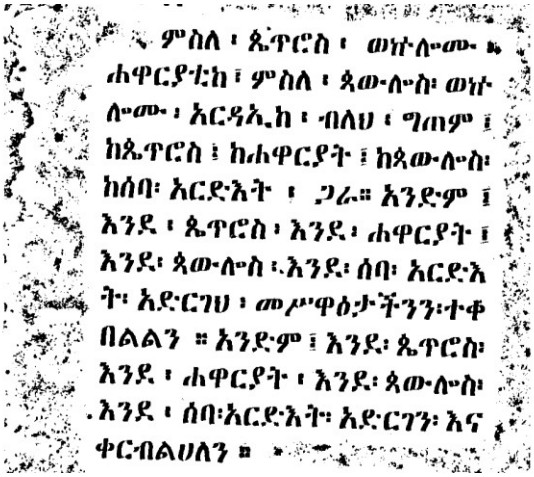
ለም ። አንድ፡ ባሕታዊ ፡ ከምስጋናሽ፡
ሁሉ፡ ማነን፡ ትወጃለሽ፡ ብሎ ፡ ቢጠይ
ቃት ፡ ልጇ ፡ ዳዊት ፡ አቡነ ፡ ዘበሰ
ማያትን ፡ ቢደግሙለት፡ እንዲወድ ፡
እኔም ፡ ውዳሴዬን ፡ ቅዳሴዬን ፡ ቢደ
ግሙልኝ ፡ እወዳለሁ ፡ ብላዋለች።
እሱ ፡ ተሹሞ ፡ ሳለ ፡ ብዙ ፡ ድርሳን፡
ደርሷል ፡ ብዙ ፡ ተግሣፅ ፡ ጽፏል ።

High Level Noise

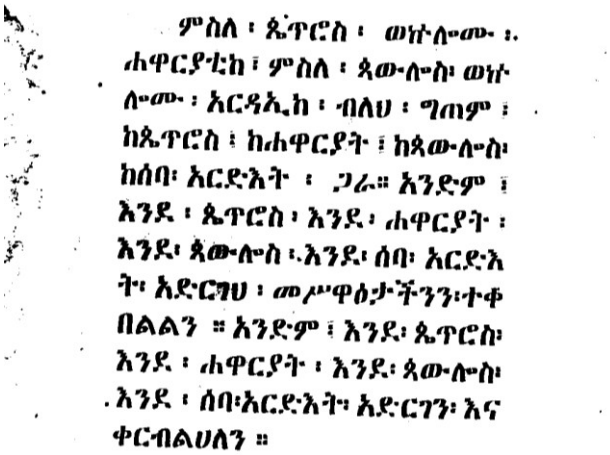
Original image



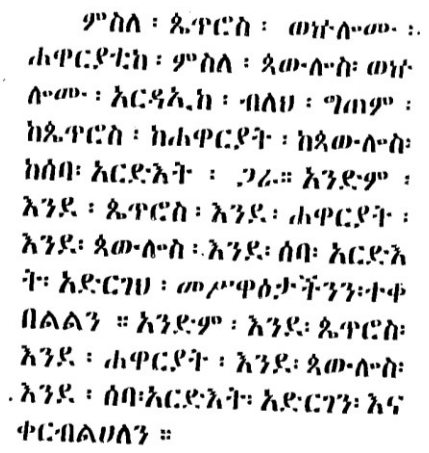
AMF Filtered and Niblack Thresholded



AMF Filtered and Otsu Thresholded

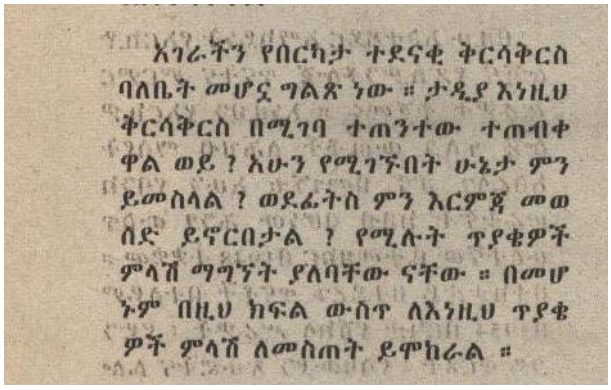


AMF Filtered and Sauvola Thresholded

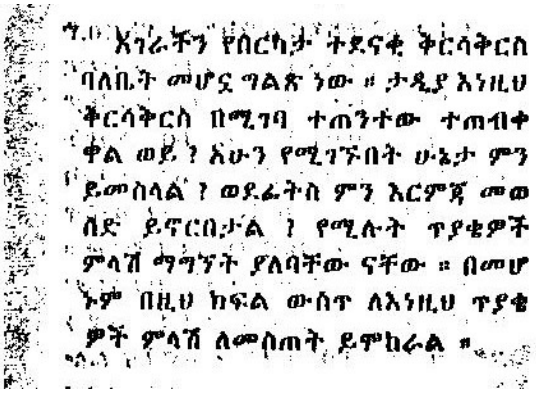


Very High Level Noise

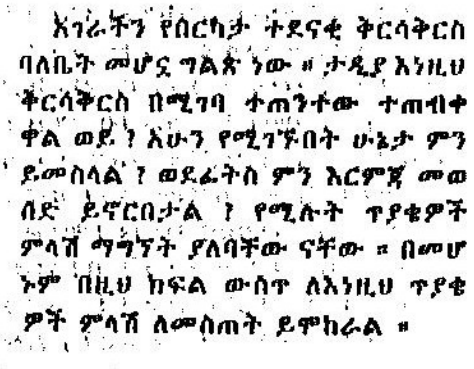
Original image



AMF Filtered and Niblack Thresholded

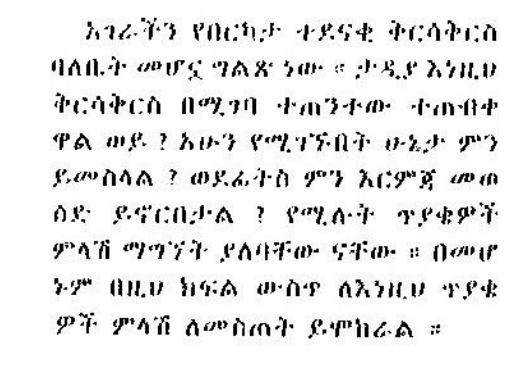


AMF Filtered and Otsu Thresholded



አገራችን የበርካታ ተደናቂ ቅርሳቅርስ ባለቤት መሆኗ ግልጽ ነው ። ታዲያ እነዚህ ቅርሳቅርስ በሚገባ ተጠንተው ተጠብቀ ቀል ወይ ? አሁን የሚገኙበት ሁኔታ ምን ይመስላል ? ወደፊትስ ምን እርምጃ መወ ሰድ ይኖርበታል ? የሚሉት ጥያቄዎች ምላሽ ማግኘት ያለባቸው ናቸው ። በመሆ ኑም በዚህ ክፍል ውስጥ ለእነዚህ ጥያቄ ዎች ምላሽ ለመስጠት ይሞክራል ።

AMF Filtered and Sauvola Thresholded



አገራችን የበርካታ ተደናቂ ቅርሳቅርስ ባለቤት መሆኗ ግልጽ ነው ። ታዲያ እነዚህ ቅርሳቅርስ በሚገባ ተጠንተው ተጠብቀ ቀል ወይ ? አሁን የሚገኙበት ሁኔታ ምን ይመስላል ? ወደፊትስ ምን እርምጃ መወ ሰድ ይኖርበታል ? የሚሉት ጥያቄዎች ምላሽ ማግኘት ያለባቸው ናቸው ። በመሆ ኑም በዚህ ክፍል ውስጥ ለእነዚህ ጥያቄ ዎች ምላሽ ለመስጠት ይሞክራል ።

APPENDIX III: Experiment 3 Source Codes and Sample Document Images

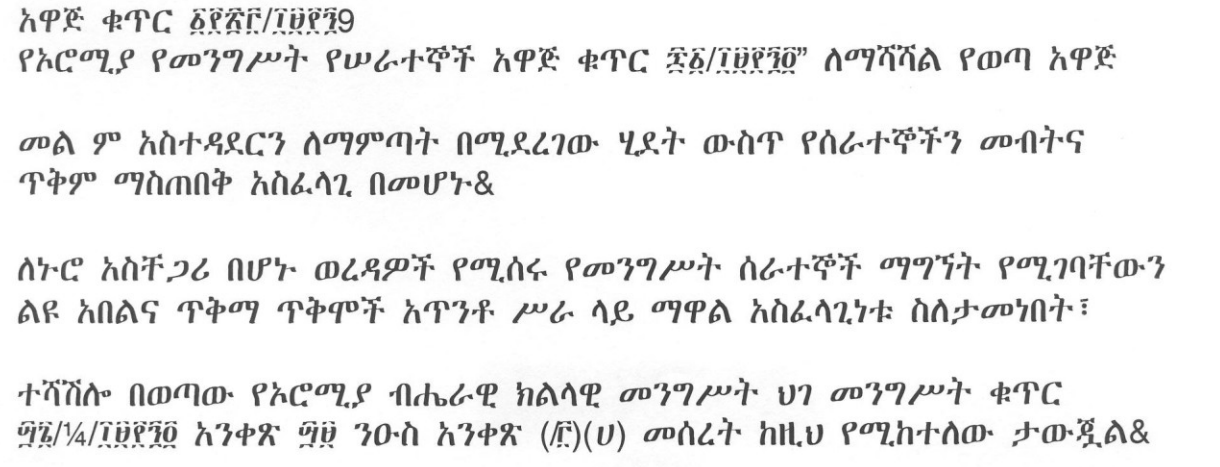
```
% This is the third experimentation
% wiener filtering is used for noise reduction and
% Niblack, Otsu and Sauvola thresholdinsg is used for binarization
% MSE calculates Mean Square Error
% PSNR is used to measure the peak signal to noise ratio
function ExWiener
    % Clear the command window
    clc
    % Close all figures opened
    close all
    % Read the image by calling ImageReader function and passing the file name
    [OriginalImage, GrayImage] =
    ImageReader('G:\FinalImageCorpus\VeryHighLevelNoise\Newspaper3.jpg');
    % Apply wiener filter by calling WienerFilter function and passing the gray
    % image file and the dimension
    FilteredImage = WienerFilter(GrayImage, 3);
    % Call NiblackThresholding method by passing the Median Filtered Image
    ThresholdedImage = NiblackThresholding(FilteredImage);
    % Call OtsuThresholding method by passing the Median Filtered Image
    ThresholdedImage = OtsuThresholding(FilteredImage);
    % Call SauvolaThresholding method by passing the Median Filtered Image
    ThresholdedImage = SauvolaThresholding(FilteredImage);
    % Display the images
    ImageDisplayer(OriginalImage, 'Original Noisy Image');
    ImageDisplayer(GrayImage, 'Gray Image');
```

```
ImageDisplayer(FilteredImage,'Filtered Image');

ImageDisplayer(ThresholdedImage,'Thresholded Image');

% Calculate PSNR (Peak Signal to Noise Ratio) by calling the function calcPSNR.

PSNR = calcPSNR(GrayImage, FilteredImage)

% Write the thresholded image to a jpg file

report = ImageWriter(ThresholdedImage,
'G:\FinalImageCorpus\VeryHighLevelNoise\EX_WN_Newspaper3.jpg');

disp(report);
```

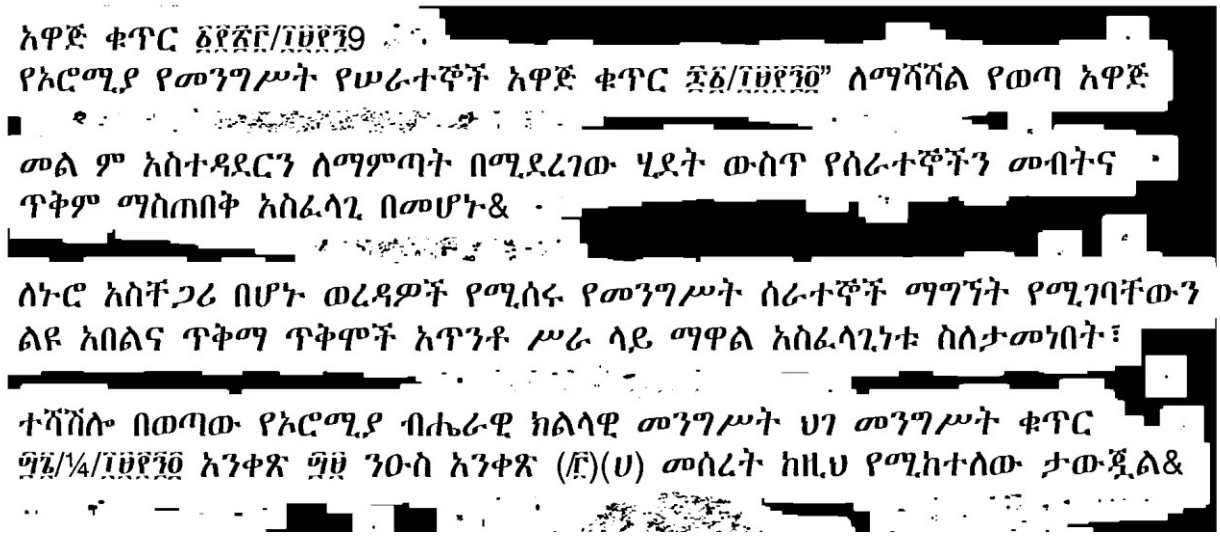
==========

Low Level Noise

Original image



Wiener Filtered and Niblack Thresholded



Wiener Filtered and Otsu Thresholded

አዋጅ ቁጥር ፩፻፸፫/፲፱፻፺9
የኦሮሚያ የመንግሥት የሠራተኞች አዋጅ ቁጥር ፰፩/፲፱፻፺፬" ለማሻሻል የወጣ አዋጅ

መል ም አስተዳደርን ለማምጣት በሚደረገው ሂደት ውስጥ የሰራተኞችን መብትና
ጥቅም ማስጠበቅ አስፈላጊ በመሆኑ&

ለኑሮ አስቸጋሪ በሆኑ ወረዳዎች የሚሰሩ የመንግሥት ሰራተኞች ማግኘት የሚገባቸውን
ልዩ አበልና ጥቅማ ጥቅሞች አጥንቶ ሥራ ላይ ማዋል አስፈላጊነቱ ስለታመነበት፣

ተሻሽሎ በወጣው የኦሮሚያ ብሔራዊ ክልላዊ መንግሥት ህገ መንግሥት ቁጥር
፵፮/¼/፲፱፻፺፬ አንቀጽ ፵፱ ንዑስ አንቀጽ (ፎ)(ሀ) መሰረት ከዚህ የሚከተለው ታውጇል&

Wiener Filtered and Sauvola Thresholded

አዋጅ ቁጥር ፩፻፸፫/፲፱፻፺፬
የኦሮሚያ የመንግሥት የሠራተኞች አዋጅ ቁጥር ፰፩/፲፱፻፺፬" ለማሻሻል የወጣ አዋጅ

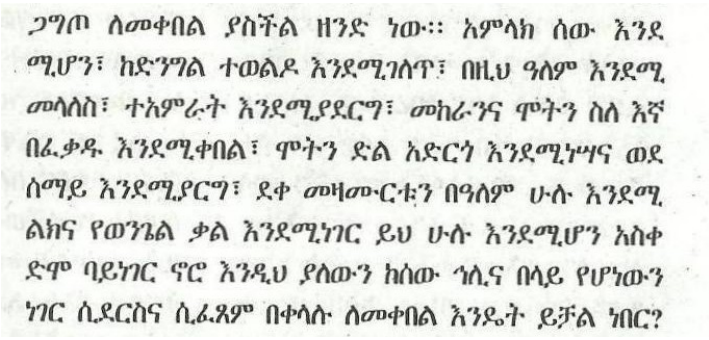
መል ም አስተዳደርን ለማምጣት በሚደረገው ሂደት ውስጥ የሰራተኞችን መብትና
ጥቅም ማስጠበቅ አስፈላጊ በመሆኑ&

ለኑሮ አስቸጋሪ በሆኑ ወረዳዎች የሚሰሩ የመንግሥት ሰራተኞች ማግኘት የሚገባቸውን
ልዩ አበልና ጥቅማ ጥቅሞች አጥንቶ ሥራ ላይ ማዋል አስፈላጊነቱ ስለታመነበት፣

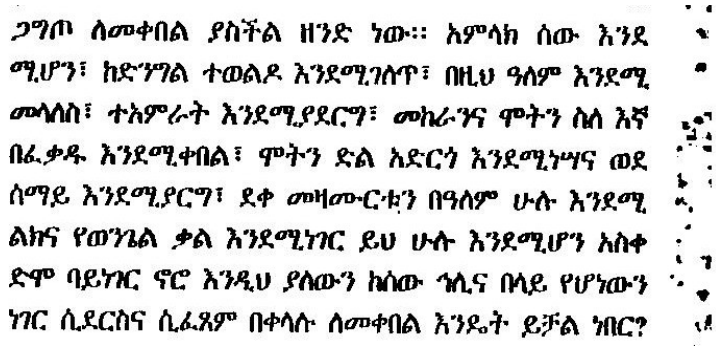
ተሻሽሎ በወጣው የኦሮሚያ ብሔራዊ ክልላዊ መንግሥት ህገ መንግሥት ቁጥር
፵፮/¼/፲፱፻፺፬ አንቀጽ ፵፱ ንዑስ አንቀጽ (ፎ)(ሀ) መሰረት ከዚህ የሚከተለው ታውጇል&

Medium Level Noise:

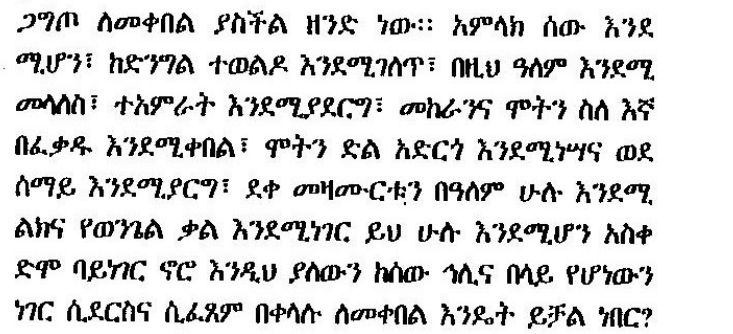
Original image


ጋግጦ ለመቀበል ያስችል ዘንድ ነው። አምላክ ሰው እንደ
ሚሆን፣ ከድንግል ተወልዶ እንደሚገለጥ፣ በዚህ ዓለም እንደሚ
መላለስ፣ ተአምራት እንደሚያደርግ፣ መከራንና ሞትን ስለ እኛ
በፈቃዱ እንደሚቀበል፣ ሞትን ድል አድርጎ እንደሚነሣና ወደ
ሰማይ እንደሚያርግ፣ ደቀ መዛሙርቱን በዓለም ሁሉ እንደሚ
ልክና የወንጌል ቃል እንደሚነገር ይህ ሁሉ እንደሚሆን አስቀ
ድሞ ባይነገር ኖሮ እንዲህ ያለውን ከሰው ጎሊና በላይ የሆነውን
ነገር ሲደርስና ሲፈጸም በቀላሉ ለመቀበል እንዴት ይቻል ነበር?

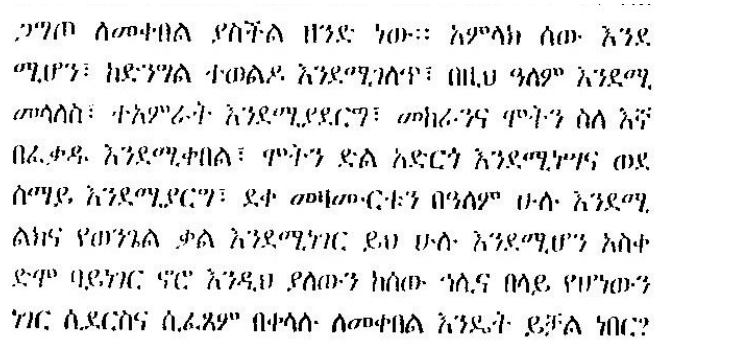
Wiener Filtered and Niblack Thresholded


ጋግጦ ለመቀበል ያስችል ዘንድ ነው። አምላክ ሰው እንደ
ሚሆን፣ ከድንግል ተወልዶ እንደሚገለጥ፣ በዚህ ዓለም እንደሚ
መላለስ፣ ተአምራት እንደሚያደርግ፣ መከራንና ሞትን ስለ እኛ
በፈቃዱ እንደሚቀበል፣ ሞትን ድል አድርጎ እንደሚነሣና ወደ
ሰማይ እንደሚያርግ፣ ደቀ መዛሙርቱን በዓለም ሁሉ እንደሚ
ልክና የወንጌል ቃል እንደሚነገር ይህ ሁሉ እንደሚሆን አስቀ
ድሞ ባይነገር ኖሮ እንዲህ ያለውን ከሰው ጎሊና በላይ የሆነውን
ነገር ሲደርስና ሲፈጸም በቀላሉ ለመቀበል እንዴት ይቻል ነበር?

Wiener Filtered and Otsu Thresholded

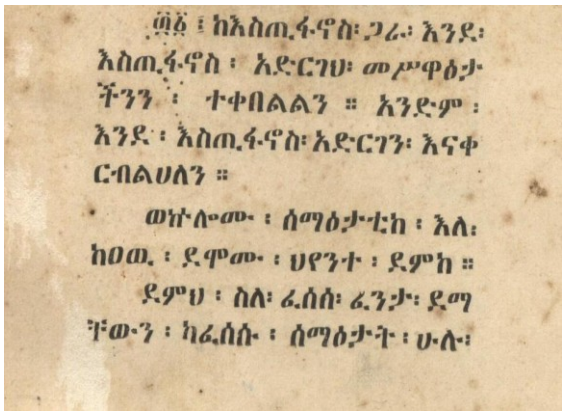

ጋግጦ ለመቀበል ያስችል ዘንድ ነው። አምላክ ሰው እንደ
ሚሆን፣ ከድንግል ተወልዶ እንደሚገለጥ፣ በዚህ ዓለም እንደሚ
መላለስ፣ ተአምራት እንደሚያደርግ፣ መከራንና ሞትን ስለ እኛ
በፈቃዱ እንደሚቀበል፣ ሞትን ድል አድርጎ እንደሚነሣና ወደ
ሰማይ እንደሚያርግ፣ ደቀ መዛሙርቱን በዓለም ሁሉ እንደሚ
ልክና የወንጌል ቃል እንደሚነገር ይህ ሁሉ እንደሚሆን አስቀ
ድሞ ባይነገር ኖሮ እንዲህ ያለውን ከሰው ጎሊና በላይ የሆነውን
ነገር ሲደርስና ሲፈጸም በቀላሉ ለመቀበል እንዴት ይቻል ነበር?

Wiener Filtered and Sauvola Thresholded

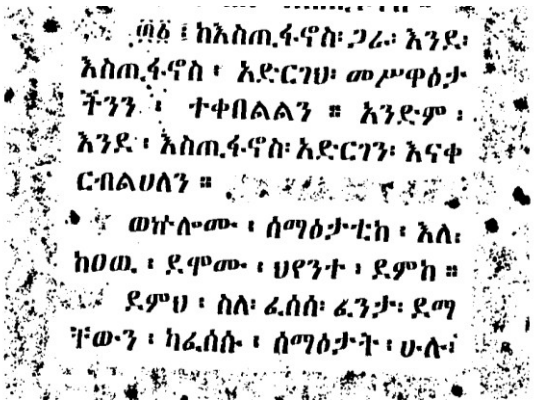

ጋግጦ ለመቀበል ያስችል ዘንድ ነው። አምላክ ሰው እንደ
ሚሆን፣ ከድንግል ተወልዶ እንደሚገለጥ፣ በዚህ ዓለም እንደሚ
መላለስ፣ ተአምራት እንደሚያደርግ፣ መከራንና ሞትን ስለ እኛ
በፈቃዱ እንደሚቀበል፣ ሞትን ድል አድርጎ እንደሚነሣና ወደ
ሰማይ እንደሚያርግ፣ ደቀ መዛሙርቱን በዓለም ሁሉ እንደሚ
ልክና የወንጌል ቃል እንደሚነገር ይህ ሁሉ እንደሚሆን አስቀ
ድሞ ባይነገር ኖሮ እንዲህ ያለውን ከሰው ጎሊና በላይ የሆነውን
ነገር ሲደርስና ሲፈጸም በቀላሉ ለመቀበል እንዴት ይቻል ነበር?

High Level Noise

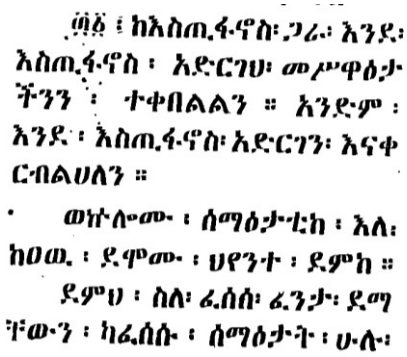
Original image


ፀጋ ፣ ከእስጢፋኖስ ጋራ እንደ
እስጢፋኖስ ፣ አድርገህ መሥዋዕታ
ችንን ፣ ተቀበልልን ። አንድም ፣
እንደ ፣ እስጢፋኖስ አድርገን እናቀ
ርብልሀለን ።
ወኵሉሙ ፣ ሰማዕታቲከ ፣ እለ
ከዐዉ ፣ ደሞሙ ፣ ህየንተ ፣ ደምከ ።
ደምህ ፣ ስለ ፈሰሰ ፈንታ ደማ
ቸውን ፣ ካፈሰሱ ፣ ሰማዕታት ፣ ሁሉ

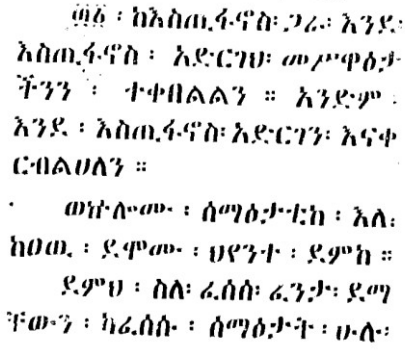
Wiener Filtered and Niblack Thresholded


ፀጋ ፣ ከእስጢፋኖስ ጋራ እንደ
እስጢፋኖስ ፣ አድርገህ መሥዋዕታ
ችንን ፣ ተቀበልልን ። አንድም ፣
እንደ ፣ እስጢፋኖስ አድርገን እናቀ
ርብልሀለን ።
ወኵሉሙ ፣ ሰማዕታቲከ ፣ እለ
ከዐዉ ፣ ደሞሙ ፣ ህየንተ ፣ ደምከ ።
ደምህ ፣ ስለ ፈሰሰ ፈንታ ደማ
ቸውን ፣ ካፈሰሱ ፣ ሰማዕታት ፣ ሁሉ

Wiener Filtered and Otsu Thresholded

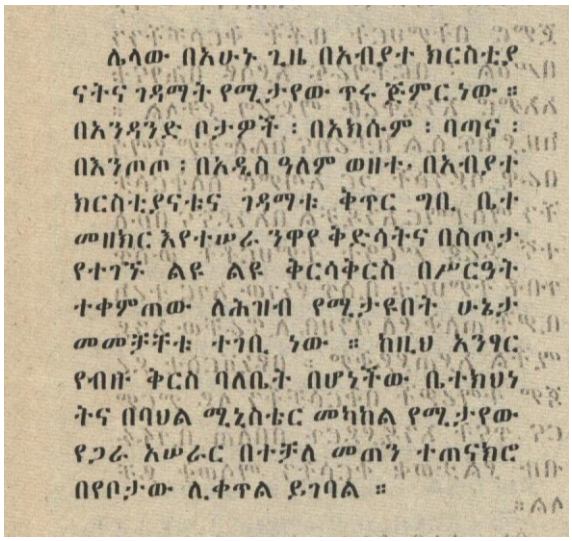


Wiener and Sauvola

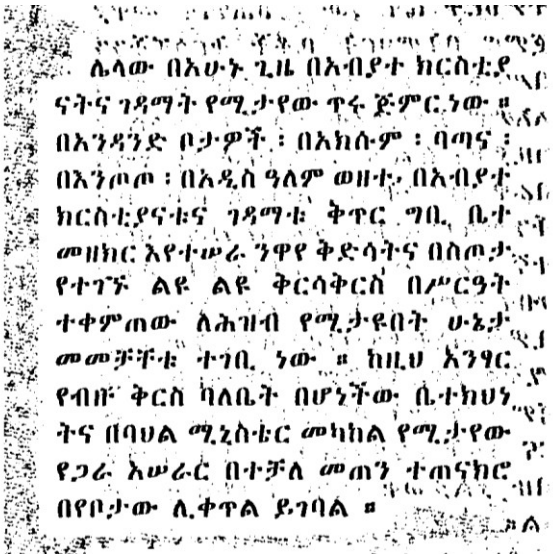


Very High Level Noise

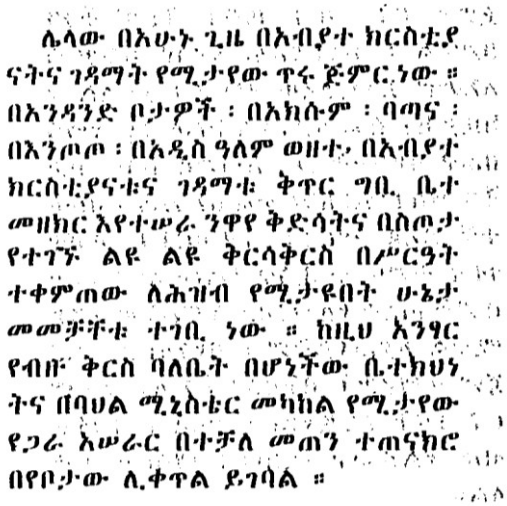
Original image



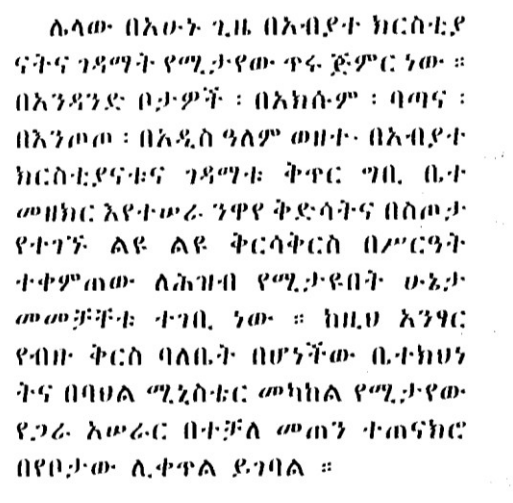
Wiener Filtered and Niblack Thresholded



Wiener Filtered and Otsu Thresholded



Wiener and Sauvola



APPENDIX IV: Multiple Word Querying Method Developed

(TextRendering.java)

```
private void jButtonWordActionPerformed(java.awt.event.ActionEvent evt)
{
      TextRend = jTextFieldWord.getText().trim();
      int ind = TextRend.indexOf(" ");
      String query1 = TextRend.substring(0, ind);
      String query2 = TextRend.substring(ind + 1, TextRend.length());

      // COMBINED MULTIPLE QUERY WORD
      Vector <String> returnedDocs = new Vector <String>(2, 2);
      Vector <String> returnedDocs2 = new Vector <String>(2, 2);
      Vector <String> FinalRankedDocs = new Vector <String>(2, 2);
      String docsForQuery1[] = new String[1];
      String docsForQuery2[] = new String[1];

      // Separater of directory and file name
      Pattern pat = Pattern.compile(",");

      /** Process Query 1 */
      File fileRendQuery1 = new File("query1.jpg");
      TextToimage Query1Convert = new TextToimage();
      Query1Convert.textRend(query1, fileRendQuery1);
      File imageQuery1 = fileRendQuery1;
      String imageNameQuery1 = imageQuery1.getPath();
      PlanarImage piQ1 = JAI.create("fileload", imageNameQuery1);
      SampleModel smQ1 = piQ1.getSampleModel();
      int widthQ1 = piQ1.getWidth();
      int heightQ1 = piQ1.getHeight();
      AllQueryMethods.HorbordQ(piQ1, hQ, heightQ1, hQ, widthQ1, pixelaverageQ, smQ1,
         pwwbQ, hordataStart, hordataEnd, nbands, nbandsQ, pwwvQ, imageNameQuery1);
      returnedDocs = AllQueryMethods.MatchingAlgorithms(widthQ1, heightQ1, nbands,
         nbands, widthQ1, heightQ1, nbands, nbands, pwwbQ);
      // Initialize docsForQuery1
      docsForQuery1 = new String[returnedDocs.size()];
      // Splitting returnedDocs
      for(int x = 0; x < returnedDocs.size(); x++)
      {
          String extractedFileName[] = pat.split(returnedDocs.elementAt(x));
```

```
    docsForQuery1[x] = extractedFileName[1].toString();
 }

/** Process Query 2 */
File fileRendQuery2 = new File("query2.jpg");
TextToimage Query2Convert = new TextToimage();
Query2Convert.textRend(query2, new File("query2.jpg"));
File imageQuery2 = fileRendQuery2;
String imageNameQuery2 = imageQuery2.getPath();
PlanarImage piQ2 = JAI.create("fileload", imageNameQuery2);
SampleModel smQ2 = piQ2.getSampleModel();
int widthQ2 = piQ2.getWidth();
int heightQ2 = piQ2.getHeight();
AllQueryMethods.HorbordQ(piQ2, hQ, heightQ2, hQ, widthQ2, pixelaverageQ, smQ2,
   pwwbQ, hordataStart, hordataEnd, nbands, nbandsQ, pwwvQ, imageNameQuery2);
returnedDocs2 = AllQueryMethods.MatchingAlgorithms(widthQ2, heightQ2, nbands,
   nbands, widthQ2, heightQ2, nbands, nbands, pwwbQ);

 // Initialize docsForQuery2
docsForQuery2 = new String[returnedDocs2.size()];
 // Splitting returnedDocs2
 for(int x = 0; x < returnedDocs2.size(); x++)
 {
   String extractedFileName[] = pat.split(returnedDocs2.elementAt(x));
   docsForQuery2[x] = extractedFileName[1].toString();
 }
System.out.println("Searching is Finished");

// Populate FinalRankedDocs with common docs that contain all query terms
for(int i = 0; i < docsForQuery1.length; i++)
{
   for(int j = 0; j < docsForQuery2.length; j++)
   {
      if(docsForQuery1[i].equals(docsForQuery2[j]))
      {
         if(FinalRankedDocs.contains(docsForQuery1[i]) == false)
         {
            FinalRankedDocs.add(docsForQuery1[i]);
         }
         break;
```

```
            }
        }
    }

    // Then, add docs that contain only one of the query terms
    // Populate it with docs that contain Query1 only
    for(int i = 0; i < docsForQuery1.length; i++)
    {
        if(FinalRankedDocs.contains(docsForQuery1[i]) == false)
        {
            FinalRankedDocs.add(docsForQuery1[i]);
        }
    }

    // Then, add docs that contain only one of the query terms
    // Populate it with docs that contain Query2 only
    for(int j = 0; j < docsForQuery2.length; j++)
    {
        if(FinalRankedDocs.contains(docsForQuery2[j]) == false)
        {
            FinalRankedDocs.add(docsForQuery2[j]);
        }
    }
    System.out.println("Final Ranked Docs!");
    for(int k = 0; k < FinalRankedDocs.size(); k++)
    {
        System.out.println(FinalRankedDocs.elementAt(k));
    }
}
```

APPENDIX V: Additional Methods Developed

==========

1. Processing Multiple Files at once and Display the Total MSE & PSNR

```
% Final working code
% Wiener filtering is used for noise reduction and Otsu thresholding is used for binarization
% MSE = Mean Square Error and PSNR = Peak Signal to Noise Ratio

function ExWienerOtsu
    % Clear the command window
        clc
    % Close all figures opened
        close all
    % Initialization
        totalMSE = 0;
        totalPSNR = 0;
        NumberOfFiles = 5
    % Iterate through the files
        for i=1:NumberOfFiles
                directory = 'G:\0Final Dataset\HistoricalDocumentImages\';
                file = strcat('HDI',int2str(i),'.jpg');
                df = strcat(directory, file);
                ndf = strcat(directory,'EX_WO_',file);
        % Read the image by calling ImageReader function and passing the file name
                [OriginalImage, GrayImage] = ImageReader(df);
        % Apply wiener filter by calling WienerFilter function and passing the gray
        % image file and the dimension
                FilteredImage = WienerFilter(GrayImage, 3);
        % Call OtsuThresholding method by passing the Median Filtered Image
                ThresholdedImage = OtsuThresholding(FilteredImage);
        % Display the images
                ImageDisplayer(OriginalImage, 'Original Noisy Image');
                ImageDisplayer(GrayImage, 'Gray Image');
```

```
            ImageDisplayer(FilteredImage,'Filtered Image');
            ImageDisplayer(ThresholdedImage,'Thresholded Image');
        % Calculate MSE and PSNR by calling the function calc_MSE_PSNR.
        % The first value returned is MSE and the second is PSNR
            [MSE, PSNR] = calc_MSE_PSNR(GrayImage, ThresholdedImage);
                totalMSE = totalMSE + MSE;
                totalPSNR = totalPSNR + PSNR;
            disp(['The MSE is: ', MSE]);
            disp(['The PSNR is: ', PSNR]);
        % Write the thresholded image to a jpg file
            report = ImageWriter(ThresholdedImage, ndf);
            disp(report);
        end
        avgMSE = totalMSE/NumberOfFiles;
        disp(avgMSE);
        avgPSNR = totalPSNR/NumberOfFiles;
        disp(avgPSNR);
```

==========

2. Image Writer Function

```
function report = ImageWriter(image, location)
        % Write an image to a file
        imwrite(image, location);
        report = 'The image is written to a file and saved Successfully.';
```

==========

3. Image Displayer Function

```
function ImageDisplayer(image, imageTitle)
        % Display the image with the specified title
        figure, imshow(image), title(imageTitle);
```

Declaration

I, the undersigned, declare that the thesis is my original work and has not been presented for a degree in any other university, and that all source materials used for this thesis have been duly acknowledge.

_____

Biniam Asnake

Date: 20th June 2012

This thesis has been submitted for examination with my approval as university advisor.

_____

Million Meshesha (Ph.D)

Date: 20th June 2012