Addis Ababa University

College of Natural Science

School of Information Science

Preprocessing of Mobile Captured Document Images

Alula Melaku

October 2016

A Thesis submitted to school of Information Science of Addis Ababa University in partial fulfillment of the requirements for the Degree of Master of Science in Information Science

Addis Ababa, Ethiopia

Addis Ababa University
College of Natural
Sciences School of Information Science

Preprocessing of Mobile Captured Document Images

By

Alula Melaku

Name and signature of members of examining Board:

Advisor:	Dr. Dereje Teferi (Ph.D)	_____	_____
	Name	Signature	Date
Examiner:	Million Meshesha (Ph.D)	_____	_____
	Name	Signature	Date
Examiner:	Wondwossen Mulugeta (Ph.D)	_____	_____
	Name	Signature	Date

Declaration

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

_____

Alula Melaku

Date: 17th October 2016

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

_____

Dereje Teferi (Ph.D)

Date: 17th October 2016

Acknowledgment

With the help of God this has become a reality. Therefore, my innumerable praise first goes to God Almighty for his guiding me all the way.

I would like to express my gratitude to my advisor, Dereje Teferi (Ph.D), for his support, constructive comments and suggestion throughout the thesis work.

My deepest gratitude also goes to my friends Biniyam, Berhanu, Mulualem, Andish and Cheilo for their encouragement and support in different aspects throughout my study.

Finally, my special thanks go to my family especially to my Dad, in support and encourage me all the time.

Abstract

In the last years, mobile phones cameras have high resolution cameras. Cell phone cameras are convenient image acquisition devices: they are fast, versatile, mobile, and do not touch the object. OCR of a captured business card or captured image is a process that allows a printed text from the card/image to be recognized optically and converted into machine-readable code. The recognized text can be used for further use for example from a business card that contain a personal information can be used to save it on contact list in the mobile device.

In OCR applications, however, cell phone cameras suffer from a number of limitations, like blur, lighting condition, alignment, and geometrical distortions. Moreover, the effectiveness of the system is highly dependent on the preprocessing techniques used. This study explores an effective preprocessing method that handles the noise created during the capturing process and the document noise. We use a SONY Xperia M2 mobile phone to capture the business card and we implemented the preprocessing techniques (skew correction, perspective correction, noise removal, binarization, and text region extraction) in MATLAB image processing tool.

In this study, we only deal with the preprocessing step of the character recognition for a mobile captured business card. We applied a preprocessing techniques like skew detection and correction, perspective rectification, noise removal, image binarization and text region extraction on a test data set that are collected by the researcher. Experiment on different Skew and perspective correction methods and based document boundary correction is selected, this technique is supported by the user by putting the four vertex of the document image. From the three noise removal techniques, based on the experiment, Wiener filter with 1.92 MSE and 48.99 PSNR and from the three binarization techniques, Sauvola with 0.13 MSE and 57.62 PSNR found to perform best with the highest PSNR and lower MSE. And for text region extraction a modified Connected Component and Dilation method is used. The proposed approach after preprocessing detects the text region with 93.60% Precision and 99.99% Recall. The challenges in the study are detecting correctly the text region and non-text region, in some cases the text regions are detected as non-text and vice versa, if the document captured at dark environment, and if the captured document is misaligned these needs further study.

Table of Contents

List of Figures

List of Tables

Abbreviations

ANN: Artificial Neural Network

ACM: Active Contour Model

CC: Connected Component

DPI: Dots per Inch

HPP Horizontal Projection Profile

MSE Mean Squared Error

OCR Optical Character Recognition

PSNR Peak Signal to Noise Ratio

RGB Red Green Blue

RBF Radial Basis Function

SVM Support Vector Machine

TTS Text To Speech

VPP Vertical Projection Profile

Chapter One

Introduction

1.1 Background

As demand grows for mobile phone, research in optical character recognition, a technology well developed for scanned documents, is shifting focus to the recognition of text embedded in digital photographs. Mobile cameras are convenient image acquisition device because of their size, portability and cheap.

Mobile phones are one of the most commonly used electronic devices today. Commodity mobile phones with powerful microprocessors (above 500MHz), high resolution cameras (above 2megapixels), and a variety of embedded sensors (accelerometers, compass, GPS) are widely deployed and becoming ubiquitous. By fully exploiting these advantages, mobile phones are becoming powerful portable computing platforms, and therefore can process computing-intensive programs in real time [9].

Nowadays, with the increase in computer and mobile use in everybody's life, the ability for people to convert printed documents to digitally readable format has become a necessity. Scanning and capturing documents is a way of changing printed documents into digital format [3] [1]. Printing text on paper is done every day; on some occasions however the reverse is needed, getting the original text back from a scanned image or photograph, for further editing and use. This process is called OCR. Optical Character Recognition (OCR) is a process that allows printed text to be recognized optically and converted in to machine-readable code that can be accepted by a computer for further processing.

Computing under handheld devices involves a number of challenges. Because of the non-contact nature of digital cameras attached to handheld devices, acquired images very often suffer from skew and perspective distortion. In addition to that, manual involvement in the capturing process, uneven and insufficient illumination, and unavailability of sophisticated focusing system yield poor quality images [2].

Until a few decades ago, research in the field of Optical Character Recognition (OCR) was limited to document images acquired with flatbed desktop scanners. The usability of such systems is limited as scanners are not portable because of large size of the scanners and the need of a computing system [2]. There are plenty of situations where you need an easy way to capture printed text and convert it to an easy electronically readable file; At a meeting, if you want to record notes from the whiteboard, you want to save important points from a conference poster session without lugging around a ton of paper handouts, browsing a magazine, you come across an article you'd love to refer to later, or saving someone's business card information. To address all the above issues we need to capture the images using a mobile device such as cell phones. This paper discuss how to handle the challenges of mobile captured image, show what type of technique are to be used so as to get a digitally readable information.

Wojciech et al. [5] made a couple of conclusions were drawn from their experiment. The OCR process was not much hampered by uneven lighting. Noise was much more detrimental, but for 10% amount of noise and high resolution (600 dpi) the accuracy still remained viable. The median filter appeared totally inappropriate for their scenario, resulting in further accuracy loss. The FineReader accuracy dramatically drops for image resolution below 300 dpi. Nonetheless, OCR recognition was most sensitive to geometric deformations. These types of artifacts make it difficult to trace text lines, which may imply, that further OCR processing of an image rotated by e.g. 10 degree totally fails. Unlike noise and low resolution obstacles, geometric deformations may be eliminated by preprocessing the original image.

Amharic OCR systems are developed for desktop scanners, since 1997 by Worku Alemu [19], a few research are done in this area. To mention a few Dereje Teferi [12], Million Meshesha [11], Yaregal Assabie [8], Abay Teshager [26], Biniyam Asnake [25], and Birhanu Sahle [13] in 2015 have worked on Amharic OCR. Amharic is the working language of Ethiopia. Bulks of printed Amharic texts have been circulating among governmental, nongovernmental and private sectors, including information centers, libraries, museums, etc. We also get an Amharic text in a whiteboard at a meeting, a conference poster, a magazine, a business card, and a notice board. Moreover there are countless historical, cultural, and other documents written using Amharic characters.

Some historical manuscripts are too fragile to touch we may need to use devices other than scanners such as cell phones [1] [8]. OCR includes some basic steps which are classified in different ways on various literatures to perform the recognition process. The general steps are scanning the document image, preprocessing (Skew correction, Binarization, Noise removal, slant detection, and removal), Segmentation, Character Recognition, and Post processing (Reconstruction). This steps are applied in there written order [2] [14].

Megan and Margaret [15] designed an image preprocessing suite on top of OCR engine to improve the accuracy of text recognition in natural scene images. They suggest that segmenting text and background in natural scenes is a difficult classification problem, and the accuracy of this segmentation is of extreme importance when the output of an OCR system will be transformed as in translation or speech synthesis.

Anand et al. [9] develops an application called PocketPal and PocketReader mobile which recognize a text. PocketPal is a personal receipt management tool installed in one's mobile phone. It helps users extract information from receipts and keep track of their shopping histories digitally in a mobile environment. PocketReader is a personal mobile screen reader that combines the OCR capability and a text-to-speech interface. PocketReader allows users to take pictures of any text source (magazines, newspapers, etc). It identifies and interprets the text contents and then reads them out.

As mentioned by Aparna and Chakravarthy [14] in their research, they use a radial Basis Function Neural Network method for finding the text regions in the document image for skew estimation. The network is trained to distinguish between text and non-text so as to work only on the text region and it is the main point to the system because if we don't exactly find the text region the rest of the process will fail.

Images captured by a cell phone camera differ from images scanned by a desktop scanner. They often have defects such as condition distortion at the edges and dimmed light, making it difficult for most OCR applications, to correctly recognize the text [10]. The processing speed and memory size of handheld devices are not yet sufficient enough so as to run desktop based OCR algorithms that are computationally expensive and require high amount of memory [2].

Business cards are small cards on which data is printed, typically holds business information about a company or individual. Business cards typically include a person's name, street (Location) addresses, e-mail address, phone number, fax number, website, company name, and usually a logo. They are shared during formal introduction as a convenience and a memory aid. In general business cards have texts, colors, images, and design. There are a lot researches done in other language for business card recognition but there is no research done on printed Amharic business card OCR system as to the knowledge of the researcher.

Due to the challenges of Amharic character effective OCR system is yet to be developed. Since 1997 there are a lot of researches undertaken but researchers face difficulty in recognizing the characters because of the large number of characters, similarities of shape and different fonts and styles [11][8].

Nowadays our activities are integrated with our mobile devices. This mobile devices portability and availability makes it more useable. An OCR system that uses a mobile camera for the input document image will make things easier to use.

Documents written in Amharic characters come in various font sizes, underline style, and have italics feature [13]. A common problem encountered in an OCR technology is noise when scanning documents which can occur in an image because of paper quality, the typing machine used, or it can be created by scanners during the scanning process. Among other things, noise reduces the accuracy of subsequent tasks of OCR systems [3]. The noise can appear in the foreground or background of an image and can be generated before or after scanning. The noise that will come before scanning is due to the quality of the paper and degradation of the paper, and the noise that will come after scanning is due to the scanner quality and type (desktop scanners, digital camera or mobile camera) [4] [6].

1.2 Statement of the problem

Noise might be introduced by scanning devices and transmission media, because of the many limitations of mobile camera scanning a detailed preprocessing of an image is required for the OCR before any process.

The usability of OCR systems in flatbed desktop scanners is limited as they are not portable because of large size of the scanners and the need of a computing system. Moreover, the digitization speed of a scanner is slower than that of a mobile camera [2]. Mobile cameras are convenient image acquisition devices: they are fast, versatile, mobile, do not touch the object, and are relatively cheap. In this research a mobile camera is used to scan the document image.

Real-life document images usually contain both text and non-text elements (images, graphics, logos, signatures, tables, seals, etc.). To recognize the text from the document we have to separate text regions from non-text elements. Business card specially contains more non-text element which has complicated graphics in the image.

An OCR application that is captured by a mobile camera suffer from a number of limitations, like geometrical distortions [5]. Camera-captured images can suffer from low resolution, blur, and perspective distortion, as well as complex layout and interaction of the content and background [1]. Due to the typical nature of cameras and the environment, it is not always possible to assure the high resolution and high quality images. Many discrepancies may arise due to Uneven lighting conditions, Skewness (Rotation), Tilting (perspective distortion), Blur, warping, and flooding [4][9].

An embedded camera inside the mobile phone has far less control of lighting conditions than scanners. Uneven lighting is common, due to both the physical environment (shadows, reflection, fluorescents) and uneven response from the devices. Further complications occur when trying to use artificial light, i.e. flash, which results in light flooding, the source of light is very much focused on a particular portion of the image, whereas the remaining portion is dark.

When OCR input is taken from a hand-held camera or other imaging devices whose perspective is not fixed like a scanner, text lines may get skewed from their original orientation. Feeding a rotated image to OCR engine produces extremely poor results.

Perspective distortion occurs when the text plane is not parallel to the imaging plane. It happens a lot if a hand-held camera is used to take pictures. The effect is that characters farther away look smaller and distorted, and parallel-line assumptions no longer hold in the image.

Text misalignment happens when the camera screen covers a partial text region, in which irregular shapes of the text characters are captured and imported as inputs to the OCR engine. Misalignment issue generally arises when people casually take their pictures. Misaligned images may lead to loss of important data.

Since many mobile cameras are designed to operate over a variety of distances, focus becomes a significant factor. Sharp edge response is required for the best character segmentation and recognition. At short distances and large apertures, even slight perspective changes can cause uneven focus.

Getting a text from captured document image is essential for further processing of the image like capturing a business card to save it on a mobile phone, capturing a vehicle license plate to get the vehicle plate text for searching the database about the vehicle, and capturing a schedule from a board and add it into the calendar. But a document captured using mobile devices suffer many challenges. An OCR that recognize a text from an image needs a clear image for accurate recognition, to recognize an image that are captured by a mobile device and digital camera have more noise than a desktop scanned image [5][7]. So a detail preprocessing has to be done for mobile captured images before sending to recognition engine. Different local study was made on preprocessing of a document image that are scanned by a desktop scanner like Biniam [25], Michael [22] and Birhanu [13]. They try to preprocess the image by noise removal and binarization and Birhanu [13] adds a skew detection mechanism. They did not consider correcting the perspective because when we scan a document by a desktop scanner this problem is not occurred. Although in removing the noise and binarization a mobile and camera captured document image needs further research.

To fill the above mentioned gaps, this study attempts to answer the following research questions:

- Which technique is effective for identifying the text and non-text region of the captured image?
- What suitable perspective rectification technique is applied for mobile captured image?
- Which image preprocessing techniques have to be used, to handle the challenges of mobile captured image and provides a better result?

Therefore, the purpose of this research is to apply preprocessing techniques to enhance the quality of Mobile captured Amharic document images.

1.3 Objective of the Study

1.3.1 General Objective

The main objective of this study is to explore and identify suitable preprocessing techniques for enhancing the quality of Mobile captured images and select the text area from the printed Amharic document images.

1.3.2 Specific Objectives

In order to meet the general objective, the following specific objectives are formulated

- To review previous researches and find out the specific challenges of recognizing mobile captured documents
- To find out and review the different characteristics of Amharic language fonts, types and orientation of the characters for text region extraction
- To collect training datasets from different business card documents of Amharic language that contains different level of degradations
- To select an appropriate algorithms and techniques for the proposed noise removal, skew and perspective correction, binarization and text region extraction that tackle the challenges of mobile captured Amharic document images
- To test and evaluate the performance of the proposed preprocessing techniques and report findings

1.4 Scope and Limitation of the Study

This study is about decreasing the errors of getting the text region of Amharic documents that are captured by cell phone. The present study provides a technique that is used to clear the noises and ready for recognition by using effective preprocessing techniques. The research select the best techniques by comparing each preprocess methods and see the result efficiency based on the correct text extraction we get. The researcher experiments on the correction of skewed and

perspective distorted image, removal of noise, binarization, and text region extraction techniques. The study is limited on these techniques because these techniques prevent the challenges or problems that are listed in the problem of statement which arise due to capturing a document by a mobile camera. The research is taken on the Amharic business card images that are captured by a cellphone device. Because of the many challenges in a mobile captured image it needs a specific image processing technique a skew and perspective correction for the geometric distortion, noise removal for the printed document degradation and other noises, binarization for blur and lighting condition, and text region extraction to separate a text from the image, logo, graphics, and other non-text parts. The study is undertake with a mobile camera with light environment and will not work on a white text with a black background. The study is only focus on a document captured by mobile cameras, a digital camera is not used under this study.

Because of time and tool constraints the experimentation of text region extraction is done only dilation and connected component analysis. Alternative algorithms like Edge based are not tested for the problem under investigation. Evaluation of the proposed system is performed with small number of dataset from low to high level noisy document and this might have its own limitation in measuring the exact performance of the system.

1.5 Methodology of the study

To undertake this research, the following techniques have been used.

1.5.1 Literature Review

Different literatures are reviewed from different sources like Research papers, different articles from the internet and books about OCR systems. In the last few years, mobile phones have evolved to devices with high resolution cameras and Internet connection. In this context, the idea of applying OCR techniques to the pictures taken with these devices rises [6].

1.5.2 Dataset Collection

In order to use the documents for training and testing, the researcher created a dataset that are captured by the researcher himself. The captured documents are Amharic Business card images with different level of noise and having different position. The dataset image collections have texts, logos, pictures, and columns. A total of 20 RGB business card images are used for the dataset that

are captured by the researcher. A mobile phone with 4.8 inch touch screen, 8 megapixel camera with a resolution of 229DPI is used for scanning.

1.5.3 Implementation Tool

To develop a prototype, a MATLAB® Image Processing ToolboxTM is used here because it has a good and better image processing technique. It has rich libraries for images processing such as noise removal (Median filer and Wiener filter) and binarization (Otsu) which minimizes the time needed for development, adopt some from previous literatures and develop the program for other by the researcher. Some algorithms of image processing are built-in MATLB function, so we don't need to write the code again.

1.5.4 Testing (performance evaluation)

Testing is done on the collected documents that are captured from different types of Amharic business card having different font styles and sizes. This research is only to show which preprocessing techniques has better performance for cell phone camera captured business card. The performance of proposed technique is evaluated with image quality assessments Mean Squared Error (MSE) and Peak Signal to Noise Ratio (PSNR), and performance measurement techniques Precision and Recall.

1.6 Significance of the Study

Digital cameras attached to cellular phones, PDAs, or wearable computers, and standalone image or video devices are highly mobile and easy to use; they can capture images of thick books, historical manuscripts too fragile to touch, and text in scenes, making them much more versatile than desktop scanners [1]. Because of the mobility of cell phone cameras we can capture any document found anywhere, we can capture a business card and save the content to the contact of the phone, can capture lecture notes from the notice board and get the text, can capture text from the newspaper to refer it latter.

Preprocessing of an image that is captured by mobile camera is critical for the OCR to get an attractive result from the system. Due to the number of image quality limitations in the mobile captured document, for recognition of text from this image need a detailed preprocessing to get good output. Correcting the geometric distortion before recognition increases the performance of

getting the correct text region from the image. Handling the blur that arise due to the motion of movement during the capturing process also increases the performance of getting the correct text region from the image.

OCR technology allows to automatically recognizing characters through an optical mechanism. In case of human beings, our eyes are optical mechanism. The image seen by eyes is input for brain. The ability to understand these inputs varies in each person according to many factors. OCR is a technology that functions like human ability of reading [10]. Nowadays almost every activity is done in a computer. This OCR technology makes it easy for document modification and reuse, and to further process the document image. By using this technology we edit the desired text by converting the document image into texts so that we only change the desired one, so we save our time and resource.

The OCR takes image as the input, gets text from that image, and then converts it into speech. It mainly designed for people who are unable to read any type of text documents [10]. By using the mobile device at any location they can capture the document image and with the help of OCR and TTS the user can get the speech of the text. An effective preprocessing technique has to be used to get a good result of a text region that can be used for the OCR.

1.7 Organization of the Study

This thesis is organized into five chapters. The first chapter discusses an introduction to the research area and stating the problems. It presents the objectives of the study, scope of the study, methodology of the study, significance of the study, and how we implement and test the result are discussed.

In chapter two, literature review on Optical Character Recognition, overview on the mobile captured document image processing and the Amharic writing system. The challenges of mobile captured and Amharic characters are discussed.

Chapter three describes the proposed image preprocessing techniques and algorithms. The different noise filtering, thresholding, skew correction, and perspective correction techniques are briefly discussed. The evaluation measures are presented.

Chapter four emphasizes the implementation and experimentation results that are used to confirm the validity of the proposed image preprocessing technique in the mobile camera captured business card.

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

Chapter Two

Literature Review

Optical Character Recognition (OCR) is becoming more usable and a wide research area in our current environment, that is because most of our everyday actions are done digitally. OCR is a process by which specialized software is used to convert scanned images of text to electronic text so that digitized texts can be searched, indexed, and retrieved [6]. These help we to get access to historical document that are found in museums and at different institutions by easily searching what we need from their content. It can be used to take pictures of any text source like magazines, newspapers, business cards, and posters and get the desired content. Present-day computers are electronic and digital, and thus they can only process data in digital format.

Character recognition may be classified into two types; online character recognition and offline character recognition [13]. Online character recognition deals with a data stream which comes from a transducer while the user is writing. The typical hardware to collect data is a digitizing tablet which is electromagnetic or pressure sensitive. When the user writes on the tablet, the successive movements of the pen are transformed to a series of electronic signal which is memorized and analyzed by the computer. The offline character recognition is performed after the writing is finished. The major difference between on-line and off-line character recognition is that on-line character recognition has time-sequence contextual information but off-line data does not. OCR deals with the recognition of characters acquired by optical means typically a scanner or a camera. The characters are in the form of pixelized images, and can be either printed or handwritten, of any size, shape, or orientation [28]. Digital cameras attached to cellular phones give us more usability in the character recognition area because the devices are highly mobile and easy to use and can capture different types of texts from different location. Some application of mobile camera captured OCR are for license plates, business cards, invoices, screenshots, ID cards, driver licenses, images on buildings, images on vehicles and other objects moving in a scene. This study discus in more detail in getting the content of captured business card.

In this Chapter, the general framework of the OCR system and the Amharic writing system discussed in detail.

2.1 Optical Character Recognition

Character recognition techniques associate a symbolic identity with the image of character. Character recognition is commonly referred to as optical character recognition (OCR), as it deals with the recognition of optically processed characters [32]. The modern version of OCR appeared in the middle of the 1940's with the development of the digital computers. OCR machines have been commercially available since the middle of the 1950's. The first true OCR reading machine was installed at Reader's Digest in 1954. This equipment was used to convert typewritten sales reports into punched cards for input to the computer [32]. Today there are many commercial OCR systems available in the world.

Every single day with so much of our lives computerized, it's vitally important that machines and humans can understand one another and pass information back and forth. We have to talk to computers through keyboard and mouse so they can figure out what we want them to do. As we read words from a computer or from a paper our eyes and brain are carrying out optical character recognition, our eyes are recognizing the patterns of light and dark that make up the characters (letters, numbers, and things like punctuation marks) and our brain figures out what it mean from the computer or the paper.

Vangie Beal states that "*Optical Character Recognition refers to the branch of computer science that involves reading text from paper and translating the images into a form that the computer can manipulate (for example, into ASCII codes). An OCR system enables you to take a book or a magazine article, feed it directly into an electronic computer file, and then edit the file using a word processor*" [50]. OCR is a technology that enables you to convert different types of documents such as scanned paper documents, PDF files, or images captured by a digital camera or desktop scanners into editable and searchable data. Images captured by a digital camera differ from scanned documents or image [10].

Optical Character Recognition may be defined as the electronic analysis of an image in order to identify the areas containing textual information and extracting/recognizing the text from the given image [4]. The major steps in OCR are Preprocessing (skew correction, Binarization, Noise

removal), Segmentation, Feature extraction, Character recognition and Post processing (Reconstruction of the document image) [14] [4].

The pre-processing step includes skew detection and correction, noise detection and removal, binarization, thinning which is an important approach to representing the shape of a plane region [5]. The objective of thinning is to reduce the representation of a region to a chain of single pixel width while preserving all other relevant features and normalization which is the process of enlarging and shrinking an image size. It scales the input image to a manageable size for the recognizer and for subsequent preprocessing stages. Segmentation is an operation that seeks to decompose an image into sub-images of individual symbols. There are basically two commonly used segmentation algorithms: stage by stage and recursive segmentation. In stage by stage algorithm a character is segmented in three steps: line segmentation, word segmentation, and character segmentation [13] [12]. Feature extraction followed for representing character image and classification module that label character to their proper class. These are essential components for character recognition process. Character recognition is a crucial component of an intelligent man-machine system. A variety of techniques have been proposed or implemented by researchers from worldwide. The most commonly used techniques are template matching, statistical classification, syntactic or structural matching, neural networks, and hybrid Approach. Post-processing is the last activity to occur in a series of OCR processing stages. Chiefly, the goal of post-processing is to detect and correct linguistic misspellings in the OCR output text after the input image has been scanned and completely processed. Figure 2.1 shows the general framework of an OCR System and the details are discussed below.

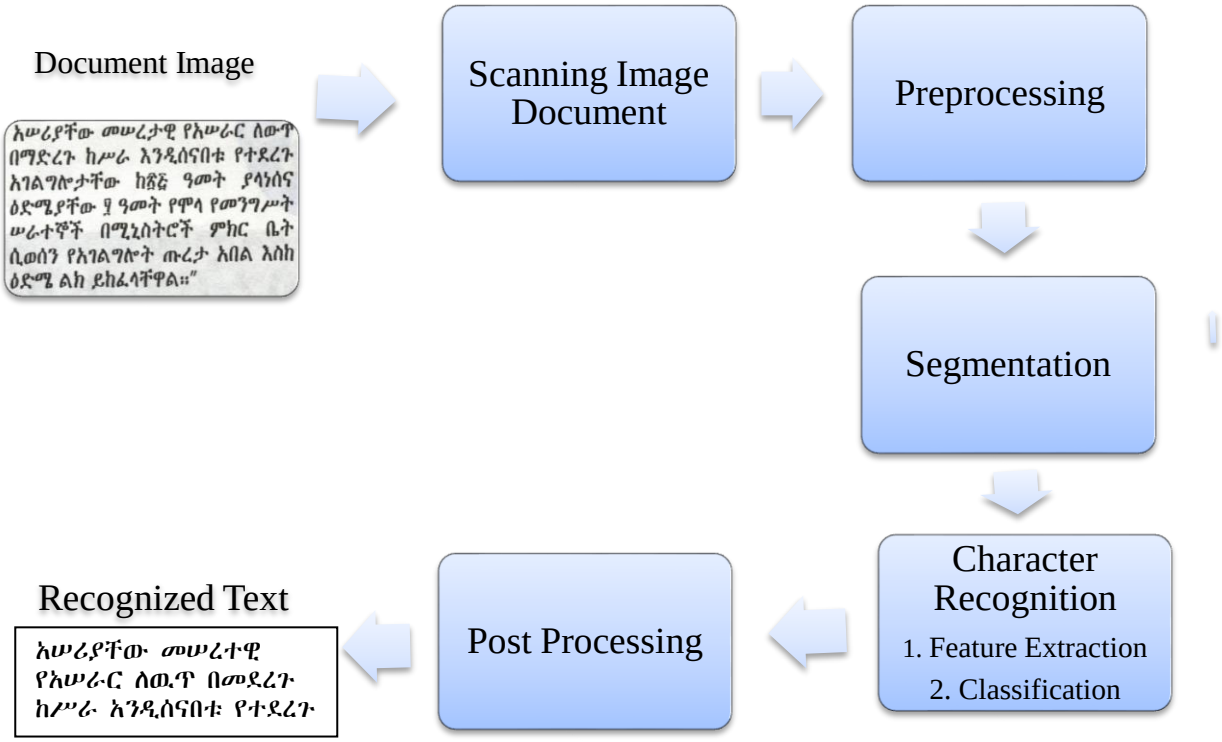


Figure 2.1 General framework of OCR

2.1.1 Scanning document images

Scanning an image is a process of converting the image into a digital format one can perform digitization through different devices such as desktop scanners, Digital cameras, mobile phone cameras, PDAs and tablet PCs [1][15]. Scanners have been developed for various imaging tasks, Scanners range from large-format drum scanners used for engineering drawings to small desktop devices used for casual scanning of business cards. The resolution of consumer-grade flatbed scanners has recently passed 2400 dpi (dots per inch), Such scanners are more than adequate for document image analysis given that 300 dpi or more is generally adequate for OCR [1][4]. 300dpi means for every square inch of paper, the scanner is capturing 300 dots horizontally and 300 dots vertically or 90,000 total dots. Scanners are devices that capture images from a photographic prints, posters, magazine pages, and similar sources for computer editing and displaying. Document scanners are categorized into to two distinct; that are portable document scanners and desktop document scanners. Portable document scanners include foldable document scanners, handheld scanners and pen scanners, all of them are compact, lightweight, and perfect for use on the go.

Examples of portable scanners are digital cameras, cell phone cameras, and Tablet PCs. Desktop document scanners are generally large in size, examples of desktop scanners are sheet-fed scanners and flatbed scanners.

	Desktop Scanners	Mobile cameras	Digital Camera
Resolution	150–600dpi	50–300dpi	50–600dpi
Object surface	Flat (almost always)	Can be arbitrary	Can be arbitrary
Distortion	Minimal for flat pages	Perspective and optical lens	Perspective and optical lens
Illumination	Even and adequate	Difficult to control	Difficult to control
Background	Domain dependent /known	Often complex	Often complex
Blur	Minimal	Motion and out-of-focus	Motion and out-of-focus
One-shot speed	Slow	Fast	Fast
Zoom / focus	Fixed	Variable	Variable
Portability	Bad	Good	Good
Size	Large	Compact	Compact
Misalignment	Low	High	High
Uneven Lighting Condition	Non	High	High

Table 2.1 Comparison of scanners and digital cameras (adopted from [1])

The comparison of scanning a document with a scanner and a digital or a mobile camera is shown in table 2.1. Offline data capturing is usually performed by optical scanning. And an optical image scanner is often used to convert the image of writing in to a bit pattern. The result will be stored in a file of picture elements called pixels. This phase is the stage where scanning device is used to scan the handwritten or printed documents in order to find the digitized form and the acquired images can be captured indifferent formats such as JPG, BMP, PNG, TIFF, etc. that range from a true color intake to binary form by adjusting the varying resolution of an image to make it in a way that a computer understands them as a matrix of pixels [12]. Therefore, we must note here that as a scanner resolution increases, the quality of document image also increases. Studies show that using high resolution is necessary while scanning real world historic documents which are very noisy and degraded. But increasing the resolution makes the captured digitalized image size big and this will result in the decrease in processing speed of OCR.

As Million [11] mentioned, there are many factors affecting OCR system. Some of them include: the mode of writing, the condition of the input page, the printing process, the quality of the paper, the presence of extraneous markings, the resolution, and quality of scanning. The quality of the scanner is the main and core input to the process, because if at the initial stage the scanned image quality is low the recognition accuracy of the system will be low. As mentioned above we are using a cell phone cameras for digitizing document that suffers from low resolution, blur, perspective distortion and complexity layout because of its capacity of resolution and capturing motion it need a good preprocessing methods in order to get a good recognition result.

2.1.2 Preprocessing

The raw digitized image is generally not suitable for recognition task and there is a need for it to be enhanced. Image noises are classified as physical noise, digitization noise, filtering noise and storage or transmission noise [25][38][42]. Physical noises are noises that are related with the physical readability or integrity of original information in the document and it is classified as internal noise that includes paper aging, paper texture, carbon copy effect, scratches, cracks and inadequate printing whereas external noises including folding marks, filing and staple punching, stain, thorn-off regions, worm holes, readers annotations, and highlighting, physical blur and sun burn. Digitization noises are introduced by the digitization process. Filtering noises are suitable manipulation of the digital file may degrade the information that exists in the digital format of the document instead of increasing it. Storage/Transmission noises are occurred due to the error or inefficiency in storage algorithms or from the network transmission errors [13][4].

As Atena et al. [3] mentioned an examples of noise in a scanned document images are page rule line, marginal noise, clutter noise and background noise. The rule line noise interferes with the object. The marginal noise usually appears in a large dark region around the document image and can be textual or non-textual. The clutter noise appear in an image because of document skew while scanning or are from holes punched in the document. The background noise, such as uneven contrast, show through effects, interfering strokes, and background spots.

Noises of a Mobile camera scanned document and some factors that affect the recognition accuracy are discussed below in detail [3][4][9][25][41]:

1. Ruled Line Noise

Handwritten documents are often written on pre-printed, lined paper. The lines can cause the ruled lines interfere with and connecting to the text; variable thicknesses in the ruled lines cause problems for the noise removal algorithms; broken ruled lines cause problems for algorithms detecting them; some letters, for example '𝐻', '𝑂', '𝑈' and '𝑖.', which have horizontal lines are removed by the algorithms as they are sometimes incapable of detecting differences between them and the ruled lines. One of the method have been proposed for ruled line removal in Atena et al. [3] is Projection Profile-based methods that work by creating a horizontal histogram in which the hills of the histogram are the center locations of the horizontal ruled lines. The Horizontal Projection Profile (HPP) is calculated by summing the white pixels in each row of the image. Projection profiles ignore the line's thickness, therefore, in the removal phase; the characters with horizontal strokes will be broken up. Another problem with this group of methods is sensitivity to rotation.

2. Marginal Noise

Marginal noises are dark shadows that appear in vertical or horizontal margins of an image. This type of noise is the result of scanning thick documents or the borders of pages in books; it can be textual or non-textual. Methods to remove marginal noise can be divided into two categories. The first category identifies and removes noisy components; the second focuses on identifying the actual content area or page frame of the document which it defines as the smallest rectangle that encloses all the foreground elements of the document image. Identifying and removing the noise component method employed vertical projection to recover document images that contain marginal noise, and decided whether the marginal noise was on the left or right side of the image based on the location of peaks in the profile. Then, by using extracted features, it detects the boundary between the shadows and cleans the area [24].

3. Clutter Noise

Clutter noise refers to unwanted foreground content which is typically larger than the text in binary images. This results from numerous sources such as punched holes, document image skew, or connecting huge amounts of pepper noise. The significant feature of clutter noise is that it is larger than the text objects in the document image. Clutter often touches or overlaps some parts of the text.

Maya et al. [7] proposes a method that is independent of clutter's position, size, shape and connectivity with text. A half residual image is achieved as the result of analysis of the distance transform, and by removing parts which are less than half of the maximum distance measured. The clutter element is then identified with a Support Vector Machine (SVM) classifier.

4. Background Noise

Historical manuscripts and scanned document images often have degradations like uneven contrast, show through effects, interfering strokes, background spots, humidity absorbed by paper in different areas, and uneven backgrounds. There are many methods in different literatures, as mentioned in Atena et al. [3] Binarization and Thresholding based methods which is one of the methods to enhance background quality of gray scale images. Local adaptive thresholding of Sauvolas is one example that removes the background noise. Although Otsu global thresholding also an example that removes the noise by dividing the image into foreground and background which is zeros and ones [3] [41].

5. Salt and Pepper Noise

Pepper noise can appear in a document image during the conversion process and is also caused by dirt on the document. This noise can be composed of one or more pixels but, by definition, they are assumed to be much smaller than the size of the text objects. Isolated pepper noise can be removed by simple filters like median but if they are larger than that, algorithms like k-fill or morphological operators will be more effective for noise removal. Salt noise looks like a lack of ink in the document image. If the fragmentation is very high, it reduces segmentation and recognition accuracy [25] [13]. It is reduced by performing filtering on the image where the background is "grown" into the noise specks, thus filling these holes. Median filter removes salt and pepper noise [3].

6. Blur

Motions blur and out of focus blur are the two types of blur in photography. These blur happen because of the variety of distance capturing and movement of hand [4] [5] [20]. The adopted method mentioned in [9] is the AutoFocus API provided by Android SDK to avoid any blurring seen in out-of-focus images. Whenever we start the application, a camera autofocus handler object is initiated so that the camera itself can focus on the text sources automatically. Wiener filter is

used to deblur the image that is occurred in the capturing process the motion. The method of correcting out of focus blur have not been addressed much in literatures. But motion blur can be eliminated by a noise removal method Wiener filter [4].

7. Tilting (Perspective Distortion)

Tilting happens because of scanning an image that is not parallel to the image. As a result tilted images are captured, wherein text closer to the camera appears bigger and that far away from the camera smaller [9] [4]. As Anand et al. [9] proposed to take advantage of the embedded orientation sensors to measure the tilt of the phone. It detects the tilt of the phone and disallows users to capture images when distortion occurs. It also allows users to calibrate plane of the document with that of the camera. This is however, a preventive technique rather than a corrective one.

Geetha and Muralif [49] proposes a method based on plane homography and transformation is used to make perspective correction. The algorithm infers frontier information directly from the images, without any reference objects or prior knowledge of the camera parameters. The frontiers are detected using geometric context based segmentation.

8. Uneven Lighting Condition

Uneven lighting is common in mobile capturing devices, due to both the physical environment (shadows, reflection, and fluorescents) and uneven response from the devices. Further problems arise when on-camera flash is used for the purpose [4] [9]. Binarization has long been recognized as a standard method to solve the lightning issue. Local adaptive thresholding techniques an example is Niblack's method which perform well and give excellent results even for severely degraded documents due to uneven lighting conditions. 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 [4].

9. Misalignment

Misalignment is the spatial property of texts that are not properly aligned where some text characters form the image text is cut. The mobile screen may only captures some part of the real document. The proposed approach in Anand et al. [9] is that a camera-captured image is misaligned if any of the four screen borders of the phone cuts through a single line of text, either horizontally, or vertically. Based on the definition, the authors set a 10-pixel wide margin along each border. If

any foreground pixel is detected within any of those 4 margins, there is high probability that the text is being cut and hence we can conclude that the image is misaligned, so we can capture the image again. The above proposed approach is a preventing the noise before happening because if this issue happen after the document captured we can't do anything to bring back the excluded part of the text.

10. Skew

A skew is a rotation of angle in the scanned document that happens in the scanning process. The document may bend because there is no fixed position to the camera. A skew detection process is needed before calling the recognition engine. If any skew is detected, an auto-rotation procedure is performed to correct the skew before processing text further. A straightforward solution to determining the skew angle of a document image uses a horizontal projection profile. This is a one-dimensional array with a number of locations equal to the number of rows in an image. Each location in the projection profile stores a count of the number of black pixels in the corresponding row of the image [47].

Preprocessing goes with some stages to handle the noises and other factors to be ready for recognition. These steps are perspective rectification, binarization, skew detection and auto-rotation, and noise removal [25][3][5][24]. All the above steps will be discussed in detail in the next chapter.

2.1.3 Segmentation

Segmentation is a process of segmenting the document images into blocks, the blocks into text area blocks and non-text area blocks, and the text area blocks into characters. It is a stage followed by preprocessing where focus is on atomizing the data at various level like sentences, words and characters [24]. High degree of accuracy is to be maintained in these two stages for any error in these stages would result in cascade effect and thereby produce a multiplied growth in steps following it which may comprise feature recognition, extraction and classification. There are basically two commonly used segmentation algorithms stage by stage and recursive segmentation [13][11].

As mentioned in article [14][24][16] stage by stage segmentation is performed in three successive steps: line segmentation, word segmentation and character segmentation. Line segmentation relates to the process of separating text lines from a given scanned image of a document, word segmentation is a separating of words from a given line of text and character segmentation is identification of characters from a given word.

Several researches report the usage of Vertical projection profile (VPP) and Active Contour Model (ACM) for word and character segmentation and Horizontal Projection Profile (HPP) for line segmentation [24][16][13]. The HPP is calculated by summing the white pixels in each row of the image. The HPP graph contains peaks and valleys symbolizing the text lines and line gaps respectively. The VPP is calculated by summing the white pixels in each column of the image. The VPP graph contains peaks and valleys symbolizing the words and word gaps respectively in a line. An active contour is also called snakes is an energy minimizing spline (in mathematics, it is a numeric function that is piecewise defined by polynomial functions, and which possesses a high degree of smoothness at the places where the polynomial pieces connect) that detects specified features within an image. It is a flexible curve which can be dynamically adapted to required edges or objects in the image (it can be used to automatic objects segmentation).

2.1.4 Character Recognition

2.1.4.1 Feature Extraction

Feature extraction methods analyze the input document image and select a set of features that uniquely identifies and classifies the character. In feature extraction stage, each character is represented as a feature vector, which becomes its identity. During training stage, feature vectors of characters are prepared as templates which will later on serve in matching of features of acquired images of symbols with the respective templates [4]. This method defines each character by the presence or absence of key features, including height, width, density, loops, lines, stems, and other character traits.

Yaregal [18] use Gradient field for the feature extraction which has been used as a traditional tool over many years for feature extraction in image analysis problems. Gradient is a vector representing the change in gray level in two orthogonal directions. This can be calculated by taking the difference in value of neighboring pixels in a given pair of orthogonal directions, producing a vector for each pixel. The magnitude of the vector at each pixel measures the amount of change in

intensity, and the angle of the vector shows the direction of maximal intensity changes and can be expressed in the range of [0. . .360) degrees.

Some of the statistical feature extraction methods mentioned in literatures includes Crossings, Fourier Transforms, Projection Histograms, Moments, and Zoning. Hossain, Ashraufl and Yan [27] used Celled Projections which is the horizontal projections, a character image is partitioned into k regions and then the projection is taken for each region.

2.1.4.2 Classification

The feature extraction is critical and many different techniques exist, each having its strengths and weaknesses. After classification the identified characters are grouped to reconstruct the original symbol strings, and context may then be applied to detect and correct errors.

Zahid et al. [27] mention three classifiers, k-nearest neighbor rule (KNN), probabilistic neural network (PNN), and feed forward back propagation neural network (FBPN). The Neural network develops its information categorization capabilities through learning process from examples known as training. In this training process the network adjust its weights and biases to perform accurate classification. Certain characteristics are the extracted from the character for classification.

Million uses a Support Vector Machine *(*SVM) classifier which has the ability to identify the decision boundary with maximal margin, which results in better generalization. A highly desirable property for a classifier to perform well on a novel data set [10] [11]. SVM is also suitable for OCR problems with high dimensional input data due to its effective training and testing algorithms and natural extension to the kernel methods [11].

Template matching, structural analysis and neural network have been very popular classification techniques for character recognition. Abay [26] mentioned some classification approaches including statistical, structural, and mathematical (discriminant function classifiers, Bayesian classifiers, artificial neural networks (ANNs), and template matching). The statistical pattern recognition relies on defining a set of decision rules based on standard statistical theory, The Structural classification methods use structural features and decision rules to classify characters example the letter "ዩ" may be described as a vertical stroke with a loop attached to the upper right side, The Mathematical template matching classifier uses individual image pixels as features, its

Classification is performed by comparing an input character image with a set of templates from each character class and the mathematical Artificial Neural Networks uses architecture, which can be trained and correctly associate input patterns with desired responses. This alternative approach, using artificial neural networks, attempts to exploit knowledge of how biological neural systems store and manipulate information.

In [14] A Radial basis function (RBF) neural network is used which is trained for the recognition of characters, the full set of 157 characters including isolated Tamil characters, English numerals and punctuation marks are taken for training the neural network. The characters are placed at the center of a 52 * 52 window and the input patterns to the RBF neural network are obtained from the response of 40 Gabor filters with 10 along each of four directions. The RBF neural network has 157 outputs each output corresponding to an alphabet. The trained neural network is used for the recognition of the segmented characters [14].

2.1.5 Post processing

OCR is still imperfect as it occasionally misrecognizes letters and falsely identifies scanned text, leading to misspellings and linguistics errors in the OCR output text. The goal of post-processing is to detect and correct linguistic misspellings in the OCR output text after the input image has been scanned and completely processed [11][29].

Basically, there are two types of OCR errors: non-word errors and real-word errors. A non-word error is a word that is recognized by the OCR system; however, it does not correspond to any entry in the lexicon. For instance, when “How is your day” is recognized by the OCR system as “Huw is your day”, then “Huw” is said to be a non-word error because “Huw” is not defined in the English language. In contrast, a real-word error is a word that is recognized by the OCR system and does correspond to an entry in the lexicon, although it is grammatically incorrect with respect to the sentence in which it has occurred. For instance, when “How is your day” is recognized by the OCR system as “How is you day”, then “you” is considered a real - word error because “you” although is syntactically correct, its usage in the sentence is grammatically incorrect. Typically, non-word and real - word errors fall under three classes of errors: deletion, insertion, and substitution errors [29].

Generally errors correction mechanisms can be broadly broken down into three major categories: manual error correction, dictionary-based error correction, and context-based error correction. Youssef and Mohamed [29] proposed a post-processing method for OCR error correction based on spelling suggestion and the "*did you mean*" feature of Google's online web search engine. The goal of this approach is to automate the proofreading (read for errors) of OCR text and provide context-based detection and correction of OCR errors.

2.2 Challenges of OCR in mobile captured image

OCR techniques assume high resolution, high quality images with a simple structure (High contrasting test and background) for good quality recognition. Scanners mostly perform very well for the above mentioned tasks. Unfortunately, due to the typical nature of mobile cameras and environment, it is not always possible to assure the above. Many discrepancies may arise, such as explained as follows; Uneven lighting conditions, Skewness (Rotation), Tilting (perspective distortion), Blur, and warping [4].

Document images from printed documents, such as books, magazines, newspapers, etc. are extremely poor in 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, Floating ink from facing pages etc.

Printed documents vary in fonts, sizes, and styles. Building character recognition system is challenging in this situation; large number of characters in the script the total number of characters in Amharic script is more than three hundred. Existence of such a large number of Amharic characters in the writing system is a great challenge in the development of Amharic character recognizer. Memory and computational requirements are very intensive. We need to design a mechanism to compress the dimension of character representation so as to come up with computationally efficient recognizers.

2.3 Challenges of Amharic Character in Separation of text region

The Amharic writing system is a working language of Ethiopia. The Amharic script which has 33 core characters each of which occurs in seven orders (one basic and six non basic forms), which represent syllable combinations consisting of a consonant and following vowel. This makes a total of 231 core characters. Other symbols representing labialization, numerals, and punctuation marks are also available. These bring the total number of scripts to 310 [11][12][16].

The Amharic characters have challenge in separating the text from background and non-text region. The Amharic character shape challenges extraction of a text region from the document image like some characters shape is disconnected each other into two or three like ሸ which is into two and ቯ into three. So in separating the text region this characters upper line may be eliminated by the system.

There is also similarity in shape among different characters in Amharic writing system. Consider characters such as; (ሸ and ሀ), (ደ and ጀ) and (ሰ and ሻ), etc. [43]. If the line under the characters is eliminated or ignored by the system it will locks exactly like the similar one for example, in the character ሸ if the line under the character eliminate it will lock exactly like ሀ. Furthermore when we use the text for further processing like OCR for example, the system will recognize the character ሸ with no difficulty as ሀ which is wrong. Therefore, such feature similarities among different characters affect the performance of system.

The printed Amharic documents vary in fonts, sizes, and styles. The document that have a highly different font size will be a challenge in separating the text from the non-text one. Because if the character size is as large as the picture or logo of the document it will be difficult for the system to differentiate the text from the non-text one.

2.4 Related Works

It is only recently that researchers started studying the issue of image processing of mobile captured documents. Due to limitations of optical character recognition systems a number of attempts are made by researchers to avoid the noise and to extract the text from the document image. In this thesis, we attempt to review researches of preprocessing that are done on documents that are scanned by a desktop scanner of Amharic language and a document scanned by mobile camera of other languages.

2.4.1 Global Research Works

Anand et al. [9] proposed a techniques to addresses the challenges of mobile captured documents. For the lighting issue they propose a binarization method, because of the illumination over the document is not uniform in mobile captured document they experiment on a local binarization method such as Niblack's algorithm, and Sauvola's algorithm, in which they compute thresholds individually for each pixel using information from the local neighborhood of that pixel. And based on their experiments, they found Sauvolas's algorithm worked better and faster.

Anand et al. [9] although propose RAST algorithm (Branch-and-Bound text line finding algorithm) for text skew detection and correction. The basic idea of this algorithm is to identify each line independently and use the slope of the best scoring line as the skew angle for the entire text segment. After detecting the skew angle, rotation is performed accordingly. Based on their experiments, they found this algorithm to be highly robust and extremely efficient and fast. However, it suffered from one minor limitation in the sense that it failed to detect rotation greater than 30^0.

Megan and Margaret [15] has presented an image preprocessing system for use in machine recognition of text in natural scene images. Of the three steps of this preprocessing suite text detection, auto-rotation, and noise reduction and text detection becomes their most difficult problem. The experimental corpus of 62 color scene images containing text from public signage in the Princeton, New Jersey and Johns Creek, Georgia areas have been used. In text detection strategy, they determine the foreground color of a region by tracing its morphologically-grouped outer edges and determining whether the color just inside the edges is lighter (higher gray values)

or darker (lower gray values) than the color outside the edges. Based on the assessment of this color, they pick out the elements higher or lower than the local threshold as foreground pixels. Foreground pixels are then separated into connected components. The average gray value in the region is determined by Otsu's method whether the foreground is the lighter or darker side of the threshold. Their experimental results show similar values for each metric: on average, about 30% of the final region image is composed of non-text regions, and about 30% of text regions get excluded from the final region image. They proposed preprocessing suite is by no means complete. They recommend that one future challenge is the integration of other preprocessing steps such as perspective correction, light correction, and better enhancement of letter shapes for increased legibility.

Geetha and Murali [49] are made an attempt on rectifying a perspective distorted image from the captured document image. Their algorithm rectifies the perspective distortion without any reference objects or prior information about the camera parameters. The rectification may not be to the scale. There is a possibility to rectify to the scale using geometrical manipulation. The scale factor has to be derived before the manipulation using camera calibration technique. As Anand et al. [9] presents taking the advantage of an embedded orientation sensors to measure the tilt of the phone. Users are prevented from taking pictures if the camera is tilted to some extent, and as a result, the chances of perception distortion are reduced considerably. But if the perspective happened Geetha and Murali [49] proposes an algorithm that can handle a captured document image with a perspective distorted. They use a plane homography to transform the document image to its corrected position. It works by detecting the edges of the image first and find the four corner points of the image then by using a homographic matrix convert the 4 corners into the desired corner points. This method is found to be fully automatic and fast to rectify the perspective.

2.4.2 Local Research Works

Local researches are conducted to preprocessing of a scanned image in different sectors like document recognition and document retrieval. Abay [26], tried to apply OCR techniques in recognizing real-life Amharic printed documents for Amharic character recognition system. He took his training and testing data from real-life documents: Holy Bible, from the popular Ethiopian fiction 'Fiker Eskemekabir', from 'Addis Zemen' newspaper, and 'Federal Negarit Gazette'.

Before applying the segmentation algorithms on the scanned images, Abay [26] first tried to remove noises that are found in many real-life documents. He tested three noise removal algorithms (Linear filtering, Median filtering, and Adaptive Wiener filtering algorithms) and found out that Wiener filtering works well for most Amharic documents thus considered for his work. It was selected because it preserved edges and other high-frequency parts of a character. And for binarization global thresholding Otsu and adaptive thresholding Two-Dimensional Variance Adaptive Thresholding of Wavelet Coefficients was tested. Form the experiment Otsu binarization method was found to be more effective in isolating the text pixels from the background pixels. By using these and other recognition technique, Abay reported a recognition average accuracy of 96.87% in the training set and 11.40% in the test set. He recommended further exploring should be done in noise removal algorithms to enhance recognition rate as some of Amharic characters are similar in structure and Amharic document images are highly degraded.

Biniam [25] made an attempt in developing a retrieval of real-life Amharic document image by 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. He use 4,974 word images that are collected from real-life books, magazines, newspapers, and regulations with varying font styles, types, size.

To clean the noise in real life printed document images, Biniam [25] evaluates three noise removal techniques explicitly Median filter, Weiner filter and Adaptive Median filtering (AMF). And from his experiment 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. For binarization he implement and evaluate Otsu global thresholding and Niblack and Sauvola local thresholding methods. Based on the experiments the best performance and quality document image is the one that is using Otsu thresholding. The performance of the system achieved 96.67% Recall and 77.24% Precision. In an image preprocessing task skew detection and correction, image restoration and edge detection are vital and recommended to integrate with the system to increase the effectiveness and efficiency of the system.

Michael [22] developed recognition of real life documents by applying different preprocessing techniques. For the purpose of testing, test set proposed by Abay [26] was used. He tested noise removal algorithms of Median and Weiner filtering algorithm and he selected Weiner algorithm

because it performs better. Also for image thresholding, he tested Otsu and Sauvola algorithms and Sauvola performed better. Michael also applied the underline removal and normalization methods. By using these and other recognition technique, Michael [22] reported a recognition average accuracy of 88.38% in the test set. He also mentioned that better noise and underlines detection and removal, thresholding, and skew detection and correction algorithms should be explored in future studies. And suggest the need for future investigation on exploring effective preprocessing is needed.

Birhanu [13] explores some preprocessing techniques of real-life document images for page and text segmentation to improve the effectiveness of Amharic OCR. The dataset he collected was a total of 59 document images of which 26 of the document images are from Biniam [25] dataset and from his own 33 document images are used.

Birhanu [13] used Hough line transformation for skew detection and correction by searching for text base black lines of text bottoms followed by white line below on the gray scale document image input. And for text/graphics segmentation algorithm he apply Morphological Dilation and Connected Component (CC) analysis technique for the separation of text and non-text components. For his work, three noise filtering algorithms (Average/Mean, Median and Wiener) are tested to clean noise from the text area of document image. The result of his experiment indicates that wiener filtering method is able to smooth out noises with a few disturbances than average and median filtering methods. Although for image binarization Otsu thresholding and Sauvola thresholding algorithms are tested with different combinations of image denoising techniques and Sauvola was voted as best performing algorithm. His proposed system successfully segments 87.61% and 82.29% of characters from low and medium noise level images. He recommend for further research on text/graphics segmentation and enhancing of the image denoising and thresholding techniques.

Based on the recommendation of previous researchers globally [9] [15] [49] on mobile captured documents and locally [13] [22] [25] [26] on scanned document by a desktop scanner, the focus of this research is mainly integrating noise detection and removal, skew and perspective detection and correction, thresholding tools as well as text region extraction.

Chapter Three

Image Preprocessing Techniques

The preprocessing of an image is used for the recognition of characters from the image. Pre-processing is done to increase the quality of the image required to allow the steps following it to deliver accurate results [24]. As the first important step, image and data preprocessing serve the purpose of extracting regions of interest, enhancing and cleaning up the images, so that they can be directly and efficiently processed by the feature extraction component [25].

The input of a typical image processing system consists of color or gray level images in which every several hundred pixels corresponds to a linear inch in the paper medium. The output of the processed image, on the contrary, consists of ASCII strings that usually can be represented by just a few hundred bytes. To achieve such a high level of data abstraction, an image processing system depends on the following main components for information extraction and processing [33]:

- Image acquisition: acquire the image of a form in color, gray level, or binary format.
- Binarization: convert the acquired form images to binary format, in which the foreground contains logo, the form frame lines, the preprinted entities, and the filled in data.
- Data extraction: extract pertinent data from respective fields and preprocess the data to remove noise and enhance the data.

Reliably interpreting text from real-world photos is a challenging problem due to variations in environmental factors. Even the best open source OCR engine available (Tesseract) is easily letdown by uneven illumination, off-angle rotation and perspective, and misaligned text, among others [35]. The above open source engine also have challenges in preprocessing. Moreover, there is no enough Amharic OCR engine developed for this purpose we need to see more techniques and algorithms of to handle it.

3.1 System Architecture

The proposed architecture for mobile captured document image preprocessing is sown in Figure 3.1. The captured digitalized image is input to the system and the preprocessing techniques removes the noise, correct the perspective, correct the skew and extracts the text from the document image that have different pictures, logos and graphics and outputs the candidate text region. The output candidate text region is used for OCR to recognize the characters for further use.

The effectiveness of the previous preprocessing of Amharic document image by Biniam [25] Birhanu [13] and Michael [22] is highly affected by the noise that occurs by the degradation of the document, paper quality, and type. Although previous studies on mobile camera scanned document [5] [9] [15] [36] [49] mentioned that the effectiveness of this preprocessing is affected by the blur, perspective distortion and misalignment.

The image preprocessing tasks are skew and perspective rectification, noise detection and removal, thresholding and text region extraction. Because the portability of the devices mobile during the capturing process there are a lot of environmental factors affecting the digital image like uneven lighting condition and blur which needs an effective thresholding technique, geographical distortion which needs an effective skew and perspective correction method. And the noise of the image that is occurred due to the paper quality and degradation needs an effective noise detection and removal method.

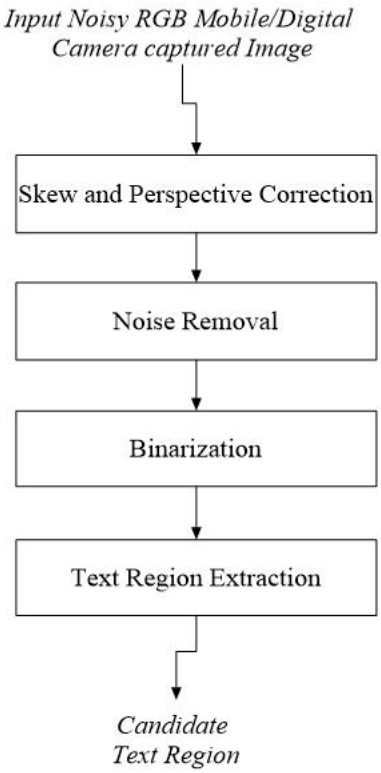


Figure 3.1: Architecture of the preprocessing Mobile captured Image

3.2 Pre-Processing Techniques

3.2.1 Smoothing and Noise Removal

As Atena [3] indicates, a document images may be contaminated with noise during transmission, scanning (A scanner, a mobile or a camera scanner/camera photo) or conversion to digital form. We can categorize noises by identifying their features and can search for similar patterns in a document image to choose appropriate methods for their removal. The noise introduced during scanning or due to poor quality of the page has to be cleaned before further processing. According to Megan and Margart [15], the noise is often a problem in camera-based text imaging; the input often has much lower resolution that that offered by scanners.

Smoothing operations in gray level document images are used for blurring and for noise reduction. Blurring is used in preprocessing steps such as removal of small details from an image. In binary (black and white) document images, smoothing operations are used to reduce the noise or to straighten the edges of the characters, for example, to fill the small gaps or to remove the small

bumps in the edges (contours) of the characters [13][33]. The smoothing implies both filling and thinning. Filling eliminates small breaks, gaps, and holes in the digitized characters, while thinning reduces the width of the line. The most common techniques for smoothing is moves a window across the binary image of the character, applying certain rules to the contents of the window [32]. Image de noising becomes a challenge for researchers because noise removal itself introduces some artifacts and blurring that can causes a loss of vital information from the document image.

Smoothing and noise removal as discussed in Biniyam [25] can be done by filtering. In the field of digital image processing, the primary technique to remove noise from any type of image is filtering. Filtering is a neighborhood operation, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel [33]. A pixel's neighborhood is a set of pixels, defined by their locations relative to that pixel.

Each of the filters can remove the small pieces of the noise (salt and pepper noise) in the gray level images; also they can blur the images in order to remove the unwanted details. Connected component algorithm is proposed by Deepa and Leema [24] for the elimination of these border noises. It scans an image and groups pixels based on its connectivity. Generally CC analysis is done over binary images. The output of CC analysis gives a label matrix having same dimension as of the original image.

There are two types of filtering approaches: linear filter and nonlinear filter [25][33][41], where linear methods are fast enough which are used to remove certain type of noise, but they do not preserve the details of the images, whereas the non- linear methods preserve the details of the images. A Mean and Wiener filter are liner filters and Median filter is a non-linear filter. The figure below shows example of filter mask for smoothing:

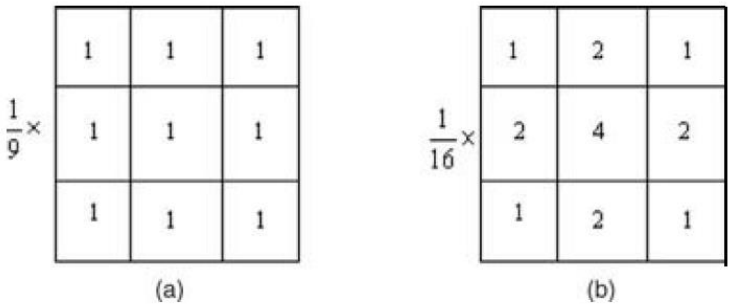


Figure 3.2: Two 3 by 3 smoothing (averaging) filter masks. (Adopted from [41])

3.2.1.1 Mean filtering

Mean filter is an averaging linear filter. Here the filter computes the average value of the corrupted image in a pre-decided area, which is a simple sliding window spatial filter. Then the center pixel intensity value is replaced by that average value. This process is repeated for all pixel values in the image. As mentioned in Sakthivel and Prabhu [40] mean filters perform spatial filtering on each individual pixel in an image using the grey level values in a square or rectangular window surrounding each pixel. See an example below that is adopted from [41].

Unfiltered Values		
8	4	7
2	1	9
5	3	6

8+4+7+2+1+9+5+3+6=45 45 / 9 = 5

An Example of mean filtering of a 3x3 kernel of values is shown below

Mean Filtered		
*	*	*
*	5	*
*	*	*

In this Center value which is previously 1 in the unfiltered value is replaced by the mean of all nine values that is 5.

Figure 3.3: Method of Mean Filter

3.2.1.2 Median filtering

Median filter is a non- linear filter, whose response is based on the ranking of pixel values contained in the filter region. Median filter is quite popular for reducing certain types of noise. It is used for reducing the amount of intensity variation between one pixel and the other pixel. Here the center value of the pixel is replaced by the median of the pixel values under the filter region. Median filter is good for salt and pepper noise. These filters are widely used as smoothers for image processing, as well as in signal processing. A major advantage of the median filter over linear filters according to literatures is that the median filter can eliminate the effect of input noise values with extremely large magnitudes (de-noises different types of noise).

According to Priyanka and Versha [41] the median filtering processes to calculate the median by first sorting all the pixel values into ascending order and then replace the pixel being calculated with the middle pixel value. If the neighboring pixel of image which is to be contain an even numbers of pixels, than the average of the two middle pixel values is used to replace. Example of a median filter of 3*3 kernel is shown in figure 3.4:

8	4	7
2	1	9
5	3	6

1, 2, 3, 4, **5**, 6, 7, 8, 9

Median (Central value 1 is replaced by 5)

Figure 3.4: Method of Median Filter

3.2.1.3 Wiener filtering

The wiener filter is the MSE (Mean Square Error) - optimal stationary linear filter for images degraded by additive noise and blurring. The Wiener filtering executes an optimal tradeoff between inverse filtering and noise smoothing. It removes the additive noise and inverts the blurring simultaneously. According to Priyanka and Versha [41] it minimizes the overall mean square error in the process of inverse filtering and noise smoothing. It performs the deconvolution by inverse filtering (highpass filtering) and removes the noise with a compression operation (lowpass filtering). The Wiener filter in Fourier domain can be expressed as follows:

$$W(f_1, f_2) = \frac{H^*(f_1, f_2) S_{xx}(f_1, f_2)}{|H(f_1, f_2)|^2 S_{xx}(f_1, f_2) + S_{\eta\eta}(f_1, f_2)} \quad (3.1)$$

Where $S_{xx}(f_1, f_2)$, $S_{\eta\eta}(f_1, f_2)$ are respectively power spectra of the original image and the additive noise, and $H(f_1, f_2)$ is the blurring filter.

The technique is also used for removal of blur type of noises in images. An example is the noises that are caused by linear motion or unfocused optics. The wiener filtering starts by calculating the mean and variance of neighboring pixels specified by window size of *m x n*. The approach often produces better results than the previous linear filtering algorithm. This is because it is more selective in preserving edges and other high-frequency parts of an image [13][22].

3.2.2. Image Binarization

Image binarization is a process of converting a grey scale image in to binary image that contain only pixel values of ones and zeros. Traditionally, OCR systems have a binarization step that classifies image pixels as text (black) or background (white). The method to binarization is to change the pixel values of area of interest (here greatly characters or text) to higher value that is to one and to make all other part zero by fixing a threshold value. The pixels values below this threshold are to be converted to zero and above this threshold to one [24][13][15]. The simplest implementation of thresholding is to choose an intensity value as a threshold level and the values below this threshold become 0 (black) and the values above this threshold become 1 (white).

As mentioned in Anand et al. [9] an embedded camera inside the mobile phone has far less control of lighting conditions than scanners. Uneven lighting is common, due to both the physical environment (shadows, reflection, fluorescents) and uneven response from the devices. Further complications occur when trying to use artificial light, i.e. flash, which results in light flooding. Binarization has long been recognized as a standard method to solve the lightning issue.

Thresholding is a technique that separates a pixel to object or background by comparing its gray level or other feature to a reference value. Binarization techniques based on gray levels can be divided into two classes: global and local thresholding [33]. Global thresholding algorithms use a single threshold for the entire image based on the properties of the entire image and Local (or adaptive), which considers the properties within smaller regions to produce possibly differing partitions for each region [33] [15].

Real-life documents especially mobile captured business card documents are designed deliberately with stylistic, colorful, and complex backgrounds, causing difficulties in character extraction methods. While global thresholding techniques can extract objects from simple, uniform backgrounds at high speed, local thresholding methods can eliminate varying backgrounds at a price of long processing time [4][25][20]. From the local adaptive thresholding algorithms *Sauvola's* Method, *Bernsen's* method, *Chow* and *Kaneko's* method, *Eikvil's* method, *Mardia* and *Hainsworth's* method, *Niblack's* method, *Taxt's* method, *Yanowitz* and *Bruckstein's* method, White and Rohrer's *Dynamic Threshold* Algorithm, White and Rhorer's *Integrated Function* Algorithm, *Parker's* method, and *Trier* and *Taxt's* method the Sauvola's, Niblack's and Bernsen's methods along with post processing step are proposed by literatures were the fastest and best.

3.2.2.1 Otsu Thresholding Method

Among many global thresholding techniques, the Otsu threshold selection is ranked as the best and the fastest global thresholding method as mentioned by many scholars [7][33]. The thresholding operation corresponds to partitioning the pixels of an image into two classes, C_0 and C_1, at a threshold t. The Otsu method solves the problem of finding the optimal threshold t^* by minimizing the error of classifying a background pixel as a foreground one or vice versa. Without losing generality, we define objects as dark characters against lighter backgrounds. For an image with gray level ranges within $G = \{0, 1, \ldots, L-1\}$, the object and background can be represented by two classes, as $C_0 = \{0, 1, \ldots, t\}$ and $C_1 = \{t+1, t+2, \ldots, L-1\}$ [33][26].

Otsu's thresholding method 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 [25]. 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 number of pixels in each class is close to each other. The Otsu method still remains the reference for comparing different thresholding and binarization methods in general. Camera flash has uneven lighting conditions due to this global thresholding method for binarization is not ideal.

3.2.2.2 Niblack's Local Adaptive Thresholding Method

Niblack's local adaptive thresholding is one of the most popular local threshold binarization method which is proposed by W. Niblack. 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 [25]. The algorithm of Niblak's method is calculated as:

$$T(x,y) = m(x,y) + w \times s(x,y) \tag{3.2}$$

Where, T is the threshold, m(*x*,*y*) is the mean, s(*x*,*y*) is the standard deviation of all pixels in the window and *w* is a constant, which determines how much of the total print object edge is retained, and has a value between 0 and 1.

3.2.2.3 Sauvola Local Thresholding Method

Sauvola's algorithm claims to improve Niblack's method by computing the threshold using the dynamic range of image gray-value standard deviation [25][38]. Niblack proposed that a threshold for each pixel be calculated based on the local mean and local standard deviation. Sauvola's variant of Niblack's method is implemented by dividing the grayscale image into N ×N adjacent and non-overlapping blocks and processing on each block separately. In Sauvola's binarization method, the threshold $t(x, y)$ is computed using the mean $m(x, y)$ and standard deviation $s(x, y)$ of the pixel intensities in a $w \times w$ window centered on the pixel (x, y) [25]:

$$t(x,y) = m(x,y)\left[1 + k\left(\frac{s(x,y)}{R} - 1\right)\right] \tag{3.3}$$

Where, R is the maximum value of the standard deviation (R = 128 for a grayscale document), and k is a parameter which takes positive values in the range [0.2, 0.5]. The local mean $m(x, y)$ and standard deviation $s(x, y)$ adapt the value of the threshold according to the contrast in the local neighborhood of the pixel.

3.2.2.4 Bernse's Local Thresholding Method

The Bernsen's local thresholding method computes the local minimum and maximum for a neighborhood around each pixel $f(x, y) \in [0, L-1]$, and uses the median of the two as the threshold for the pixel in consideration [33]:

$$g(x, y) = (F_{\max}(x, y) + F_{\min}(x, y))/2, \tag{3.4}$$

$$b(x, y) = \begin{cases} 1 & \text{if } f(x, y) < g(x, y), \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

$F\max(x, y)$ and $F\min(x, y)$ are the maximal and minimal values in a local neighborhood centered at pixel (x, y).To avoid "ghost" phenomena, the local variance $c(x, y)$ can be computed as:

$$c(x, y) = F_{\max}(x, y) - F_{\min}(x, y). \tag{3.6}$$

3.2.3. Skew Detection and Correction

During the document scanning process, the whole document or a portion of it can be fed through the loose-leaf page scanner. So, document skew often occurs during document scanning. Camera captured images very often suffer from skew. Skew detection is one of the primary tasks to be solved in document image analysis systems, and it is necessary for aligning a document image before further processing. Example of skewed business card image is showed in figure 3.5:

Figure 3.5: Skewed business card image

There are several commonly used methods for detecting skew in a page. Some methods rely on detecting connected components and finding the average angles connecting their centroids, others use projection profile analysis [33]. According to Birhanu [13], skew detection techniques can be roughly classified as analysis of projection profile, Hough transform, connected components, Clustering and Correlation between lines techniques.

A skew detection process is needed before calling the recognition engine. If any skew is detected, an auto-rotation procedure is performed to correct the skew before processing the text. According to Anand et al. [9] traditional methods based on morphological operations and projection methods are extremely slow and tends to fail for camera-captured images, they proposed in their research to choose a more robust approach based on Branch-and-Bound text line finding algorithm (RAST algorithm) for skew detection and auto-rotation. The basic idea of this algorithm is to identify each line independently and use the slope of the best scoring line as the skew angle for the entire text segment. After detecting the skew angle, rotation is performed accordingly.

In the study of Birhanu [13], he adopted a Hough line transformation, it is based on searching for text base black lines of text bottoms followed by white line below on the gray scale document image input which supposes that a white-background document is provided with black letters on a text document images. Those various techniques of skew detection and correction perform two stage processes; the first is determining rotation angle ɵ and the second is rotating image by the angle -ɵ [13].

After the skew angle of the page has been detected, the page must be rotated in order to correct this skew. A rotation algorithm has to be both fairly fast and fairly accurate. The detected skew angle of the document image can be corrected by this Equation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad (3.7)$$

3.2.3.1 Hough Transform

The Hough transform is a well-known technique in computer vision that has been used to detect lines and curves in digital images. The Hough transform maps the individual pixels from the image domain into the parameter domain. It computes the values for the parameters of all the curves of a particular type (e.g., straight lines) that can pass through each black pixel.

In the standard Hough transform, all figure points $\{(x1, y1), \ldots, (xn, yn)\}$ from the picture space are mapped to the parameter space [28]. For instance, a line in the picture space, $y = mx + b$ is defined in the parameter space as follows:

$$\rho = x\cos\theta + y\sin\theta \quad (3.8)$$

Where the parameter ρ represents the distance between the line and the origin, and θ is the angle of the vector from the origin to this closest point. In which X and Y are axis and ρ is distance and θ the angle.

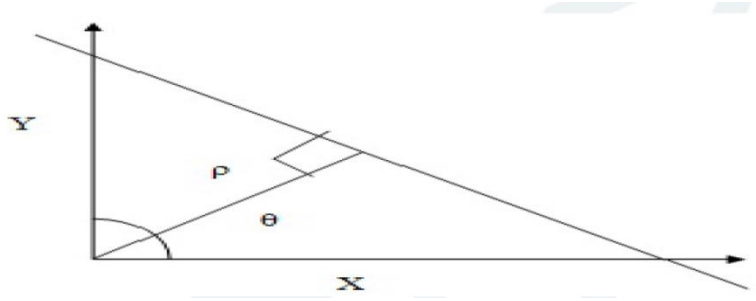


Figure 3.6: parameter plane of ρ and θ representation of a straight line

The skew angle is calculated on the basis that at the skew angle the density of Transform become is maximum. After mapping (x, y) into (ρ, θ), the count of points where a sinusoidal curve intersects another sinusoidal curve with a different (ρ, θ) value increases the probability that a line determining the skew angle. A lot of research has been carried out on using Hough transform for

determining skew angle. Some properties of Hough Transform are it is "Voting" algorithm, since each point is handled independently, parallel implementations are possible and Hough is computationally expensive algorithm and so it is advised to preprocess the document to reduce the number of pixels in order to reduce the processing.

3.2.3.2 Fourier transformation method

According to the property of the Fourier transform, low-frequency components determine global shape, and high-frequency components determine the fine details. To calculate Fourier transforms of the complex series composed by the *X* and *Y* coordinates of the boundary pixels, it is necessary to uniformly resample the image contour pixels at a number of power of 2 [28].

The Fourier transform of an intensity image function f (x, y) is given by [33]

$$F(wx, wy) = R(wx, wy) + jI(wx, wy) \qquad (3.9)$$

Where, $R(wx,wy)$ and $I(wx,wy)$ are the real and imaginary parts at each frequency (wx,wy), respectively

The Fourier Transform method works on the basic principle that skew angle is the one at which density of spectrum is largest for the document. However, the computational complexity is high.

3.2.4. Perspective Correction and Geometric Correction

The effect of perspective distortion is that characters farther away look smaller and are distorted and parallel-line assumptions no longer hold in the image. Hence the number of pixels to represent farther objects in the image requires lesser pixels as compared to the number of pixels required to represent the closer objects in the image. It happens a lot when we are using a hand-held camera to scan the document [1][9]. It occurs when the text plane is not parallel to the imaging plane. The ultimate effect on page segmentation and OCR depends on the specific algorithm. In Jian et al. [1] they found that small to mild distortion may cause significant trouble for some commercial OCR packages. For flatbed scanners, documents are perfectly aligned with the scanning surface, so this could not be a challenge at all. Correcting this distortion is a necessary because they have a direct effect on the reliability and efficiency of the segmentation and feature extraction stage. If the

system is perspective intolerance it may cause lower recognition rate and accuracy [4]. Perspective rectification is performed to restore fronto-parallel view of distorted image. Most current text OCR engines are not perspective tolerant.

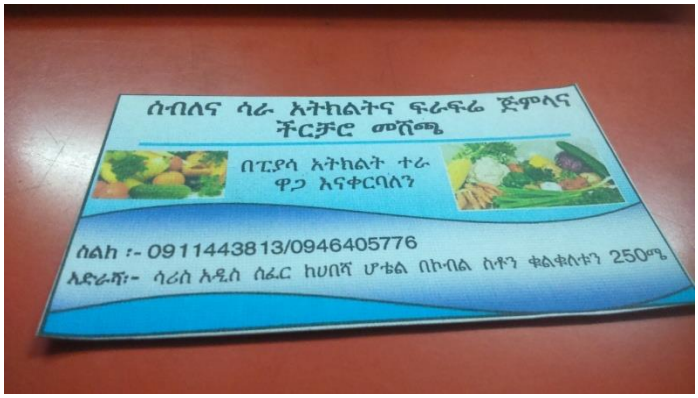


Figure 3.7: Perspective distorted captured image

Typical document image analysis algorithms to correct perspective distortion are document boundaries, page layout information, and organization of text and graphics components. A perspective image of a business card is rectified with the help of its bounding rectangle [44]. The border of a business card which is a rectangular document helps to correct the distorted perspective.

When two dimentional objects are scanned, the images observed from multiple views are related by a linear projective transformation, referred to as Homography [44].

$$\mathbf{x_i}' = \mathbf{H}\mathbf{x_i} \tag{3.10}$$

Now, if the positions of its vertex are calculated, the perspective can be corrected by using equation:

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}$$
$$y' = \frac{h_{12}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \tag{3.11}$$

Where: (x, y) - are coordinates of pixels in image

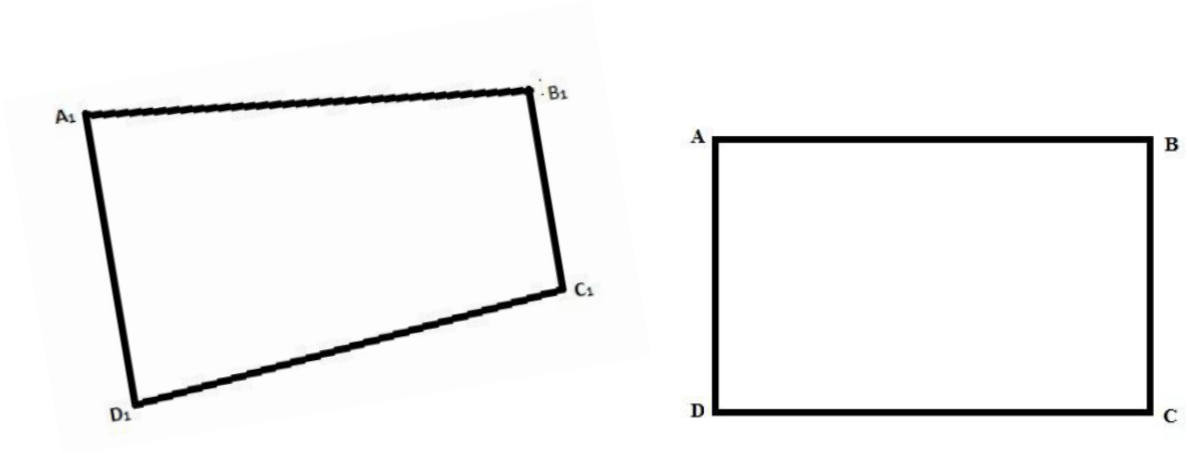
(x', y') - are coordinates of pixels of approximation of the original document

And H is the homography of a matrix size 3 * 3.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

Atishay et al. [4] in their paper, OCR in mobile devices take advantage of orientation sensors embedded in the mobile devices. It detects the tilt of the phone and disallows user to capture images when distortion occurs. This is a preventive technique rather than a corrective one.

The four corner points are required to implement the perspective transformation. The plane homography is interested with only four points i.e., the intersection point of the edges. It is obtained by taking the minimum and maximum values in the corner points array [49].



(a) (b)

Figure 3.8: (a) Perspective distortion (b) Rectangular plane

The corner points are further sorted in clockwise direction. The four corner points D1(X1, Y1), C1(X2, Y2), B1(X3, Y3) and A1(X4, Y4) of the rectangular object (view volume) of the perspective image is used in the plane homography method as in Figure (a). The rectified image is obtained having the corners D(x1,y1), C(x2,y2), B(x3,y3) and A(x4,y4) where (x1,y1)=(0,0), (x2,y2)=(L,1), (x3,y3)=(L,B) and (x4,y4)=(1,B) as in the above Figure (b).

3.2.5 Text Area Segmentation Techniques

Segmentation is a process of separation of an image into regions that contain pixel groups that are similar in value. Text and graphics are the major constituents of any document image. Segmentation occurs at two levels; on the first level, blocks of text, graphics, and other parts are

separated, on the second level, text lines and words in the text image are located. Segmentation of graphics is essential for better OCR performance to get the text region and non-text region (logo, pictures) from the business card and it is particularly difficult in the context of graphics made of small components having features similar to texts.

One of the most important steps of designing an efficient Business Card Recognition system is to segment the business cards properly. Improper segmentation affects the recognition process and thus the overall performance of the system. So, the text regions and the non-text elements such as logo, images, graphics, etc. must be identified at first. Then from the text region the lines, words, and characters are to be segmented. There are several methods to extract text region from a document that are proposed by different authors such as color clustering, intensity profiles, spatial variance, support vector machine (SVM) classifier, texture analysis based, connected component, edge based, region based, morphological operations and projection profiles [13][36][43][51].

Text segmentation is the stage that focuses on the textual section of document image extracted from segmented text region that are preprocessed. This stage identifies text lines, words, and characters from the preprocessed image. The accuracy of the recognized characters or texts are depending on mainly in the segmentation of these characters, word and lines. Horizontal projection profile for line segmentation and Active Contour Model (ACM) for word and character segmentation has demonstrated significant potential in handling OCR for Devanagari script [24]. Projection profile is also used for Amharic script in literatures by Worku [19], Dereje [12] and Million [11] for line, word and character segmentation which is a stage by stage segmentation.

At first, the camera captured business card images are virtually split into small blocks. A block is part of either background or a foreground component. This classification is done on the basis of intensity variance within the block. The intensity variance is defined as the difference between the maximum and the minimum gray scale intensity within the block [36]. It is observed that the intensity variance of a text block is considerably more than that of a background block. If the intensity variance of a block is less than an adaptive threshold, it is considered as part of the background. Otherwise, it is considered as part of a foreground component. The foreground components including texts are isolated from the background. Then the non-text components among them such as logos, lines, graphics, pictures, and noises are eliminated.

To subdivide the image *img* into their groups (text, graphics, space) the image has to be subdivided into blocks; the block size is predefined and changes correspond to the size of the image document. Each block becomes the smallest unit for further processing. For an image *Img* is defined as [51]:

$$Img = \{p_{ij},\ 0 \leq i < H, 0 \leq j < W\} \quad (3.12)$$

Where p_{ij} is a pixel at location *i, j*; H and W are respectively the height and the width of the image. The subdivision of *Img* into blocks can be expressed

$$Img = \{b_{IJ},\ 0 \leq I < \frac{H}{h}, 0 \leq J < \frac{W}{w}\} \quad (3.13)$$

Where b_{IJ} is a block at I_{th} row and J_{th} column; h and w are respectively the height and the width of the blocks. A block b_{IJ} is defined as [51]:

$$b_{IJ} = \{p_{ij},\ h \times I \leq i < h \times (I+1),\ w \times J \leq j < w \times (J+1)\} \quad (3.14)$$

3.2.5.1 Morphological Dilation

Morphology is biological term that refers to study of form and structure. Morphological operators often take a binary image and a structuring element as input and combine them using a set operator (intersection, union, inclusion, complement) [37][13]. Morphological processing can be employed for many purposes including preprocessing, edge detection, segmentation, and object recognition. Erosion and Dilation are considered the primary morphological operations; Dilation adds pixels while erosion removes the pixels at boundaries of the objects. This removal or adding of pixels depends on the structuring element used for processing the image.

Erosion causes the objects to shrink or become thin in size. Erosion basically erodes away the boundaries of the foreground which results in areas of those pixels shrink in size and holes of those areas become larger. It changes a foreground pixel to background.

Dilation causes the objects to grow in size. The effect of this operation will gradually increase the boundaries of foreground pixels, thus areas grow in size, and holes in that region become smaller. Suppose A be a set of input image coordinates and B be a set of structuring element coordinates and Bx is a translation of B so that its origin is at x. Thus dilation of A by B is set of all points of

x such that intersection of Bx with A is not null. In terms of set operations dilation of A by B is defined as [36]:

$$A \oplus B = \{x \mid (\hat{B})_x \cap A \neq \phi\} \quad (3.15)$$

Example of dilation that change a background pixel to a foreground if it has a foreground pixel as a 4-neighbor.

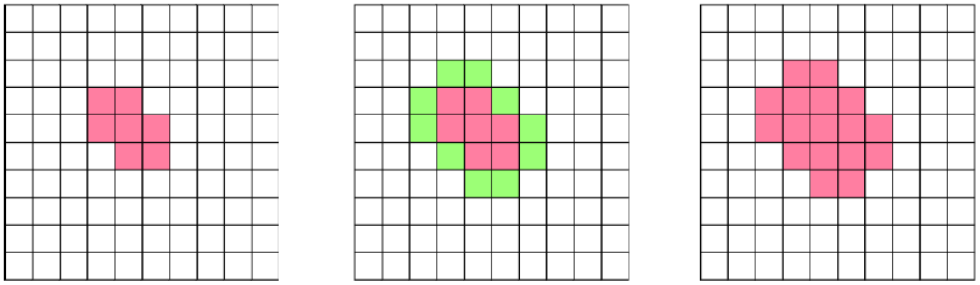
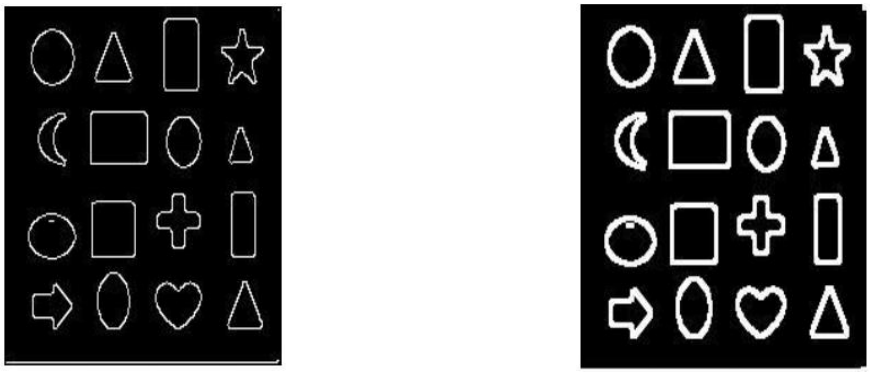


Figure 3.9: Example that show dilation

Below image shows the output of diluted image on edge detected image output:



(a) Binary Image (b) **Diluted image**

Figure 3.10: Dilation image

3.2.5.2 Connected Component (CC) Labeling

Connected Component labeling is used in computer vision to detect connected regions in binary digital images. Connected components analysis scans all the pixels of document image and groups them into components based on pixel connectivity, i.e. all pixels in the connected component shares similar pixel intensity values and are in some way connected with each other. Once all groups have been determined, each pixel is labeled with a gray-level or color labeling according to the component it was assigned to. Extracting and labeling of various disjoint and connected

components in an image is central to many automated image analysis applications such as OCR systems [13][43].

Connected Component also called as blob detection or region extraction, CC is commonly used analysis in image segmentation algorithms. It scans an image and groups pixels based on its connectivity. Generally CC analysis is done over binary images. The output of CC analysis gives a label matrix having same dimension as of the original image [24]. There are two types of connected component labeling; one pass and two pass. The one pass version goes through each pixel only once and for each pixel in an image, all the neighbor pixels are tested for connectivity to label connected components and the two pass scans the image two times.

3.3 Performance Evaluation

To measure the performance of the preprocessing technique proposed in this study PNSR, MSE, Precision and Recall is used. The final result (preprocessed) found by the experiment is evaluated by manually counting the expected correct text area which is the characters by the researcher and calculating the Precision, and Recall.

3.3.1 Peak signal-to-noise ratio (PSNR)

PSNR is a term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. To compute the PSNR, the block first calculates the mean squared error using the following equation [31]:

$$PSNR = 20\log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right) \quad (3.16)$$

Where:

MAX_f = is the maximum signal value that exists in our original “known to be good” image

3.3.2 Mean Square Error (MSE)

MSE allows us to compare the “true” pixel values of our original image to our degraded image. The MSE represents the average of the squares of the "errors" between our actual image and our noisy image. The error is the amount by which the values of the original image differ from the degraded image [31]:

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}\|f(i,j) - g(i,j)\|^2 \quad (3.17)$$

Where:

f = represents the matrix data of our original image

g = represents the matrix data of our degraded image in question

m = represents the numbers of rows of pixels of the images and i represents the index of that row

n = represents the number of columns of pixels of the image and j represents the index of that column

3.3.3 Precision

The precision rate is defined as the ratio of correctly detected text region (*True positive*) which is a character to the sum of correctly detected character plus false positive. *False positives* are those regions in the image which are actually not characters of a text, but have been detected by the algorithm as text regions [23]:

$$Precision\ Rate = \frac{True\ Positive}{True\ Positive + False\ Positive} \times 100\% \quad (3.18)$$

3.3.4 Recall

The Recall rate is defined as the ratio of correctly detected text region (*True Positive*) which is a character to the sum of correctly detected character plus false negatives. *False Negatives* are those regions in the image which are actually text characters, but have not been detected by the algorithm as text region [23]:

$$Recall\ Rate = \frac{True\ Positive}{True\ Positive + False\ Negative} \times 100\% \quad (3.19)$$

.

Chapter Four

Experimentation and Evaluation

This chapter presents the results of the experiment carried out to find a good preprocessing technique for the mobile captured business card document. For the experimentation TOSIBA Laptop computer with specification Intel® Core™ i5 CPU 550 @ 2.20 GHz 2.20GHz, 8GB RAM. Microsoft Windows® 8 operating system and MATLAB R2013a Version 8.1.0.604 image processing tool was used for image processing and a SONY Xperia™ M2 mobile phone for scanning or capturing a business card which has a 4.8 inch touch screen, 8 megapixel camera with a resolution of 229DPI (Dots Per Inch) used for the research. Most researchers point out that a resolution of and image less that 300DPI makes the recognition rate decrease. Our scanning machine the mobile phone has less resolution that requires a deep preprocessing before we sent the image into the recognition.

The systems assume that the user points the camera directly at (or even manually marks) the object that is captured, with blurring, scaling and perspective bias being the major issues that is addressed. Our objective is to utilize the visual capabilities of the smart mobile phone with internet connection to extract information from a business card with the aid of internet connection.

4.1 Dataset Preparation

The dataset collected contain real life business cards document images with different level of noise. The Amharic printed business card that have an embeedded graphics and images are used. The datasets are collected and captured by the researcher himself. The dataset doesn't contain handwritten and typewritten document images; rather it only contains printed documents.

The system was tested on 20 distinct RGB Amharic printed business card images with different noise level and all the test images were taken by SONY Xperia M2 mobile phone. The dataset contains both simple and complex cards including complex backgrounds and logos. Some cards contain multiple logos and some logos are combination of text and graphics. Most of the images are skewed, perceptively distorted, and degraded. All the test data images were taken by the researcher and have saved in JPG format, the default image format of the mobile phone. Table 4.1 shows the different document contents and noise level.

Document Content	Document Number
Logo	8
Column	3
Picture	4
Only Text	2
Both (Logo and Column)	3
Total	20

(a) Contents of the business card

Document Noise Level	Number of Document
Low Level	8
Medium Level	7
High Level	5
Total	20

(b) Level of noise of the document

Table 4.1 Summary of the dataset used in the study

4.2 Preprocessing

Preprocessing is vital use for the recognition processes. It improves the recognition rate of the OCR system. Moreover mobile captured documents have more challenges than the desktop scanners because of their poor scanning capacity, lighting condition and their perspective is not fixed like scanner. This makes the digital image to have more noise, so this has to be handled for better recognition.

We implemented and used built-in methods in MATLAB Image Processing Toolbox to reduce noise and convert images types. The original images were scanned as color (RGB - Red Green Blue) so we converted the image file from RGB to gray-level using *rgb2gray (image)* built-in method in MATLAB to make processing efficient and easy.

Once the document image becomes ready for processing, the first image preprocessing method implemented in this study is skew and perspective correction, forward by noise removal and binarization.

4.2.1 Skew and Perspective correction

In camera captured image we face not only the skewness of the image, there are also a perspective distortions due to the scanning process. A skew detection and perspective detection process is needed before calling the recognition engine. If any skew or perspective distortion detected, an auto-rotation procedure is performed to correct the skew and the perspective before processing text further.

A perspective and skewed image of a business card can be easily rectified with the help of its bounding rectangle as the researcher mentioned in the previous chapter. When the rectangular boundary is clearly distinguishable by locating the four corners of the card, it is possible to correct the image.

(a)

(b)

Figure 4.1: (a) Perspective distorted (b) Skewed document

We follow the following steps to correct the skewed and perspective distortion in this research:

First, identify the corners of the bounding quadrilateral (vertex of the card) in the given image.

The first step toward image perspective correction or skew correction is pointing out the quadrilateral containing the text scope. This operation can be carried out manually or with aid of the technique. But in this research the researcher use manual way to put the quadrilateral points.

The user after capturing the image it will identify the vertexes of the card by pointing to them in the touch screen of the mobile, the coordinates will be sent to the server with the image. For the experiment purpose the researcher use a MATAB built in function *ginput*() to get the coordinates. *[*X Y*] = ginput(4)* function assign the four coordinates to the array variables X and Y.

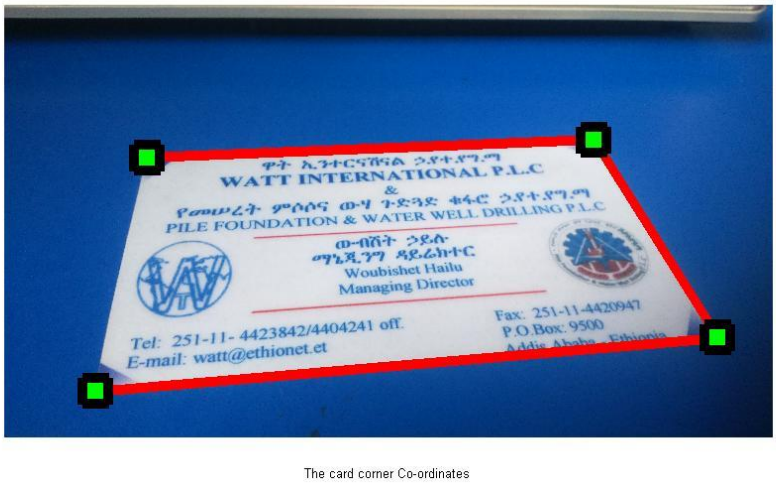


Figure 4.2: The card corner Co-ordinates (Finding the bounding rectangle)

Second, map each vertex of the quadrilateral to the corresponding vertex in the known rectangle.

The mapping of the retrieved coordinates from the MATLAB function *ginput()* is mandatory, because we just get the four coordinates in an array format that does not tell which one is for which coordinate X (x1, x2, x3, x4) and Y (y1, y2, y3, y4). After sorting the Y values, the 1st 2 high values for the y-axis are the top 2 edges so the upper corners of y axis are identified. Then we sort the coordinates according to X values. The sorting is done along different dimensions of the array, and arranges those elements in ascending order. We use *sort()* MATALB built in function to sort them. Get the vertex of the quadrilateral coordinates location upper/lower and left/right, the upper left and right corners and also the lower left and right corners will be identified.

v1 = (x_i, y_i)= left_top

v2 = (x_i, y_i) = left_bottom

v3 = (x_i, y_i) = right_top

v4 = (x_i, y_i) = right_bottom

Third, Transform using equations (eq. 3.11) and an additional constraint (eg. || h || is of unit norm), and the corresponding coefficients h_{ij} by computing the Homography H. The concept of Homography is to explain and study the difference in appearance of two plane objects viewed from different points of view.

The homography H is a matrix of size 3*3. This is defined only up to a scaling and hence has only 8 unknowns. Given four corresponding points (8 equations) in a general position, H can be uniquely computed. The four corresponding points in this research are (1,1), (2400,1), (2400,1500) and (1, 1500) which are the target corrected corner (vertex) point of the document image, the researcher set this values based on the business card size and the size of the image.

Basically, you have 2 collections of 4 points, A and B rectangles, in which we want to map a point in rectangle A to rectangle B (target point).

A = { p1x,p1y ; p2x,p2y ; p3x,p3y ; p4x,p4y }

B = { p1x,p1y ; p2x,p2y ; p3x,p3y ; p4x,p4y } = {1,1; 2400,1; 2400,1500; 1,1500}

Then we build a 8 x **n** matrix (M1), where **n** is number of points * 2(x,y). (Result will be a 8x8 matrix), the MATLAB function *zeros*(8,8) will create and set zeros value to the matrix.

For each point:

{xA, yA, 1, 0, 0, 0, -xA*xB, -yA*xB}

{0, 0, 0, xA, yA, 1, -xA*yB, -yA*yB}

We also make a 1 x 8 matrix (M2) with the "target"(B) points. Result must be 1x8

For each point:

{xB}

{yB}

Once calculated, these 8 parameters can easily be used to transform any point from the first reference system to the second. Then we can calculate the homography, result will be a 3 x 3 matrix (H). H = M1 / M2, and we reshape the matrix size to 3*3

Fourth, using H rectify the image to the frontal view.

To rectify the image from the original position to the frontal view we use this MATLAB built in functions *maketform()* and *imtransform().* The *maketform()* function creates a multidimensional spatial transformation structure that can be used with the *imtransform* function. The *imtransform()* transforms the image according to the 2-D spatial transformation defined by tform (Transformation Structure), in RGB image it applies the same 2-D transformation to all 2-D planes along the higher dimensions.

Projective transformation code:

```
T=maketform('projective',H');
TImage=imtransform(image,T,'XData',[1 2400],'YData',[1 1500]);
```

By using this method all document that are skewed or perspective distorted scanned (captured) image is corrected, if we correctly input the corner coordinates of the card. If we select the boundary corners of the card falsely we will get a wrongly rectified image.

Figure 4.3: Skew and Perspective corrected document image

An experiment shows that the text image surrounding or the image background surrounding of the card makes it hard to use another method for the skew and perspective correction. The experiment and Literatures also show traditional methods based on morphological operations and projection methods are extremely slow and tend to fail for camera-captured images.

4.2.2 Noise Filtering (Removal)

Noise removal is filtering out background textural matters, interfering strokes, shades and dots introduced due to input devices. To remove these noises different filtering algorithms are applied on images. Techniques used in this research are average filtering, median filtering, and wiener filtering. These filters are proposed by different literatures to remove the noise that arise due to the camera of the mobile and the paper quality [24][25][40][41][42].

The MATLAB Image Processing Toolbox provides a number of different ways to remove or reduce noise in an image some examples are *medfilt2() and wiener2().* For the experimentation purpose the MATLAB code *imnoise()* is used that adds noise to the image *imnoise(I,'salt & pepper',0.02)* this add salt and pepper noise to the image *imnoise(I,'gaussian',0.01)* this add gaussian noise.

4.2.2.1 Mean (Average) Filtering

Mean filter is a liner filter which is useful for removing Gaussian noise from a scanned document. Because each pixel gets set to the average of the pixels in its neighborhood, local variations caused by Gaussian are reduced. Gaussian noise is characterized by each image pixel having value from a zero mean Gaussian distribution. This filter tends to blur the sharp edges, destroy the lines and other fine details of image, and perform badly in the presence of signal dependent noise. There is no built in MATLAB code for this filter, the implementation is shown in Annex I.

MSE and PSNR are used to measure their performance of noise removal algorithms. Mean Square Error (MSE) is the cumulative square error between the encoded and the original image. Thus MSE should be as low as possible for effective compression. Peak signal to Noise ratio (PSNR) is the ratio between maximum possible power of a signal and the power of distorting noise which affects the quality of its representation [41]. Good performance is when we get lower values for MSE and higher values for PSNR.

The performance result of Average filtering having different window size with Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) is shown in below Table 4.2. Experimental result of the filter is shown in Figure 4.4 (c).

Noise Level	3 *3		7*7		20* 20	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
Low	**3.75**	**42.38**	6.96	39.89	94.31	28.94
Medium	**3.45**	**42.75**	7.49	38.67	102.91	28.01
High	**8.63**	**38.76**	9.35	37.61	159.47	26.10

Table 4.2: Experimental result (MSE and PNSR) of Mean Filtering using different window size

From the Table 4.2 we see that the MSE with 3*3 window size have less value than others and PNSR of 3*3 window size have greater value than others this imply we get good performance at this window size. We see that the increase in window size shows the increase in level of noise registered in the document that decreases the PSNR value and the visibility of each word decreases. The document image becomes blurred and the texts are unreadable to separate it from the non-text part.

4.2.2.2 Median Filtering

The median of a set is the middle value when they are sorted. If there are even numbers of values, the median is the mean of the middle two. MATLAB has a built in function *medfilt2()* that has two parameters the first one is *medfilt2(Image)* which accepts an image and filter it with a default 3-by-3 neighborhood matrix and the second one is *medfilt2(Image, [m n])* which accepts an image and the neighborhood matrix size m and n (Window size). The MATLAB functions medfilt2 expect a two-dimensional array (or grayscale image). Median often does a better job than the mean filter of preserving useful detail in the image.

Implementation of Median Filter:

```
function [outputImage] = median_filter(grayscale_image,m,n)
      outputImage = medfilt2(grayscale_image, [m n]);
end
```

The performance result of Median filtering having different window size with Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) is shown in below Table 4.3. Experimental result of the filter is shown in Figure 4.4 (d).

Noise Level	3 *3		7*7		20* 20	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
Low	**0.11**	**58.03**	3.46	46.07	24.75	31.45
Medium	**1.94**	**45.24**	4.32	41.76	64.56	30.03
High	**3.91**	**42.21**	5.30	40.05	128.13	27.05

Table 4.3: Experimental result (MSE and PNSR) of Median Filtering using different window size

From the Table 4.3 we see that the MSE with 3*3 window size have less value than others and PNSR of 3*3 window size have greater value than others this imply we get good performance at this window size.

4.2.2.3 Wiener filtering

MATLAB built in function wiener2 uses a pixel-wise adaptive wiener method based on statistics estimated from a local neighborhood of each pixel. *wiener2(I, [m n], noise)* filters the image I using pixel wise adaptive Wiener filtering, using neighborhoods of size m- by-n to estimate the local image mean and standard deviation. *wiener2(I)* has a neighborhood window size of default 3-by-3.

Experiment shows 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.

Implementation of Wiener Filter:

```
function [outputImage] = wiener_filter(grayscale_image,m,n)
     outputImage = wiener2(grayscale_image, [m n]);
end
```

The performance result of Median filtering having different window size with Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) is shown in below Table 4.4. Experimental result of the filter is shown in Figure 4.4 (c).

Noise Level	3 *3		7*7		20* 20	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
Low	**0.10**	**59.10**	1.69	50.52	15.09	34.07
Medium	**1.83**	**45.52**	4.12	42.01	17.44	35.71
High	**3.815**	**42.316**	4.93	41.17	26.30	33.93

Table 4.4: Experimental result (MSE and PNSR) of Wiener Filtering using different window size

From the Table 4.4 we see that the MSE with 3*3 window size have less value than others and PNSR of 3*3 window size have greater value than others this imply we get good performance at this window size.

4.2.2.3 Experimentation on Filters

The experimentation we have seen above for each filters is to find out the best window size for each of them. In this section we compare and contrast the three different filters with Mean Square Error (MSE) and Peak signal to Noise ratio (PSNR). The Figure 4.4 presents results of experimentation on the different filtering algorithms to the skewed and perspective corrected RGB image.

(a)

(b)

(c)

(d)

(e)

Figure 4.4: Experimental result of Noise Removal with window size of 3*3 (a) Original image (b) Perspective corrected image (c) Mean Filtering (d) Median Filtering (e) Wiener Filtering

The window size of a filters of median and wiener as an argument is expected, an experiment is performed to determine the one that gives better result. In this regard, a MATLAB default window size of 3x3 is found to perform best for both algorithms in this experiment. The window size is proved to smooth out noises and cause less distortion on the images. An iterative experiment on test set showed that a window size greater than this created blurred images.

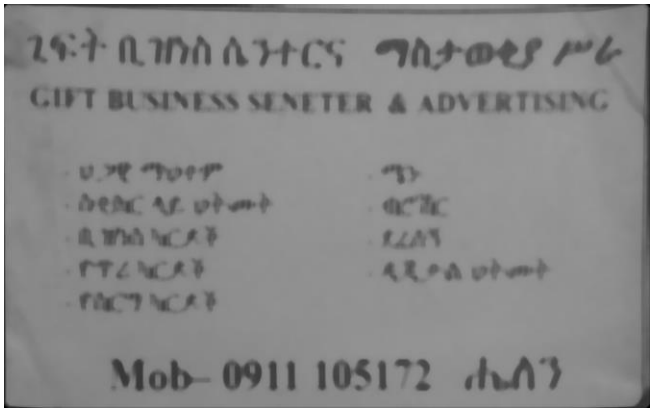


Figure 4.5: An image that show a greater window (20*20) size that create a blur image for Wiener filter

Based on the experiment under taken the wiener filter is an able to smooth out noise creating relatively small disturbances than its counter median filter. The result of MSE and PSNR in the three filters with window size of 3*3 is presented below in Table 4.5:

Noise Level	Mean Filtering		Median Filtering		Wiener Filtering	
	3*3		3*3		3*3	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
Low	3.75	42.38	0.11	58.03	**0.10**	**59.10**
Medium	3.45	42.75	1.94	45.24	**1.83**	**45.52**
High	8.63	38.76	3.91	42.21	**3.815**	**42.36**

Table 4.5: Experimental result (MSE and PNSR) of the three filters

From the Table 4.5 we see that a good performance in MSE and PNSR with a Wiener filter that is; experimental result shows less MSE in all noise levels and high PSNR in all noise levels with Wiener filter.

4.2.3 Binarization

Binarization is the next stage after image noise removal, which is converting the gray level images to binary images. Binarization has long been recognized as a standard method to solve the lightning issue. The goal of binarization process is to classify image pixels from the given input grayscale or color document into either foreground (text) or background and as a result, reduces the candidate text region to be processed by later processing steps. When a grayscale or colored image is binarized, a certain threshold value must be set so that pixel intensity levels that are less than this value are assigned to black pixels and values greater are assigned to white pixels.

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* (Threshold). Thresholding is a non-linear operation that is a simplest method of image segmentation from a grayscale image to create a binary image. There are lot binarization methods but in this paper the experimentation is done on three thresholding that are Otsu global thresholding, Sauvola local adaptive thresholding and Niblack local adaptive thresholding.

4.2.3.1 Otsu Global Thresholding

The Otsu thresholding algorithm is easily implemented because there is a built in function in MATLAB Image Processing Toolbox. The MATLAB function *im2bw(i, level)* is used to find the binarized image from the grayscale image. Where *i* is the input image and *level* the threshold value of the grayscale image. The threshold value is calculated by the built in function *graythresh(i)*. The function *graythresh*() can be used to compute the level which is a value between 0 and 1.The output image 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). By using this function we can easily find the global threshold of the image. The result of the algorithm test image is shown in Figure 4.6.

Implementation of Otsu Binarization:

```
function [outputImage] = otsusThreshold(inputImage)
    image = rgb2gray(inputImage);
    level = graythresh(inputImage);
    outputImage = im2bw( inputImage, level);
end
```

4.2.3.2 Niblack Local Adaptive Thresholding

The reason we choice this algorithm and Sauvola algorithm in our experiment is that they performs well as we have seen from previous literature experimental results. The local adaptive binarization technique is required in a complicated document images which has different background in the same document, in this case a local threshold needed which is calculated for each pixel.

The Niblack filter is a local thresholding algorithm that separates white and black pixels given the local mean and standard deviation for the current window. Setting the filter radius is mandatory and it is critical for the algorithm because the quality of the image is determined by this.

Filter Radius	MSE	PSNR
25	0.25	54.15
60	0.21	54.73
70	**0.19**	**55.21**
80	0.20	55.09

Table 4.6: Experimental result to select better filter radius

We use a filter radius of 70 and weight parameter(w) which is a constant (between 0 and 1) is 0.01 in our experiment as shown in the above Table 4.6 the values are found based on different experimental results. The filter size of 70 cleans the noise better than any other values. The step of this algorithm is first we build a filter of size 70 and then we calculate the mean and the standard deviation based on the filter we create. Then we calculate the threshold based on the formula which is multiplying all the three variables the mean, the standard deviation and the constant 0.01. Finally the pixel values that are greater than the threshold will be black and the others white this is the binary image. We found out that Niblack's binarization method produces more background noise than other see Figure 4.6.

The implementation code for Niblack thresholding algorithm in MATLAB is presented in Annex III. The result of the algorithm test image is shown in Figure 4.6.

4.2.3.3 Sauvola Local Adaptive Thresholding

Sauvola recently presented promising results using a variation of Niblack's binarization [7]. The local mean $m(x,y)$ and standard deviation $s(x,y)$ adapt the value of the threshold according to the contrast in the local neighborhood of the pixel. When there is high contrast in some region of the image, the standard deviation s(x,y) ~ R which, results in the threshold $T(x,y) \sim m(x,y)$ with the mean.

This is the same result as in Niblack's method. However, the difference comes in when the contrast in the local neighborhood is quite low. In that case the threshold $T(x,y)$ goes below the mean value thereby successfully removing the relatively dark regions of the background. The parameter k controls the value of the threshold in the local window such that the higher the value of k, the lower the thresholds from the local mean [35].

Window Size	MSE	PSNR
5	0.384	52.286
10	0.378	52.356
25	0.361	52.546
30	**0.357**	**52.594**
40	0.363	52.525

Table 4.7: Experimental result to select the better window size

The value of the parameter k is a bias, which takes positive values in the range [0.2, 0.5]. A value of $k = 0.5$ was used by J. Sauvola. Experiments with different values of k show that $k = 0.34$ gives the best results. The value of R which is the maximum deviation is calculated based on the mean square and the mean of the image. A block size of 30 × 30 window size was used for the research experiment because as it is shown in the Table 4.7 this window size performs good quality. Finally we use the Sauvola formula to compute the threshold (see formula 3.3). If the pixel value is greater than the optimal threshold it is black otherwise it is white, this is a binary image.

The implementation code for Sauvola thresholding algorithm in MATLAB is presented in Annex II. The result of the algorithm test image is shown in Figure 4.6.

4.2.3.4 Experimentation on Binarization

The researcher has point out in the previous section that Wiener filter performs better in our experiment and also from previous literatures. We use this filter and combine with the three thresholding techniques to see which binarization method is better. The Figures 4.6 shows the result the three binarization methods of a business card that is perspective & skew corrected and noise removed in Wiener filter.

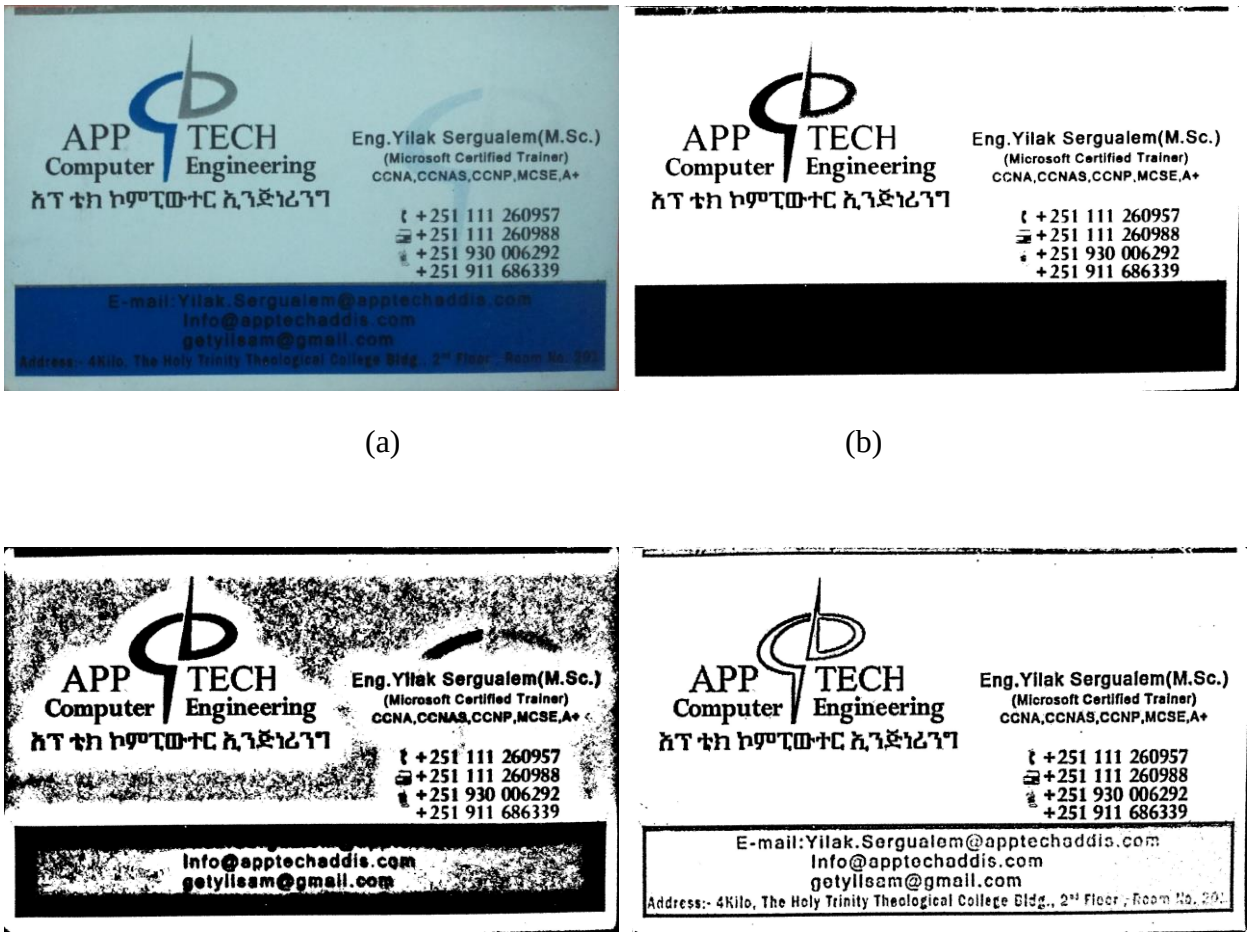


Figure 4.6: Experimental result of Image Binarization (a) Original image after skew and perspective corrected (b) Otsu threshold (c) Niblack threshold (d) Sauvola threshold

Noise Level	With Otsu		With Niblack		With Sauvola	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
Low	0.20	55.09	0.16	56.07	**0.07**	**59.75**
Medium	0.36	52.52	0.19	55.21	**0.08**	**59.06**
High	0.43	49.16	0.29	52.39	**0.25**	**54.05**

Table 4.8: Experiment Result of performance evaluation (MSE and PNSR) of the three thresholding algorithms using a Wiener filter

Global binarization methods are very fast and they give good results for typical scanned documents. However, if the illumination over the document is not uniform, for instance, in the case of camera-captured documents, global binarization methods tend to produce marginal noise along the document borders. Local binarization methods, such as Niblack's algorithm, and Sauvola's algorithm, compute thresholds individually for each pixel using information from the local neighborhood of that pixel. That is why we get good result from the MSE and PNSR for the algorithms of Niblack and Sauvola. Based on the experiments as we have seen above in Table 4.6 Sauvolas's algorithm worked better and faster. An effective pre-processing is followed up by the process of segmentation. Text area segmentation is presented the succeeding subsection.

As Biniam in [25] proposed the best performance and quality document image is the one that is first filtered using wiener filter and binarized using Otsu thresholding. But in our experiment we found that thresholding with Otsu results poor quality. This is because of the image variety in background and having an RGB colored image has variety of intensity throughout the document.

4.2.4 Text Region Extraction

Separation of the text regions from background texture and graphics is an important step of any optical character recognition system for the images containing both texts and graphics. Digital image processing deals with system that perform various operation on digital image to improve the quality of the image by removing noise and unwanted pixels and to obtain intentional information from an image [37]. Image segmentation is a key step in digital image processing that subdivides an image into its constituent region or object that share homogeneous attributes. The main purpose of the segmentation process is to get more information in the region of interest in an image. In this section we separate out interest area form the whole document image which is called text region extraction or text area segmentation.

In this research to separate the text from graphics and pictures (text from non-text components) we use a segmentation algorithm of Morphological Dilation and Connected Component (CC) analysis technique. The algorithm takes advantage of the fact that characters have limited numbers and sizes whereas the shapes of non-text elements are unlimited.

The background is eliminated based on intensity variance which we undertake in binarization. This makes the foreground components distinct from each other. Then the non-text components are removed using various characteristic features of text and graphics/pictures.

4.2.4.1 Connected Component (CC) Analysis

We can apply 4 and 8 connected region algorithm to identify the distinct foreground Connected Components (CC) from background eliminated card images. A CC may be a picture, logo, graphics, noise, or a text region. In the current work, we focus to identify only the text regions.

Too small regions that are unlikely to become text regions and horizontal/vertical lines detected by checking their width, height, and aspect ratio are considered as non-text components. Typically, a text region has a certain range of width to height ratio. So, we consider a CC as a potential text region if range of width to height ratio lies within the range (min, max). We assume that neither horizontal nor vertical lines can be drawn through a logo and it is larger than the largest possible character within the card. Thus, logos and other components satisfying the above specification get eliminated.

We use a MATLAB built in function bwconncomp() returns the connected component found in the binary image and *bwlabel()* to label the connected component. A sample example of the CC labeling is shown below in figure 4.7.

Implementation of CC Labeling:

```
function [cc,num] = ConnectedComp(binary_image)
cc = bwconncomp(binary_image,4); % cc using 4 connectivity
num = cc.NumObjects; % number of connected components
end
```

[L, num] = bwlabel(bw) is also used to return a matrix L of the same size with bw, containing labels for the connected object in bw and num is the number of connected objects found in bw.

Figure 4.7: Sample experimental result of CC labeling

Components width, height, and area analysis is used to identify big connected elements like: images, graphics, logos, etc. and small connected elements like punctuation marks and small dots.

Another important property of text regions is that the number of foreground pixels in a text region is significantly less than that of the background pixels. We consider a certain range of ratio of the foreground pixels to the background for the candidate text regions. When doing this we will eliminate the non-text region by removing it from the image. A certain limitations will happen when we use this CC labeling technique. Sometimes, one character will be labeled into many characters if the characters are not connected to each other. For example, numbers like ፩, ፪ and ፫ will be labeled into three, some special characters like ሸ will be labeled into two. Although, we may get some characters separate from its part in printing or paper degradation ሸ the upper may separate from it, and gets removed during CC analysis see; the figure 4.8 to see how it labeled wrongly.

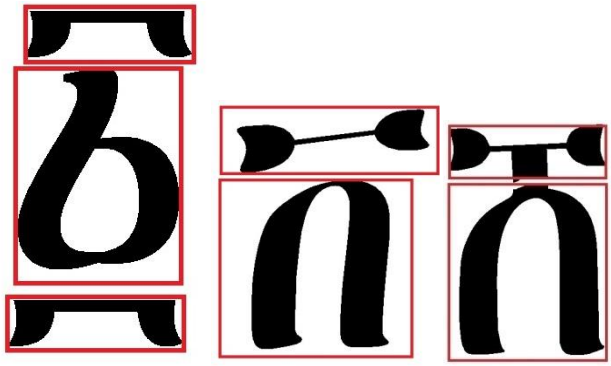


Figure 4.8: Wrongly labeling a text through CC analysis

When text and image/logo are very close to each other, they together often form a single CC and get wrongly classified as background or text.

4.2.4.2 Dilation

Morphology is a vast extent of image processing operations that modifies the images based on shapes. Dilation is a morphological operation that adds pixels at boundaries of the objects or grows the object in size. Dilation takes two parts as data. First one is the input image to be dilated and second is the structuring element also known as kernel. With the help of this structuring element only it determines how much the image is to be dilated. A MATLAB built in function *imdilate(bw, se)* which returns a dilated the input image. The argument se is an array of structuring element object. A sample example of Dilation is shown below in figure 4.9.

Implementation of Dilation:

```
function [dialatedIm] = Dilat(bw,tresh)
d = imdilate(~image, se);
dialatedIm = bwdist(~d) >= tresh;
end
```

For solving the above CC labeling (as shown in the figure 4.8) this study apply dilation to connect the gap or space of the character. We used a vertical dilation technique to connect the gap. We used the structuring element (kernel) se = [0 0 1 1 0 0; 0 0 1 1 0 0; 0 0 1 1 0 0; 0 0 1 1 0 0] for the experiment to dilate the image vertically.

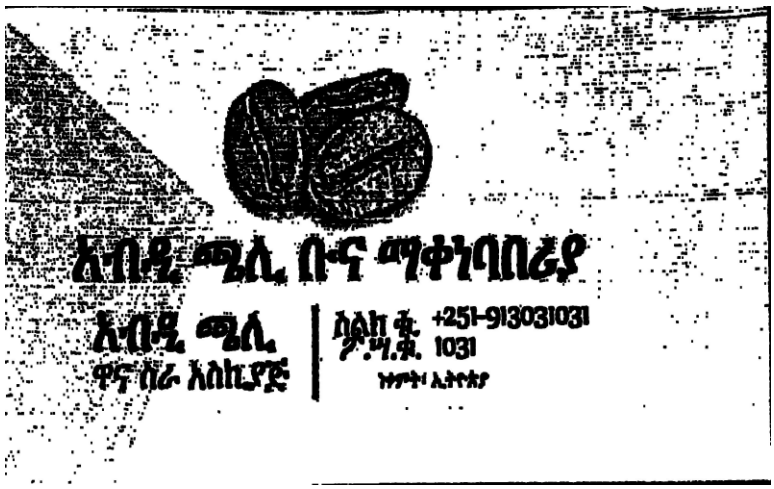


Figure 4.9: Experimental result of dilation over the image of figure 4.8

4.2.4.3 Experimentation on Text Region Extraction

The geometric ratio between the width and the height of the text characters is considered to eliminate possible non-text regions. This ratio value will be defined after experimenting on different kinds of images to get an average value.

The next step performed on CC labeled document image is height, width, and area analysis of component to decide threshold value. The threshold is used to separate the text or graphics or other non-text elements of the image such as logs, lines and so on. However, the experimental threshold decided is based on the aim of segmenting texts from graphics.

Components width, height, and area analysis is used to select the text region from the business card document. In this project, the area regions with 1000 and 35000 are selected as a text region and from them we eliminate the lines that have an area with the range by setting a range that has a long width and small height and has a long height and small width. The smaller pixel we set is 30 and the longer than 80 are eliminated as lines. This method will eliminate the smaller pixel noise and the larger objects (pictures/logos). The text region candidates will be used for further processing.

As Birhanu [13] uses an array to store the height, width, and area of each component label to compare with the other document and to set a threshold value. In this research we adopt this technique to select the text region to set the area region and add some techniques to eliminate lines.

As shown in figure 4.10 in the text region extraction vertical and horizontal lines are eliminated, the logo and small pixel noises are eliminated. The figure below shows some non-text regions are wrongly assigned as text region. Based on the experiment some very small text items may be eliminated from the text region. On the other hand, the graphics elements that are classified as texts may be subsequently eliminated in further steps like segmentation, recognition, and post-processing. The text region extraction may fall to ignore the character number 1, because it will go see it as a vertical line.

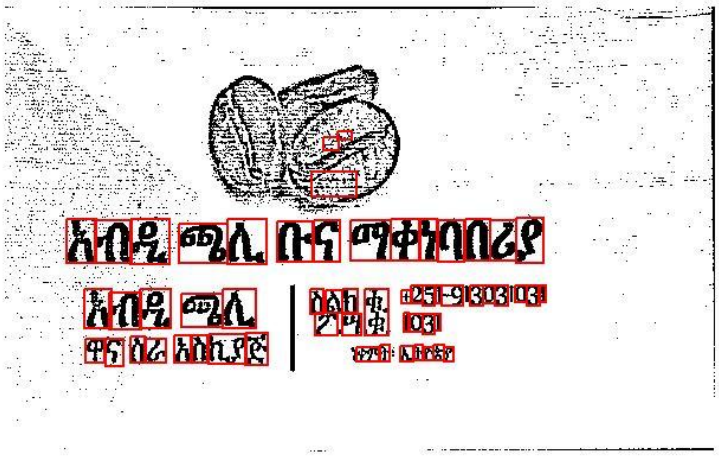


Figure 4.10: Experimental result of the Text Region Extraction

The current technique is not applicable for white texts on a dark background and text in white background together in the same document as shown in the figure 4.11. Because the texts in the image that are white and black background with the current method will not be recognized as a text region.

(a) (b)

Figure 4.11: Example image of white texts on a dark background (a) Original document Image (b) Binarized and skew corrected document image

4.3 Performance Evaluation of the Proposed Technique

The proposed technique is found based on different experimental result of different business cards. The researcher proposed for mobile phone camera captured business card to use a document boundary user assisted homography skew and perspective correction, Wiener filtering, Sauvola binarization, and Connected Component labeling and Dilation to extract the text region from the card.

The input noisy RGB digital image is passed to the skew and perspective correction technique which uses a bounding rectangle user assisted homography that the user have to select the vertex of the business card. The proposed technique will rectify the image from the original position to the frontal view by using the homography and the vertex of the business card. This technique will help the OCR system not only for correcting the perspective or the skew it is also used to eliminate the border noise and other external images that appear around the business card.

After the image skew and perspective corrected the noise removal algorithm Wiener which performs good in the experiment used and then the Sauvola binarization algorithm is used for thresholding the text from the background. Finally for the text region extraction, we propose Connected Component analysis and Dilation algorithm. In CC analysis we apply the height, width, and area analysis to differentiate the text from the logo, picture, and other noise that are found in the image. The proposed system outputs the candidate text region for next stage segmentation and character recognition.

The performance of the proposed technique is evaluated based on the precision rate and the recall rate of the text region extracted.

To quantify the text region separation precision and recall we have designed the following method. Business card components have a text and non-text region. The non-text region includes background texture, logos, pictures, and noises. All the remaining connected regions are identified as texts. While estimating the classification precision and recall four different situations may arise in the text region (TT, BB, BT, and TB).

We will put all the non-text regions as a Background (B) and the text regions as Text (T). Table 4.7 shows all such possible cases.

CC/Region	**Identified as**	**Justification**
Background	Background	BB(True)
Background	Text	BT (False)
Text	Background	TB (False)
Text	Text	TT (True)

Table 4.9: Justification of classification rules

The text region extraction Precision and Recall of the current technique is defined in Eq. 3.18 and 3.19. BB is considered as true classification of non-text, TT is considered as true classifications of text and BT, TB are misclassified ones. BT is wrong classification of background as a text and TB is wrong classification of the text as a background.

The evaluation of the system performance with the selected noise filter to other thresholding algorithms for low, medium, and high level noise is shown in Table 4.8, 4.9 and 4.10. The average results of text region extraction at different noise level business card documents as we have seen Wiener with Sauvola have highest precision and recall.

	Wiener with Otsu	Wiener with Niblack	Wiener with Sauvola
Precision	85.46	90.76	93.6
Recall	87.70	92.56	100

Table 4.10: System performance of thresholding for low-level noisy document images

	Wiener with Otsu	Wiener with Niblack	Wiener with Sauvola
Precision	84.92	90.23	92.1
Recall	87.43	91.03	100

Table 4.11: System performance of thresholding for medium-level noisy document images

	Wiener with Otsu	Wiener with Niblack	Wiener with Sauvola
Precision	79.89	87.32	91.76
Recall	85.77	90.97	99.78

Table 4.12: System performance thresholding for high-level noisy document images

The precision and recall of a text region extraction of the proposed skew and perspective corrected image increases. The result is shown in the table 4.11. The average system performance is shown an increase in precision and recall in the skewed and perspective corrected image.

	Skewed and Perspective Distorted Image			Skew and Perspective Corrected Image		
	Low level	Medium level	High level	Low level	Medium level	High level
Precision	90.87	90.5	54.43	93.6	92.1	91.76
Recall	98.40	98.21	95.86	100	99.9	99.78

Table 4.13: Average system performance for low, medium, and high-level noisy document images before and after skew and perspective correction

For document images that contain small amount of noise, the performance of the system without skew correction shows 90.87% Precision and 98.40% Recall. After the proposed skew and perspective rectification applied a 93.6% Precision and 100% Recall is registered.

4.4 Findings and challenges

This study attempted to show what preprocessing algorithms are good for documents that are captured by a mobile phone/digital camera. We found a best and easy way to rectify a distorted or skewed document image by a support from the user to select the four vertex of the business card, using an advantage of the shape and size of the business card/document image. The noise removal technique Wiener performs well in our scope area which removes a blur from the image. And Sauvola thresholding has shown an attractive result on the captured document which face lighting problem, we propose to use this algorithm when we are inputting a document which has different type of texture. For example if the document have a complicated graphics, logo, picture and designs.

The challenge is to find and determine the skewness of the text, when we scan a document by a desktop scanner there is no additional surrounding noise to the image. But when we scan it by the mobile camera there will be a lot of noise and external image found in the documents which are the surroundings of the business card. So finding and determining the skewness of the text is challenging. That is why we went to use a document boundary rectification. However, correcting the card based on its boundary is also challenging because of finding the bounding rectangle of the business card. Based on this challenge we propose to use a user support for the rectification of the text skewness.

The other challenges we face in this study are that in finding the candidate text region from the image. The proposed technique may find the correct text region as a non-text region (True Negative) and find a correct non-text region as a text region (False Positive). The selected non-text region may be eliminated in the segmentation or post-processing stage but the text regions that are selected as non-text region will be ignored and missed.

Chapter Five

Conclusion and Recommendation

5.1 Conclusion

A large number of people are using smart phone in the country Ethiopia and business card is means of communication with each other. In everyday life we exchange these cards as an introduction to one another people enter this address to their mobile phone each time. Digitalizing these cards to access information from it has been given no emphasis in Amharic printed card.

In the current work, we have exploited different approaches to address the different problems that arise during the scanning process and the noise of the document itself and proposed a fairly accurate preprocessing methodology for camera captured business card images. A series of experiments were conducted to select the optimal noise removal algorithm and a combination of noise removal and thresholding techniques was done to select the optimal thresholding. A Wiener filtering is selected with highest PSNR of 59.10 (low level noise), 45.52 (medium level noise) and 42.36 (high level noise). A Sauvola thresholding tested with the Wiener filtered image is selected with highest PSNR of 59.75 (low level noise), 59.05 (medium level noise) and 54.05 (high level noise).

For the documents which have low level noise, the proposed approach has achieved an overall 93.60% precision and 100% recall in detecting the text region. It is evident from the experiment that the computationally efficiency of skew correction algorithm contributes to an average of 2.73% Precision and 1.6% Recall rise in the overall computation time.

The current work may be viewed as a significant step towards development of effective segmentation/recognition technique for designing a complete and efficient Amharic BCR (Business Card Reader) system for mobile devices. The different capacity of the mobile cameras the scanned image quality will vary and mobile captured image suffers a number of limitations such as; perspective distortion and lighting the present approaches need improvement in the future.

5.2 Recommendation

The current research found an attractive image preprocessing technique for captured business card. To improve the effectiveness and efficiency the following recommendations are forwarded by the researcher.

- When we get a misaligned image during capturing, that is when a camera captured part of the card text. We loss of an important data from the card, so research has to be done to handle this which has to be in the capturing process.
- We recommend to handle the preprocessing and all other recognition technique in the mobile phone because the increase in capability of mobile devices in memory and speed now and is expected to grow further in the future.
- Our scope of the captured card is on day light only and not using a flashlight during scanning. For this lighting condition additional preprocessing technique is required.
- Blur of the image is handled by proposed technique but it decreases the quality of the text. A further research has to be done to control the blur (out of focus) before happening at the time of scanning.
- The current technique is not applicable for white text on a dark background. Technique to handle this has to be explored.
- In skew detection if the text found on the business card is bending or have some degree in rotation in the printed card the current work doesn't correct this. Therefore an improvement needs to be explored.
- The developed text extraction technique extracts a logo or a picture with same size as a text and ignores a very small text found in the card. A detailed research has to be done to handle it.
- This research is done on preprocessing on the mobile captured document images, therefore there is a need to research on a digital camera captured document images because of their capacity and the difference in terms of the quality of photos taken in a variety of environments.

References

[1] Jian Liang, David Doermann, Huiping Li, "*Camera-based analysis of text and documents a survey."* IJDAR, Vol 7, 2005, pp. 84-104

[2] Ayatullah Faruk Mollah, Nabamita Majumder, Subhadip Basu and Mita Nasipuri, "*Design of an Optical Character Recognition System for Camera based Handheld Devices."* ,IJCSI Vol. 8, No 1, 2011, pp. 90-98

[3] Atena Farahmand, Abdolhossein Sarrafzadeh, and Jamshid Shanbehzadeh, "*Document Image Noises and Removal Methods"*, IMECS Vol I, 2013, pp. 431-436

[4] Atishay J., Akriti D., Rachit G., Nitin J., Pooja T., "*Document Image Noises and Removal Methods."*, IJIRS Vol 2, 2013, pp.87-101

[5] Wojciech Bieniecki, Szymon Grabowski and Wojciech Rozenberg, "*Image Preprocessing for Improving OCR Accuracy"*, MEMSTECH, 2007, May 23-26, pp.75-80

[6] Xose R. De La Puente, Ismael Hasan (no date), mOCRa: *Mobile OCR Application, FEDER* [Online] Available: http://www.ceur-ws.org/Vol-697/2.pdf [May 21, 2016].

[7] Maya R. Gupta, Nathaniel P. Jacobson, Eric K. Garcia, "*OCR binarization and image pre-processing for searching historical documents"*, Pattern Recognition 40, 2007, pp. 389 – 397

[8] Yaregal Assabie Lake, "*Optical character recognition of Amharic text: An integrated approach"*, M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2002

[9] Anand J, Mi Zhang Ritesh K, Karthik D, Sameera P and Gaurav S. Sukhatme (no date), "*OCRdroid: A Framework to Digitize Text Using Mobile Phones"*, University of Southern California, Los Angeles, CA 90089, USA

[10] Ravina Mithe, Supriya Indalkar, Nilam Divekar, "*Optical Character Recognition"*, IJRTE Volume-2, Issue-1, 2013, pp. 72-76

[11] Million Meshesha, C. V. Jawahar , "*Recognition of Printed Amharic Documents"*, International Institute of Information Technology, 2000, pp.547-553

[12] Dereje Teferi, "*Optical character Recognition of Typewritten Amharic Text",* M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 1999

[13] Berhanu Sahle, "*Segmentation of Real Life Amharic Documents for Improving Recognition",* M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2015

[14] K.H. Aparna and V. S. Chakravarthy, "*A Complete OCR system Development of Tamil magazine documents",* Tamil internet, Chennai, Tamilnadu, India, 2003, pp. 45-51

[15] Megan Elmore and Margaret Martonosi, "*A morphological image preprocessing suite for OCR on natural scene images"*, International Conference, 2008

[16] Mesay Hailemariam, *"Line Fitting to Amharic OCR: The case of postal address"*, M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2003

[17] Dr John Cowell and Dr Fiaz Hussain, *"Amharic Character Recognition using a fast signature based algorithm", 2003*

[18] Yaregal Assabie and Josef Bigun, "*Offline handwritten Amharic word recognition"*, Pattern Recognition Letters 32, 2011, pp. 1089–1099

[19] Worku Alemu, "*The application of OCR Techniques to the Amharic Script"*, M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 1997

[20] Maria Christina, Melissa Juanillo, and Rosmina Joy, "*Camera-Captured writing system Recognition of Logosyllabic Han Character"*, International Journal of Computer and Communication Engineering, Vol. 3, No. 3, May 2014, pp. 166-171

[21] M. L. Bender, R. L. Cooper, and C. A. Fergusen, "*Languages in Ethiopia"*, ERIC 1976 pp. 191-209

[22] Michael, Abebaw, "*Recognition of Real-Life Amharic Document Images"*. M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2014

[23] Mathieu Delalandre, "*Performance Evaluation of Document Image Analysis",* A Primer IJDAR, 2010, pp. 187-207.

[24] Dr. Deepa Gupta, Leema Madhu, "*Improving OCR by effective pre-processing and segmentation for Devanagiri Script: A quantified study"*, JATI. Vol. 52 No.2, 2013, pp. 142-153

[25] Biniyam, Asnake, "*Retrieval from Real-Life Amharic Documents",* M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2012

[26] Abay Teshager, "*Amharic character recognition system for printed real life document",* M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2010

[27] M. Zahid, Ashraful Amin, Hong Yan, "*Rapid Feature Extraction for Optical Character Recognition"*, Manuscript Draft, June 4, 2012

[28] Qing Chen, "*Evaluation of OCR Algorithms for Images with Different Spatial Resolutions and Noise"*, M.S Thesis, Ottawa-Cartleton Institute for Electrical Engineering, University of Ottawa, Ottawa, Canada, 2003

[29] Youssef and Mohammed, "*OCR Post-Processing error correction algorithm using Google's online spelling suggestion"*, ISSN 2079-8407, Vol. 3, No. 1, January 2012, pp.90-99

[30] Sertse Abebe, "*Bilingual script identification for OCR of Amharic and English printed documents",* M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2011

[31] L. J. Galbiati, "*Machine Vision and Digital Image Processing Fundamentals",* Prentice-Hall, Englewood Cliffs, New Jersey, 1990, pp. 747-753

[32] Line Eikvil ,*Optical Character Recognition*, Norsk Regnesentral, P.B. 114 Blindern, N-0314 Oslo December 1993

[33] M. Cheriet, N. Kharma, C. -L. Liu and C. Y. Suen, "*Character Recognition Systems: A Guide for Students and Practitioner"*, John Wiley & Sons, Inc. 2007, pp. 5-53

[34] Thomas M. Breuel (no date), *Robust Least Square Baseline Finding using a Branch and Bound Algorithm*, Palo Alto, CA 94304, USA

[35] Sonia Bhaskar, Nicholas Lavassar (no date), Scott Green, *Implementing Optical Character Recognition on the Android Operating System for Business Cards*, EE 368 Digital Image Processing

[36] A. F. Mollah, S. Basu, M. Nasipuri, D. K. Basu, "*Text/Graphics Separation for Business Card Images for Mobile Devices"*, IIWGR, 2009, pp. 263-270

[37] Diya Chudasama, Tanvi Patel, Shubham Joshi, Ghanshyam I. Prajapati, "*Image Segmentation using Morphological Operations"*, IJCA *(0975 – 8887) Volume 117 – No. 18, May 2015, pp. 16-20*

[38] T.Romen Singh, Sudipta Roy, O.Imocha Sing, Tejmani Sinam, Kh.Manglem Singh, "*A New Local Adaptive Thresholding Technique in Binarization"*, IJCSI, Vol. 8, Issue 6, No 2, November 2011, pp. 271-278

[39] Samantha Patricia Bail, "*Image Processing on Mobile Platform*", M.S Thesis, Faculty of Engineering and Physical Sciences, University of Manchester, Manchester, England, 2009

[40] N.Sakthivel, L.Prabhu, "*Mean – Median Filtering For Impulsive Noise Removal"*, IJBAS, Vol. 02, No. 04, April 2014, pp. 47-57

[41] Priyanka Kamboj and Versha Rani, "*A Brief Study of Various Noise Model and Filtring Techniuques"*, JGRCS, Volume 4, No. 4, April 2013, pp. 166-172

[42] Malothu Nagu, N.Vijay Shanker, "*Image De-Noising By Using Median Filter and Wiener Filter"*, IJIRCCE, Vol. 2, Sep 2014, pp. 5641-5650

[43] Gedion Asefa, "*Page Segmentation in Amharic Document image collection"*, M.S Thesis, Department of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2013

[44] L.Jagannathan and C,V.Jawahar (no date), *Perspective Correction methods for camera based document analysis*, India, pp. 148-155

[45] Chandan S. Nitin B. Amandeep K., *"Hough transform based few skew detection and accurate skew correction methods"*, Pattern Recognition 41, 2008, pp. 3528 - 3546

[46] Joost V. Faisal S. Thomas M., "*Combined Orientation and Skew Detection Using Geometric Text-Line Modeling"*, Technical University of Kaiserslanutern, Germany, 2010

[47] Jonathan J., "*Document image skew detection: Survey and annotated bibliography"*, Scientific, 1998, pp. 40-64

[48] Abbyy Mobile OCR Engine [Online], Available: http://www.abbyy.commobile-ocr/ [April 1, 2016].

[49] Geetha Kiran A and Muralif (2013), *Automatic Rectification of perspective distortion from a single image using plane homgography,* IJCSA Vol.3, No.5, 2013, pp. 47-58

[50] Webopedia (2016), Available: http://www.webopedia.com/TERM/O/optical_character_recognition.html, [June 2 2016]

[51] W. Lin, R. Tapamo, B. Ndovie, *"A Texture-based method for document Segmentation and Classification"*, ARIMA/SACJ, No. 36. 2006, 49-56

[52] Sneha Sharma, "*Extraction of Text Region in Natural Images"*, M.S Thesis, Department of Computer Science, Rochester Institute of Technology, Rochester, USA, 2006

Annex

Annex I.

MATLAB code for Mean Filter

```
Function [outputImage] = mean_filter(image, varargin)
%The input argument image is a grayscale image
%AVERAGEFILTER 2-D mean filtering.
image = rgb2gray(image);
numvarargs = length(varargin);
if numvarargs > 3
        error('myfuns:somefun2Alt:TooManyInputs', ...
         'Possible parameters are: (image, [m n], threshold, padding)');
end
optargs = {[3 3] 0};            % set defaults for optional inputs
optargs(1:numvarargs) = varargin;
[window, padding] = optargs{:}; % use memorable variable names
m = window(1);
n = window(2);

if ~mod(m,2)
    m = m-1; end       % check for even window sizes
if ~mod(n,2)
    n = n-1; end

if (ndims(image)~=2)            % check for color pictures
    display('The input image must be a two dimensional array.')
    display('Consider using rgb2gray or similar function.')
    return
end

% Initialization.
[rows, columns] = size(image);   % size of the image

% Pad the image.
imageP  = padarray(image, [(m+1)/2 (n+1)/2], padding, 'pre');
imagePP = padarray(imageP, [(m-1)/2 (n-1)/2], padding, 'post');

% Always use double because uint8 would be too small.
imageD = double(imagePP);

% Matrix 't' is the sum of numbers on the left and above the current cell.
t = cumsum(cumsum(imageD),2);

% Calculate the mean values from the look up table 't'.
imageI = t(1+m:rows+m, 1+n:columns+n) + t(1:rows, 1:columns)...
    - t(1+m:rows+m, 1:columns) - t(1:rows, 1+n:columns+n);
 % Now each pixel contains sum of the window. But we want the average value.
imageI = imageI/(m*n);

% Return matrix in the original type class.
 outputImage = cast(imageI, class(image));
end
```

Annex II.

MATLAB code for Sauvola Thresholding

```
function output=sauvola(img,varargin)

    % Initialization
    img = rgb2gray(img);
    numvarargs = length(varargin);
    if numvarargs > 3
        error('myfuns:somefun2Alt:TooManyInputs', ...
         'Possible parameters are: (image, [m n], threshold, padding)');
    end
    optargs = {[25 25] 0.4 'replicate'}; % set defaults
    optargs(1:numvarargs) = varargin;   % use memorable variable names
    [window, k, padding] = optargs{:};
    if ndims(image) ~= 2
        error('The input image must be a two-dimensional array.'); 121
    end
    % Convert to double
    image = double(image);
    % Mean value
    mean = mean_filter(image, window, padding);
    % Standard deviation
    meanSquare = mean_filter(image.^2, window, padding);

    mean1 = mean.^2;
    if meanSquare > mean1
        deviation = (meanSquare - mean.^2).^0.5;
    else
        deviation = (mean.^2 - meanSquare).^0.5;
    end

    % Sauvola
    R = max(deviation(:));

    threshold = mean.*(1.3 + k * (deviation / R-1));
    outputImage = (image > threshold);

end
```

Annex III.

MATLAB code for Niblack Local Adaptive Thresholding

```
function [outputImage] = Niblack(img,varargin)

filt_radius = 70; % filter radius [pixels]
k_threshold = 0.01; % std threshold parameter

%% load the image
i = rgb2gray(img);
X = double(i);
X = X / max(X(:)); % normalyze to [0, 1] range

%% build filter
fgrid = -filt_radius : filt_radius;
[x, y] = meshgrid(fgrid);
filt = sqrt(x .^ 2 + y .^ 2) <= filt_radius;
filt = filt / sum(filt(:));

%% calculate mean, and std
local_mean = imfilter(X, filt, 'symmetric');
local_std = sqrt(imfilter(X .^ 2, filt, 'symmetric'));

%% calculate binary image
outputImage = X >= (local_mean + k_threshold * local_std);

end
```

Annex IV

MATLAB code for Skew and Perspective Correction

```
Function [correctedImage] = userAssistedSkewPerspectiveCorr(image)

figure, imshow(image);
% Corner selection must be clockwise or anti-clockwise

[X Y] = ginput(4);
% Select the four corners of the skewed or distorted perspective image

[X Y] = sortPolyFromClockwiseStartingFromTopLeft( X, Y );

x=[1;2400;2400;1];
y=[1;1;1500;1500];

A=zeros(8,8);
A(1,:)=[X(1),Y(1),1,0,0,0,-1*X(1)*x(1),-1*Y(1)*x(1)];
A(2,:)=[0,0,0,X(1),Y(1),1,-1*X(1)*y(1),-1*Y(1)*y(1)];

A(3,:)=[X(2),Y(2),1,0,0,0,-1*X(2)*x(2),-1*Y(2)*x(2)];
A(4,:)=[0,0,0,X(2),Y(2),1,-1*X(2)*y(2),-1*Y(2)*y(2)];

A(5,:)=[X(3),Y(3),1,0,0,0,-1*X(3)*x(3),-1*Y(3)*x(3)];
A(6,:)=[0,0,0,X(3),Y(3),1,-1*X(3)*y(3),-1*Y(3)*y(3)];

A(7,:)=[X(4),Y(4),1,0,0,0,-1*X(4)*x(4),-1*Y(4)*x(4)];
A(8,:)=[0,0,0,X(4),Y(4),1,-1*X(4)*y(4),-1*Y(4)*y(4)];

v=[x(1);y(1);x(2);y(2);x(3);y(3);x(4);y(4)];

u=A\v;

U=reshape([u;1],3,3)';

w=U*[X';Y';ones(1,4)];
w=w./(ones(3,1)*w(3,:));
T=maketform('projective',U');

% The selected co-ordinate image is transformed to the correct one
P2=imtransform(image,T,'XData',[1 2400],'YData',[1 1500]);

correctedImage = P2;

end
```

```
function [X, Y] = sortPolyFromClockwiseStartingFromTopLeft( X, Y )

% The 1st 2 high values for the y-axis are the top 2 edges
% <upper corners identified>
top_vertices_Y = sortCoordinatesAccordToX(Y);

    for i=1:4
        for j=1:4
            if top_vertices_Y(i) == Y(j)
                top_vertices_X(i) = X(j);
            end
        end
    end

   top_vertices_X = top_vertices_X';
   X = top_vertices_X;
   Y = top_vertices_Y;

   % The larger of the x values for the 1st 2 high values for the y-axis
   % belongs to the top-right hand corner
   % <upper left and right corners identified>
   if X(1) > X(2)
       top_vertices_X(1) = X(2);
       top_vertices_Y(1) = Y(2);
       top_vertices_X(2) = X(1);
       top_vertices_Y(2) = Y(1);
   end

   X = top_vertices_X;
   Y = top_vertices_Y;

% The larger of the x values for the last 2 high values for the y-axis
% belongs to the bottom-right hand corner
% <lower left and right corners identified>
   if X(3) < X(4)
       top_vertices_X(3) = X(4);
       top_vertices_Y(3) = Y(4);
       top_vertices_X(4) = X(3);
       top_vertices_Y(4) = Y(3);
   end

   X = top_vertices_X;
   Y = top_vertices_Y;
end
```

```
function [coordinates] = sortCoordinatesAccordToX( coordinates )

X_sorted_coordinates = sort(coordinates);

    for i=1:length(coordinates)
        for j=1:length(coordinates) % scan thru the X_sorted_coordinates set
            if X_sorted_coordinates(j,1) == coordinates(i,1)
            reordered_coordinates(j,:) = coordinates(i,:);
            end
        end
    end

    coordinates = reordered_coordinates;
end
```

Annex V.

MATLAB code for Image Quality Measure

```
% MSE - Mean Square Error
% PSNR - Peak Signal to Noise Ratio
function [MSE, PSNR] = calc_MSE_PSNR(original, preprocessed)
original = rgb2gray(original);
preprocessed = im2double(preprocessed);
original = im2double(original);
[M, N] = size(original);
N = N/3;
acc = 0.0;
for i=1:M
    for j=1:N-1
        acc = acc + (original(i,j) - preprocessed(i,j))^2;
    end
end
MSE = (1/(M * N)) * acc;
PSNR = 10*(log10((255^2)/MSE));
end
```

Annex VI.

MATLAB Code to segment Connected Components

```
function [] = connectedCompSeg(image)
if size(image,3)==3 % RGB image
    image=rgb2gray(image);
end

%bw = ~im2bw(image);
bw = ~sauvola(image);
% Label connected components
[L, ~]=bwlabel(bw);
% Measure properties of image regions
propied=regionprops(L,'BoundingBox');
hold on

figure, imshow(~bw);
% Plot Bounding Box
for n=1:size(propied,1)

rectangle('Position',propied(n).BoundingBox,'EdgeColor','r','LineWidth',2)
end

hold off


end
```

Annex VII

MATLAB Code for Text Region Extraction

```
function [outputImage] = TextAreaSeg(image)

% Convert to gray scale
if size(image,3)==3 % RGB image
    image=rgb2gray(image);
end
% Convert to Binary image
bw = sauvola(image);

% Createing Kernel
se = [0 0 1 1 0 0; 0 0 1 1 0 0; 0 0 1 1 0 0; 0 0 1 1 0 0];

%Dilate the image
d = imdilate(~bw, se);
I = imcomplement(~d);

% Label Conneceted Component and measure property
[Ilabel, num] = bwlabel(I);
```

```
Iprops = regionprops(Ilabel);
Ibox = [Iprops.BoundingBox];
Ibox = reshape(Ibox,[4 num]);
hold on;

% Creating An Array to hold the values of Height, Width and Area
size_info = [0 0 0; 0 0 0];
cc= 1;

% Storing each Height, Width and Area to the Array
for cnt = 1:num
    x = Ibox(:,cnt);
    component_width = x(3,:,1);
    component_height = x(4,:,1);
    component_area = component_width * component_height;
    size_info (cc,1) = component_width;
    size_info (cc,2) = component_height;
    size_info (cc,3) = component_area;
    cc = cc + 1;
end

figure, imshow(bw);
% Selecting the Text Region From the labled Connected Components
for cnt = 1:num
    if (size_info(cnt,3) > 1000 && size_info(cnt,3) < 35000)

rectangle('position',Ibox(:,cnt),'edgecolor','b','LineWidth',2,'LineStyle','
-');
        if (size_info(cnt,2) < 30 && size_info(cnt,1) > 80)

rectangle('position',Ibox(:,cnt),'edgecolor','w','LineWidth',2,'LineStyle','
-');
        else if (size_info(cnt,1) < 30 && size_info(cnt,2) > 100)

rectangle('position',Ibox(:,cnt),'edgecolor','w','LineWidth',2,'LineStyle','
-');
            end
        end
    end
end

outputImage = image;
end
```