Addis Ababa University
College of Natural Sciences
School of Information Sciences

A Generalized Approach to Amharic Braille Recognition

By:
Amanuel Wendmu

Advisor:
Dereje Teferi (Ph.D.)

June, 2015

Addis Ababa University
College of Natural Sciences
School of Information Sciences

A GENERALIZED APPROACH TO AMHARIC BRAILLE RECOGNITION

BY: AMANUEL WENDMU

ADVISOR: DEREJE TEFERI (PHD.)

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

June, 2015

Addis Ababa University
College of Natural Sciences
School of Information Science

A GENERALIZED APPROACH TO AMHARIC BRAILLE RECOGNITION

BY: AMANUEL WENDMU

Name and Signature of the members of examining board

Name	Title	Signature	Date
Dr. Dereje Teferi	Advisor		
Dr. Martha Yefru	Examiner		
Dr. Million Meshesha	Examiner		

Dedicated to my family

ACKNOWLEDGEMENT

First and for most, I would like to thank God, the almighty for making this possible.

I am also thankful to my advisor Dr. Dereje Teferi for his guidance and support in the course of this work.

I also like to thank my friend Berhanu Sahle for the invaluable ideas and comments he provided.

I am also heartily thankful to employees of Misrach Center Ato Geremew and Ato Amare who teach me about braille writing, give their time unreservedly to respond to my questions and provided me the Amharic brialle documents that I have used as a dataset in this work.

My greatest gratitude goes to my loving parents Wendmu Zengu and Letensea Zemichael who brought me to this world, raised and educated me. I also want to thank my brothers Nigusse Wendmu and Fanuel Wendmu and sisters Misgana Wendmu and Milite Wendmu who loved me, shared me their ideas and supported me all the way.

I also would like to thank my fiancé Helina Getachew whose love, care and encouragement have carried me through my difficult times.

Finally, I would like to thank Miftah Hasssen for all the resources he shared and my friends Binyam Gedle, Yonas Tadele and Gedion Assefa who were providing me resources and ideas in the course of my years at AAU.

Amanuel Wendmu

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviation

ABR	Automatic Braille Recognition
AAU	Addis Ababa Univeristy
ANN	Arteficial Nueral Network
BMP	Bit MaP
DPI	Dots Per Inch
EABS	Ethiopian Association for Blind Society
HP	Hawlet Packard
MATLAB	MATrix LABoratroty
OBR	Optical Braille Recognition
OCR	Optical Character Recognition
RGB	Red Green Blue
SVM	Support Vector Machine
UNESCO	United Nations Educational, Scientific and Cultural Organization
WBA	World Blind Association
WHO	World Health Organization

ABSTRACT

In 1825 Louis Braille invented the writing system braille which is the preferred means of written communication for blind people around the world. In Ethiopia, there has been a large production of Amharic braille documents from the time of its introduction in 1934. Since Braille documents are usually read and understood only by visually impaired and those others who have learned braille only, their literatures are highly restricted from reaching the sighted society. Optical Braille Recognition (OBR) is an important technology to bridges the communication gap that exists between the blind and the sighted people. In addition, OBR serves as a technology to preserve documents that only exist in Braille form and can also serve as a simple mechanism for braille copy.

Past researches carried out locally have made an attempt to develop an ABR system for Amharic braille recognition. There were also some efforts made to improve the performance of the Amharic ABR system. As a continuation to those previous efforts, this research presented an approach for recognition of double-sided Amharic braille documents. In this regard, image thresholding and segmentation techniques are explored and adopted in this work with some improvements.

An adaptive thresholding algorithm that uses the pixel values on every image is used to compute two threshold values that segment the Braille image into bright, dark and background. A moving window scanning technique that scan every column of an image and extract the dot regions for the front and back side of the image and put them into two separate arrays is also adopted and improved. In addition, an improved grid construction technique is designed that automatically determines the average dot area and exclude dots which are smaller or bigger in size that can affect the performance of the grid construction. The grid construction process has incorporated techniques for refining itself when any error exists. Braille cells are formulated by using the location of six intersection points in the constructed grid. Finally, a lookup table is developed for every character in Amharic and features of braille cells are matched against corresponding Amharic print characters.

The system is tested using database from scanned clean, average noisy and high level noise Amharic braille documents. The developed system works with 100% accuracy for both single-sided and double-sided clean braille documents. However, as the level of noise increases the performance of that system fails. Overall, an accuracy level of 99.27% for

single-sided and 88.16% for double-sided braille documents is achieved. Degradation of braille document and skewness has been the major challenges. The results achieved show that the work is promising. However, to further improve the system, areas that need further consideration in future such as improved image filtering, skew angle correction, format preservation and post recognition correction are recommended.

CHAPTER ONE

INTRODUCTION

1.1. Background

According to WHO's 2014 statistics, over a billion people, about 15% of world's population, have some form of disability; 285 million people are estimated to be visually impaired worldwide of which 39 million people are blind and 246 million people have low vision. About 90% of the world's visually impaired and blind people live in low income setting [49].

Visually impaired people are part of the society that plays a significant role where they live. Therefore, it is a must to provide this people with a means and system that enable them to communicate with the world. There are different mechanisms created for visually impaired people as a means to provide them what the world has in printed document. One of the most valuable and indispensable method is through the use of braille [3].

1.1.1. Braille Writing System

Braille has been the preferred means of writing by visually impaired people since its introduction in 1825 by the French scientist Louis Braille. The innovation of braille has created an efficient means of communication for the blind and visually impaired people and allowed them access to education and employment [12].

Braille consists of a combination of punched six dots placed in a fixed two by three matrix called cells. Six dot writing is widely used giving a total of 64 (2^6) possible combinations of raised dots in a single cell that can be assigned for different characters or words. Braille has a very low information density that an average page of 26 x 34 cm sheet can only have 27 lines and 32 characters per line. Like the ordinary writing used by sighted people, it is also possible to write on both sides of a braille paper by shifting the text on the verso so that it will not conflict with the recto [26].

Figure 1.1 shows how dots on the verso are placed between the recto dots.

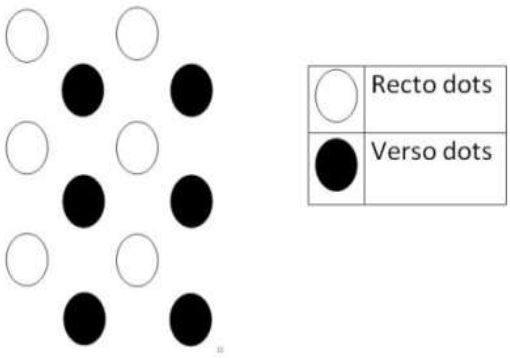


Figure 1.1 Six dot braille positioning of recto and verso [26]

People have different annotations for braille characters that vary between language and domain. But, the space (distance) between points, between characters and between lines is standard as shown in figure1.2 below [3] [4]. The dimension of point in accordance with the braille dimension and tactile resolution of the finger tips of people has been defined and specified for worldwide use. Horizontal and vertical distance between points in the letter, in addition to, the distance between the cells that represent a word after completion are specified by the library of US Congress [23].

- The horizontal distance (Ph) and vertical distance (Pv) between dots within a cell is 2.5 millimeter,
- The horizontal distance or character space (Rh) between cells is approximately 3.5 millimeter,
- The vertical distance or line space between cells is approximately 5 millimeter.

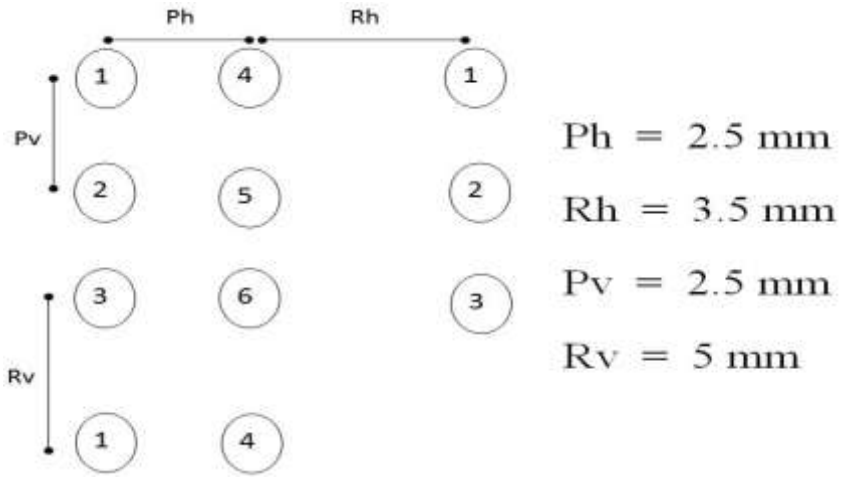


Figure 1.2 Standard distance in braille document [16]

Though the American and France braille styles are the two main types of braille styles, most countries have adopted and/or defined their braille code so as to fit their local language characters. In general, braille systems are categorized into two levels, Grade 1 and Grade 2 [42].

In the Grade 1 braille representation, each possible arrangement of dots within a cell represents only one letter, number, punctuation sign, or special braille composition sign. This is the basic representation of letters where each letter is to be represented by one cell containing two columns where each column contains three rows. In Grade 1 braille, individual cells cannot represent words or abbreviations which make documents produced in this grade of braille to be bulkier [42].

Grade 2 braille was introduced as space-saving alternative to Grade 1 braille. Since braille books are much larger than normal text books a contraction is used to reduce the length of some words. Therefore, in Grade 2 braille, a cell can represent a shortened form of a word. Many cell combinations have been created to represent common words, such as **brl** for braille, **ab** for about, **alr** for already and many others. In addition to reducing the size of books, Grade 2 braille permits faster reading and this makes Grade 2 braille to be preferred over Grade 1[11].

The nature of braille writing has made braille documents to be bulkier. One attempt made to make braille more concise was the introduction of braille contraction; while the other being the introduction of double-sided braille writing. Double-Sided braille writing can help reduce the size of bound braille documents such as books. But, it has an impact on braille recognition performance.

In a given page of inter-point braille there exist protrusions (dots of the recto) and depressions (dots of the verso). In the scanned images of braille, the dots of protrusions appear as a black region from above and a brighter region from below while the depressions appear in reverse order. The major challenge unique to double-sided braille recognition is that regions of the same type; for instance, white region form protrusion and white region from depression, are merged creating confusion in the detection process [4] [50].

1.1.2. Braille Recognition Systems

Over the period, there are many researches being conducted in order to exploit the power of information technology to improve the livings of the society [43]. Accordingly, Optical

Character Recognition (OCR) is a system designed to convert scanned printed or handwritten text image file into readable and editable text document. OCR system receives file as an image and convert it by matching or comparing features against set of characters stored in OCR database [23] [28].

OCR has opened a new light of life for blind people. It can be used to allow text-to-braille and braille-to-text conversion. But, due to the fragmented nature of the characters and the fact that characters of both sides can be seen simultaneously, braille translation in to text cannot be processed with standard OCR software. Therefore, a little different approach, OBR, was devised. OBR is a variation of OCR and is used to digitize and reproduce texts that have been produced with non-computerized systems, such as braille type writers. OBR exploits the fixed matrix positioning property of braille dots [23] [27].

As can be seen in figure 1.3 below, braille recognition system has similar processes as OCR and according to [13] it consists of six operations; Image acquisition, Pre-processing, Segmentation, Feature extraction and Interpretation.

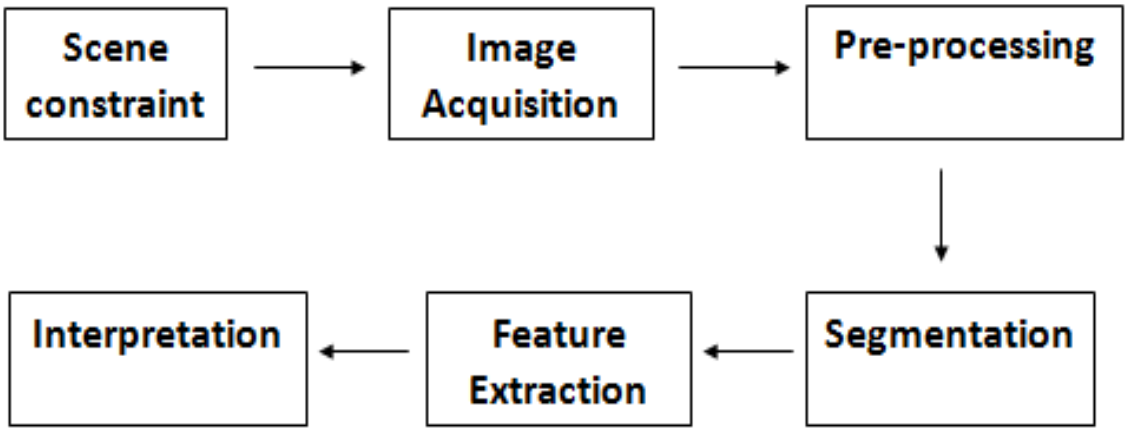


Figure 1.3 Braille Recognition Process [13]

The first phase, scene constraint, has the aim of maximizing the use of prior knowledge of the scene and lowering the problem of image analysis in any way possible. A scene constraint exploits and imposes the environment constraints aiming to reduce the complexity of all subsequent operations to manageable level. This is achieved by maximizing the use of prior knowledge of the scene by exploiting existing knowledge and minimizing the problem of image analysis by effective imposition of constraints [13].

Image acquisition is the process of inputting the braille text to our computers as an image through the use of digital camera or scanner. This process is similar to the way we input traditional text file by using a scanner, a digital camera or some other input device. There are different scanner standards available in the market that makes the digitization process easy. However, the quality of images received may not be appropriate to start processing of these images [4] [13].

The images we get from scanners are not perfect and free of noise. Therefore, before any further braille processing for recognition, preprocessing operations such as noise filtering and edge enhancement, are performed on the images to improve the quality of the image and prepare the pixel values for subsequent operations so that the segmentation, feature extraction and classification process that will follow will be simplified [2] [4].

Segmentation separates the preprocessed image in to foreground (dots) and background regions. There are different segmentation techniques that are particularly applied for braille image. Mesh-grid and local measure are the two most common. Mesh grid widely makes use of the distance rule in braille writing to construct a vertical and horizontal grid on strictly arranged braille Cell [13].Since dots produce different color intensity at top and bottom, local measure works by making use of this difference in intensity level to define a threshold for segmentation [3] [41].

Feature extraction process extracts the braille dots from the segmented image and groups them into cells. There are different feature extraction algorithms. Some algorithms work on the contours of the image while other algorithms calculate every pixel of the image. But, a good feature extraction algorithm is one that can produce an image feature that is robust to transformation and noise as well as efficient in extracting and matching features [13] [40].

The last phase, interpretation, converts the braille cells into their corresponding text. For each dot, the distance between the centeroid and possible neighbors are determined and from this information the dots are grouped into cells. The dot patterns in each cell are represented by a bit string and interpretation is made by searching a corresponding value in the braille dictionary [13].

1.2. Statement of the Problem

In Ethiopia, there are an estimated half a million visually impaired people that include students, teachers, artists, lawyers and employees that have significant contribution in

political, religious, economic, social and many other affairs of the society. Braille is the means of codifying knowledge for these people. But, only a small number of sighted people understand braille. Therefore, the works of visually impaired people is accessed only by those who understand braille. This has created a wide generation gap between the visually impaired and vision society [18] [43].

There is massive collection of Amharic braille documents that have been produced since Amharic braille is introduced in 1934 [52]. These documents are found at different places like Addis Ababa University Kennedy library Braille Documentation, Misrach Center, Sebeta visually impaired school, German Church visually impaired school, Ethiopian Association for Blind Society (EABS), Entoto Blind People School and in different regions of the country. Digitizing of these documents not only facilitates the communication with visually impaired but also help preservation of historic braille documents [18] [41] [43].

Literature reveals that there are a lot of attempts made to develop Braille Character Recognizer for different foreign languages, including English [4] [11], Arabic [2] [16] [24], Spanish [50] and Chinese [30]. As a result of these efforts, some countries are able to build a well developed braille recognition system for their local languages. However, none of the available recognition systems developed so far in foreign countries support Amharic braille. Therefore, it is a must to have OCR system for local languages like Amharic that addresses the unique nature of the language.

The attempt for applying OCR technology in recognition of Amharic braille documents was started in 2009. So far, few researches have been carried out locally towards developing OBR for Amharic. To the best of the researchers' knowledge the work of Teshome [43] was the first attempt in developing Amharic Braille Recognizer. Teshome [43] developed and tested his system on a clean braille document, which is not the case in most real life braille documents due to its tactile nature. As a result, Ebrahim [18] designed a system that explores different noise detection and removal algorithms in order to enhance the performance of the system on real life documents. So, he has used Gaussian filtering with adaptive histogram equalization and morphological operation to detect and remove noise. The effort by Shumet [41] was to enhance the Amharic Braille Recognition system by focusing on feature extraction and classification. With these regard, three feature extraction algorithms: fixed cell measure, horizontal and vertical projection and grid construction were experimented. The test result shows that the fixed cell measure performs very well. Miftah [32] presented an

approach for recognition of Amharic documents using direction field tensor that uses Gaussian filters and derivative of Gaussian for noise removal and isolation of braille dots from background. He has also designed a skewness correction technique, which exploits the horizontal direction nature of braille dots. The most recent work by Berihu[9] was to improve the Amharic braille recognizer and extend it to work for Tigrigna braille recognition as well.

All attempts made towards recognition of Amharic braille so far are on recognition of single-sided braille document. This is because double-sided braille writing was not commonly applied in Amharic braille. But, as Ebrahim [18] mentioned, with the advent of Amharic braille printer, double-sided braille writing is widely being used. Therefore, in order to have a fully function Amharic Braille Recognition System, previous researches recommended designing a system considering double sided documents. As a result, this study has attempted the recognition of double-sided Amharic braille documents which, to the best of the researcher's knowledge, have not been studied so far.

In effort to solve the stated problem, this study has addressed the following research questions:-

- What are the special features of double-sided Amharic braille documents?
- What kind of preprocessing techniques are required to prepare the dataset for recognition?
- What separation technique is suitable to achieve higher performance in separating front and back side dots in double-sided writing?

1.3. Objective of the Study

The general and specific objectives of this study are described here under.

1.3.1. General Objective

The general objective of this project is to explore techniques and algorithms for developing double-sided Amharic braille recognition system that can enable to convert the content of braille into textual format.

1.3.2. Specific Objectives

To meet the general objective stated above the following specific objectives are set and addressed:

- To review previous efforts made towards the development of OCR for Amharic braille document.
- To collect relevant Amharic braille document and prepare the dataset required for training and testing the classifier.
- To identify special features of double-sided braille characters in character image recognition.
- To explore algorithm for recognition of double-sided Amharic braille.
- To develop a prototype for optical braille recognition for Amharic braille.
- To evaluate the performance of the prototype developed.
- Provide conclusion and the way forward for future works.

1.4. Scope and Limitation of the Study

This study is a continuation of previous efforts in the area of Amharic braille recognition. Past researches carried out locally by Teshome [43], Ebrahim [18], Shumet [41] Miftah[32] and Berihu[9]; all attempted recognition of single-sided Amharic braille. The scope of this study limited to exploring techniques suitable for recognition of both single-sided and double-sided Amharic braille documents which have not been attempted so far. To this end, various double-sided braille recognition techniques are explored and the best ones are selected and integrated to the Amharic OBR system.

Amharic braille characters can be represented with either one, two or three characters. In this work we have included all representations and the system is tested on all characters, numerals and punctuation marks in the Amharic writing system. There was no organized document image corpus that can be used in this work. Therefore, to test the developed system, a database that contains both single sided and double sided Amharic Braille document is created. The created database contains braille images classified in to three noise level; clean, medium noisy and high noisy. But, due to limitation in time the test is made only on 30 double-sided and 6 single-sided documents only.

Braille can be written on paper sheet or plastic. Double-sided Braille are prepared only with paper. The embossing process can be done with either a typewriter or a braille embosser. Due to its availability, only braille documents prepared with the help of embosser are used in this work.

1.5. Methodology of the Study

Methodology provides a way to achieve the objectives of a research problem. The type of research design that is used in this study is experimental design. Experimental, also called empirical research is a type of research design that relies on experience or observation alone, than any abstract idea or theory. It is a data based research that requires setting up hypothesis and coming up with facts to prove or disprove the hypothesis and conclusions which are capable of being verified by observation or experiment [10]. The nature of Experimental research design used in this work is mixed qualitative and quantitative. it is qualitative in the sense that is exploratory in nature and provides a complete detailed description of the research topic but it is also quantitative since it focuses on counting and classifying features and constructing models and figures to explain what is observed.

In order to undertake and achieve the objective of this research the following methods were followed.

1.5.1. Literature Review

Literature review is conducted on published information and data relevant to a research question to summarize the state of the art in the field. It provides an up-to-date account and discussion of the researches carried out in a particular topic. It helps to test our research question against what is already known about the subject. Therefore, previous views expressed by Authors in OCR and OBR in general and Amharic Braille Recognition in

particular is collected from published sources such as books, thesis, journal articles, proceeding and the internet and were reviewed and discussed.

1.5.2. Dataset Collection

There is a large collection of Amharic braille document in AAU-Kennedy Library, Misrach Center and Entoto Blind School. These centers have served past researchers in Amharic Braille Recognition by providing Amharic braille Documents to be used as data set.

For the purpose of this research, 30 double-sided and 6 single-sided Amharic Braille documents are collected from Misrach Center and are used as a dataset to experiment the developed system. The collected braille documents are scanned using HP flatbed scanner with a resolution of 200 dpi.

The focus of this research is the recognition of double-sided Amharic braille. But, single-sided braille documents were also collected. This is because it is believed that the system to be developed should be tested and proved to work on a single-sided braille documents as well.

1.5.3. Implementation tool

As discussed above, an HP flatbed scanner is used to create images of braille from the collected braille document. These images are not free from noise and error. Therefore, preprocessing activities such as image dimensionality reduction, noise filtering and intensity reduction, that enable cleaning noise from images of double-sided braille document are applied. The preprocessing activity is performed by integrating new preprocessing techniques with algorithms developed by past researches using MATLAB image processing toolbox. MATLAB is selected because it is easy to use and have extensive library packages and commands for image processing.

In addition to the preprocessing activity, the segmentation operation, that uses thresholding and local analysis to segment the dots from background, and feature extraction processes, that uses horizontal and vertical projection technique with grid construction are also performed on MATLAB. However, for the simplicity of operation with Unicode characters Python is used to match features extracted against their equivalent Amharic textual representation.

1.5.4. Testing Procedure

The collected Amharic braille documents are classified into three dataset. Every dataset contains images of twelve Amharic Braille documents, 10 double sided and 2 single sided. The first dataset contains images of clean Amharic braille document while the second contains images of medium noisy Amharic braille documents while the third contains high level noisy Amharic braille documents.

The system was experimented and its performance was tested at two stages. The first test was done after completion of segmentation process to measure the number of correctly segmented dots against number of dots in the original text. The other test was performed to measure the overall performance of the system. At this level we used character level matching to check the number of correctly translated characters against the total number of characters on the original braille page.

After manually counting the results, an accuracy measure that shows the percentage of correctly detected instances normalized against the expected number of result is used.

There are no standard measures that are good for evaluating OBR system [22]. Various researches have used accuracy to measures the performance of their OBR system. As shown in formula3.2 below accuracy is the percentage of correct instances identified by the system against the expected number of instance [45].

$$Accuracy = \frac{instances\ correctly\ identified}{Total\ number\ of\ instances}$$

Equation 3.2 Accuracy measure [45]

Evaluation of the performance of the Amharic braille recognizer is performed on two stages. After the segmentation process we used manual count of dots to check the accuracy of correctly segmented dots against the total number of dots in the braille sheet. In segmentation we want to classify the image into dot area and non-dot area. But, due to error in the segmentation two kinds of error could arise, a dot region recognized as a background and a background recognized as a dot. Therefore, the performance measure considers the following four results:-

A. A dot recognized to be a dot
B. A dot recognized to be background

C. A background recognized to be a dot
D. A background recognized to be a background

The point of interest in segmentation is the extraction of the dots from the background. For such two class problem the use of a confusion matrix, as in table 1.1, simplifies the problem.

		Predicted class	
		Dot	None Dot
Actual class	Dot	A	B
	None Dot	C	D

Table 1.1 confusion matrix

Therefore, from the confusion matrix we can set out that, accuracy is the sum of A and D normalized against the sum of A, B, C, D.

We can use this same measure to evaluate the performance of the recognizer on character level, which is after translation. To do this, we simply count the total number of characters in the test braille sheet against the number of correct translations made.

1.5.5. Tools Used

MATLAB Image Processing Toolbox

MATLAB image processing toolbox is software that enables to perform image processing, analysis, visualization and algorithm development. It has an extensive list of library packages that allow performing image processing activities that an OCR tool applies such as image property analysis and visualization, noise reduction, enhancement, color and contrast adjustment, transformation and segmentation. One of its advantages is that it supports a diverse set of images types. MATLAB also provides several products that can perform together with the image processing toolbox software and extend its capability [36]. For this reason, the interface building, preprocessing, segmentation and feature extraction processes are done on MATLAB.

Python

Python is an interpreted object oriented and high-level programming language with dynamic semantics. It has simple and easy to learn interface that emphasizes code readability. It is a freely available, widely used general purpose programming language that supports modules and packages [23]. Here in this work, python is used to map numbered feature class labels with equivalent Amharic characters in a dictionary. Both operations, building of the dictionary and matching against equivalent print characters, are performed on python. Python is selected for the benefits mentioned above and for the ease of operation with Unicode characters such as Amharic.

Hardware

The images used in this work as a dataset are scanned using HP flatbed scanner since it is recommended by previous researchers. The Image processing and translation is done by using the above tool running on ordinary windows computer with 2.6 GZ processer and 4GB RAM. But, it is also possible to use computers with lesser capability but can satisfy the lowest requirement for running MATLAB and Python applications.

1.6. Significance of the Study

Braille Character recognition in general allows digitizing hard coded braille documents in to machine readable and editable texts. This can facilitate communication between visually impaired and vision people. The other importance could be, it allows visually impaired people to share their work to others and also enables preservation of historic braille documents and can facilitate text-to-speech conversion.

In Ethiopia there are over half a million visually impaired people. Developing such a system will benefit this people and any one working with them, such as family members, teachers and public organizations. This research in particular will add to the search for Amharic braille Recognition System, which to present time is not fully developed. There were few research works on the area that searched for recognition of single-sided braille document. But, there are a lot of braille document that are embossed in both sides. This research focused on double-sided braille recognition for Amharic Documents and adds significant contribution to the effort in the area.

1.7. Organization of the study

This thesis is organized into five chapters. The first chapter, Introduction, includes; the background, the statement of the problem, objective, scope and limitation, Methodology and Significance of the research.

The second chapter, Literature Review, presents review of previous works in the area of Amharic braille writing and recognition, features of braille characters and characteristics of braille recognition. It has also included general review of international researches in braille recognition and double-sided braille recognition in particular.

The third chapter, Braille Recognition Systems, focused on the general process of braille recognition with explanation techniques and algorithms used to develop the current system involved in each step of recognition. This section will give emphasis to issues and main techniques that can be used for separation of braille dots of double-sided braille.

The focus of the fourth chapter, implementation and experiment, is discussion activities undertaken in implementation of the algorithms developed for the current system. It will also discuss the results achieved and difficulties faced in implementing the current system. The results are presented as recognition rate achieved both at dot recognition and character translation level.

Finally, concluding remarks and recommendations for feature research are forwarded.

CHAPTER TWO

LITERATURE REVIEW

Braille has undergone through many modification and refinements in the past 190 years before it gained worldwide recognition. Braille is a system with many advantages for the users. It has great flexibility and ease of writing and recognition by touch. Any individual can write Braille using a slate and stylus. These materials are relatively easy to get and as a result nearly every blind person in every part of the world can enjoy the benefit of reading and writing independently [42] [47].

2.1. History of Braille

Braille was not the first, or by any means the only method of touch reading. There were a lot other means created before and even after the invention of Braille. The earnest desire of the blind to find access to literature of sighted people opened the door for them which led to many experiments in a variety of media [12].

The first ever recorded attempt to create script for the blind was made at about 1517, by Francisco Lucas of Saragossa, who plotted a simple set of letters carved in thin tablets of wood. This was later improved in Italy at about 1575 by Rampansetto of Rome, which changed the nature of the writing into larger blocks of cut instead of raised. Nevertheless, none of the systems were successful because of the difficulty in writing and reading them.

In 1651, Geoge Harsdorffer of Nuremberge revived the traditional method of writing on wax tablet for use by the blind people through cutting the letters with a stylus. Twenty five years later, Pardre Terzi devised a kind of cipher code based on a system of dots enclosed in square and other figures and also an arrangement of knots tied in strings. There were also many others who tried to create a method of writing and reading using movable raised letters of lead, tin and cast metals. The problem with such systems was the letters were rough to touch and hard to sculpt [12].

Another earlier effort includes the work of Maria Theresa Von Paradise, an Australian music performer and composer, who assisted Valentine Hauy in establishing school for the blind, devised a system of teaching the blind with the aid of pins stuck in cushions. But, when Valentine Hauy founded his school in Paris in 1784, his first student, Francois Lesueur, found

by accident that he could feel the outlines of character “O” which had been strongly impressed on a sheet of paper. Hauy later tried embossing books to teach his student. His student made rapid progress and the system was a big success. However, it was the innovation of this script by Louis Braille in 1825 which devised the system under which the blind read to-day. But, it was not until some fifty years later that the Braille system was universally adopted and, in the mean time, numerous other forms of embossed type were planned and used; some employing lines and dots, others having the form of simplified Roman capitals [12].

Braille got wider acceptance because it is more compact than any other system which proceeded or followed it. It was outstandingly versatile, equally able to express the languages and scripts of any nation. It has been readily adaptable to mathematics, musical notation and other purposes. Its main advantages however lay in the fact that, unlike other embossed types, it could be simply and easily written and read by the blind. It is a remarkably practical script, preferred by the blind, which opened wide the gates to knowledge, Literary employment, the ease to correspond privately with blind friends and wider opportunities for employment for which the blind longed. Despite its manifest advantages, the general adoption of Braille was a slow process [12].

Although, Louis Braille originally devised his writing for his expression of music, Braille was adapted by many other countries that applied his system to various other practical purposes such as the expression of mathematical and chemical symbols, the marking of the faces of watches, meters, gauges, thermometers and playing cards and adapted too, to the outlining of geographical maps and plans of cities and buildings [1] [12].

2.2. Braille Reading and Writing

Braille can be written in several ways. At the beginning it was written manually with stylus which embosses the dots on strong thick paper framed into the slate to facilitate the printing of dots. This raised dots formed in units called cells which are read with the finger tips of blind people. But, currently, Braille is also written with Braille typewrites which can be owned by many visually impaired and blind individuals for their day to day use. Recently, technological developments in computing industry have provided additional means for reading and writing Braille. It has offered software program and portable Braille note taking

devices that enable saving and editing of notes for use in places like lecture room and meetings. These devices also enable displaying back either verbally or tactually. There are also Braille embossers (Braille printer), which can be connected to any kind of computer with the software and produce a hardcopy of a document in the same way ordinary printers work. But, these devices are found only in some organizations with large requirement for Braille production and duplication [1].

The most widely used Braille system consists of six dots and hence, sixty-three (2^6-1) combination of dots can be produced which can be assigned for different characters. In the six dot writing the dots are arranged in a matrix of three rows and two columns. The dots are conventionally numbered. Those dots in the first column are numbered 1-2-3 from top to bottom while the dots of the second column are numbered 4-5-6 in same manner. The numbering is required to facilitate the description of individual symbols and for simplicity of learning [12] [42]. For example, letter "A" is dot 1; letter "B" is dot 1&2; "C", is dot 1&4 and so on. The English alphabets, numerals and the most commonly used punctuation marks with their Braille representation is shown is figure 2.1 below.

Louis Braille arranged the sixty-three Braille cell combinations into a symmetrical group of seven. The first group of ten letters is composed solely from the upper four dots (1-2-4-5). The second ten were formed by adding dot 3 to each of the first ten (1-2-4-5 + 3), the third group of ten is again the same as the first ten, but this time with dots 3 and 6 added (1-2-4-5 + 3-6), while the fourth group of ten was once more the original line with dot 6 only added (1-2-4-5 + 6). The fifth group is a repetition of the first line, but formed from the bottom four dots (2-3-5-6) of the domino instead of the top four. The remaining thirteen signs were composed of right-hand and bottom dots. The grouping of Braille cells is again believed to facilitate Braille order for listing of Braille signs [42]. The complete order of the seven Braille groups is shown in appendix I.

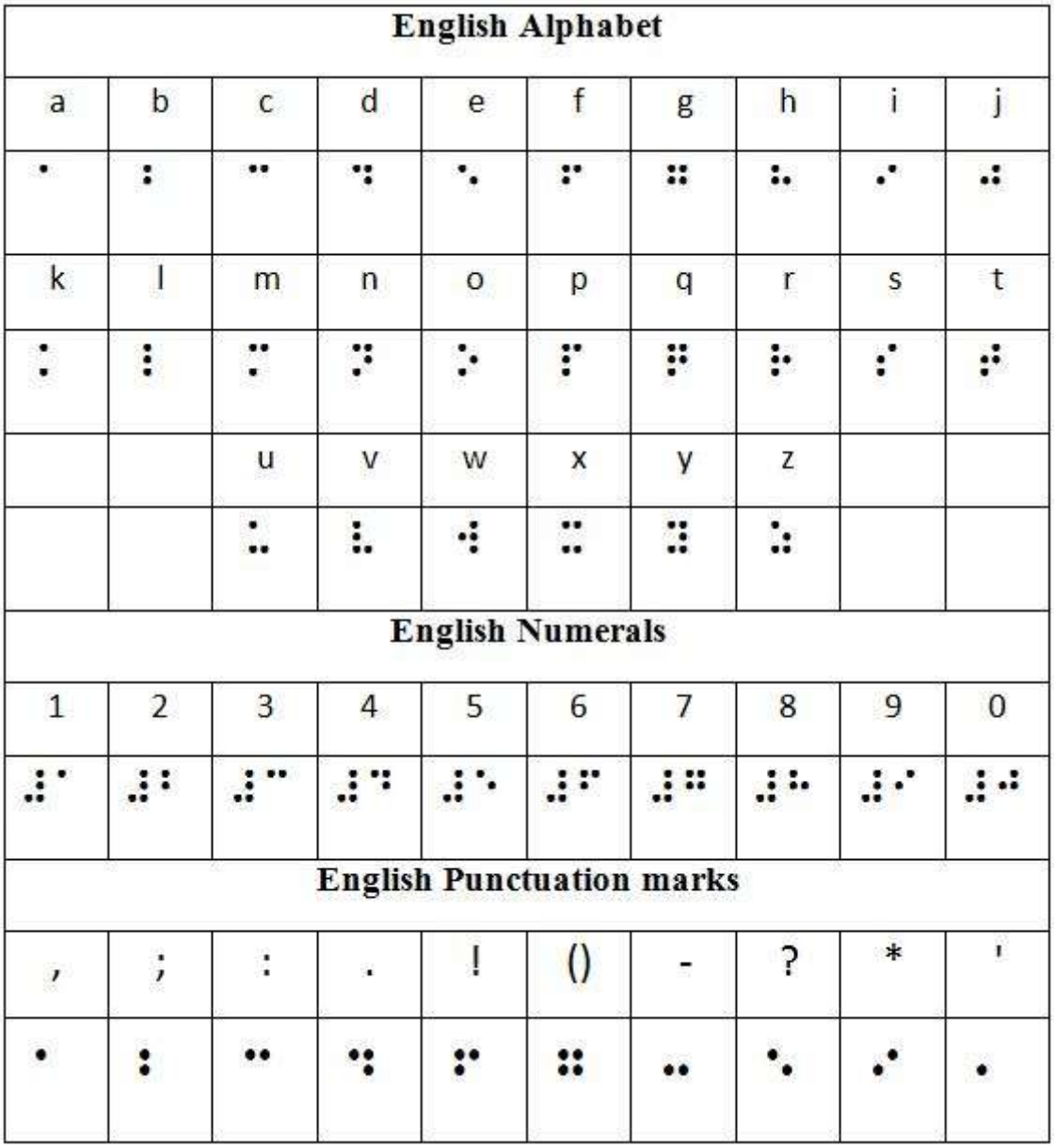


Figure 2.1 English Braille alphabets, numeral and punctuation marks

In English braille, the alphabets take up twenty-six of the sixty-three signs. The first twenty letters are composed from the first two lines of groups. While the remaining five are formed from the first 5 combinations of the third group and the letter "W" which was not found in French alphabet is later represented by dots (2-4-5-6) from the fourth group. Ten are devoted to international punctuation marks, while numbers are represented by the first ten letters but preceded by a special numeral sign composed of dots (3-4-5-6). In case of writing large numbers we only need a single numeral sign at the beginning. The normal Braille numbers are used in sentences or page numbers only. For mathematical calculations, the modified Braille notation, Nemeth code, which is not under the scope of this study, is used. The

remaining twenty-seven combinations are used variously to meet the special needs of individual languages [12].

For a number of languages, two Grades of Braille have been established; Grade 1 and Grade 2 and in some countries cases Grade 3.

Grade 1 Braille

Grade 1 Braille, which is also called uncontracted, is the basic representation of Braille symbols where a single cell represents only one letter, number, punctuation sign, or special Braille composition sign. In this grade of Braille, we also have every letter of every word spelt, letter for letter with the visual script. This grade of Braille is preferred for beginners who are not familiar with contractions. There is also a language rule that disallowed the use of contraction. The disadvantage of this grade of Braille could be that it is resource consuming and as a result fewer books are transcribed in Grade 1[42].

Grade 2 Braille

Grade 2 is the everyday form of Braille writing used for general purpose Braille productions of periodicals, books and letter-writing. It embraces a greater or lesser range of abridged signs for the expression of conjunctions, prepositions, pronouns, prefixes, suffixes, frequently recurring groups of letters and common words. In this system a cell can be used individually or in combination with others to form a variety of contracted or short form of words. Its primary purpose is to reduce the volume of Braille books which is in turn a saving in the cost of production as well as in storage space and cost of distribution. It also saves the braillist some of the effort involved in reading and writing [12] [42].

Few countries and languages have established highly abridged system, usually considered as Grade 3 Braille, in which the original full text is scarcely recognizable and which borders to a true shorthand. But, these are too complex for readers who are not endowed with three qualifications; an extensive command of the language, an excellent memory and a highly sensitive touch [12].

2.3. Double-Sided Braille

Braille has a very low information density. In average, a single page of 26 x 34 cm braille sheet can only hold 27 lines and 32 characters per line [26]. This nature of braille has made

the production of documents in this writing to be voluminous and expensive. Many attempts were made to overcome this problem. The introduction of braille contraction has significantly helped this attempt of making braille documents concise. But, production of braille documents was still bulky and expensive. Another effort taken was the introduction of double-sided writing. Double-sided writing has made it possible to write on both sides of a braille sheet the same way as the ordinary writing used by sighted people.

Double-sided writing or also known as interpoint is an offset printing or writing of braille to allow embossing of the braille dots on both side of a braille sheet with the dots of one side appearing between the dots of the other by shifting the dots on the other side. Double-sided braille writing is found to be very economical to produce and a space saving option in braille writing. It has also helped in reducing the size of bound Braille documents such as books.

However being economical and a space saving option, double-sided writing has created a huge problem for OBR. This is because, in a given interpoint braille exists protrusions (dots of the recto) and depressions (dots of the verso). And in a scanned interpoint braille image, the dots of protrusions appear as a black region from above and a brighter region from below while the depressions appear in reverse order. The major challenge in the recognition process is that regions of the same type; for instance, white region form protrusion and white region from depression, are merged creating confusion in the detection process [4] [50].

2.4. Amharic Writing and Braille System

Amharic has brought its own unique challenges for OBR attributed to the fact that it has its own script that contains large set of characters. Therefore, understanding the unique nature of the script is important in building Amharic Braille Recognition System.

2.4.1 Amharic Writing System

Ethiopia is an East African country with a population of more than 94 million people as per World Bank 2013 statistics [17]. The country is home to many nations and nationalities speaking more than eighty-four indigenous language. All the Languages in Ethiopia are divided into four major groups known as Cushitic, Omotic, Nilo-Saharan and Semitic. Amharic or *Amarénya* is the national language of Ethiopia as well a working language for several regional states within the federal system of the country and as well for non-government organizations and private institution. It is a southwest Semitic language of the

Afro-Asiatic language family. Being a Semitic language of the Afro-Asiatic root, it is related with Hebrew, Arabic and Syrian languages [20] [33].

Amharic is believed to evolve from Geez which originated at around the first millennium B.C. It is brought by immigrants from south Arabia who crossed the Red Sea into northern Ethiopia, present day Eritrea, and mixed with local Cushitic languages and created Geez. Currently Geez is not a spoken language but it is the liturgical language of the Ethiopian Orthodox church. Amharic has also some affinities with Tigre, Tigrinya and the south Arabic dialects. But Amharic has been highly influenced by local Cushitic languages, specially "Oromo" and "Agew" languages, which can be seen from the non-Semitic words it used and the sounds of the character sets of the language [20].

Amharic is spoken principally in the central highlands of Ethiopia. But, in terms of number of speakers, currently it is one of the two major languages of Ethiopia along with "Oromifa". According to 2007 statistics 30% of the Ethiopian population speaks Amharic as a first language and a roughly estimated additional 20% of the population use Amharic as a second language. Therefore, it can be said that half of the Ethiopian population speak Amharic [17] [33]. It is also known that outside Ethiopia Amharic has a large number of speakers in Eritrea and it is also the language for millions of emigrants living in countries such as Egypt, Israel and Sweden [20]. This fact had enabled Amharic to be the second most spoken Semitic language in the world next to Arabic [33].

Amharic has a writing system called 'fidel' which evolved from the writing system for Geez which is known as Ethiopic or Geez otherwise. Ethiopic is a writing system which in turn evolved from the old Sabaean writing system. The Ethiopic script has been in use since the 5^{th} century B.C. But, the oldest extent records for Amharic are found only in songs and poems dating back to the 14^{th} century and literature in any significant quantity did not begin until the 19^{th} century [20].

Amharic Alphabet

According to Baye [21], there are three types of writing systems in the world, namely logographic, syllabic and alphabetic. In the logographic writing system, a single symbol is used to represent one word. It was first used around 5000 years ago near present day Palestine and Syria. The old Chinese writing is also an example of logographic writing system. In the second writing system, syllabic system, a symbol represents neither a word nor

a sound rather a phoneme, which is a combination of vowel and consonant. Therefore, the number of symbols needed for a given language is determined by the number of basic sounds used in the language. The third writing system, Alphabetic system, originated from the syllabic system used in the Semitic languages of Middle East. It is also called the Greek alphabet since its origin is strongly tied to the Greeks. The Latin alphabet is an example which was originally known as the Greek alphabet before the Roman adapted it from Greek and helped its wide usage in their Latin colonies [21].

Most African countries which were colonized by the west use the Latin script. Ethiopia is the only African county that managed to avoid colonization by defeating the Italians during the European scramble for Africa in the 19th century. As a result, it has not used the Latin alphabet or other. It has its own writing system called Ethiopic. Ethiopian writing system differs from the three writing systems discussed above. This system was originally a syllabic system but, unlike syllabic, the vowels inside each phoneme are not represented by a number of vowels but a single vowel [21].

The Ethiopic (Geez) writing system originated approximately 2500 years ago by the Semetic Sabaean people of southern Arabia. The character sets are derived from Sabaean writing, which has had twenty nine, according to baye[26], symbol in its unvocalized shape. It has been in use for longer time in northern part of Ethiopia, particularly, Yeha until the Axumite time whence it gave way to Geez. Amharic took the Geez script and become a written language. It adapted all 26 Geez characters and modified seven of them by placing a bar/hut on top of the characters and increased the number of characters to represent the additional sounds acquired from Cushitic languages. Therefore, Amharic has now 33 core characters where a single character appears in seven orders, one basic and six derived. Therefore, the total number of syllables is 231(33*7) [21]. The non-basic forms are derived through modifying the basic form by adding a circle or a dash at the top, bottom or middle position of the basic character. There are also other symbols representing labialization, numerals and punctuation marks. Figure 2.2 shows part of the Amharic alphabets. The full list of Amharic alphabet can be found on Appendix II.

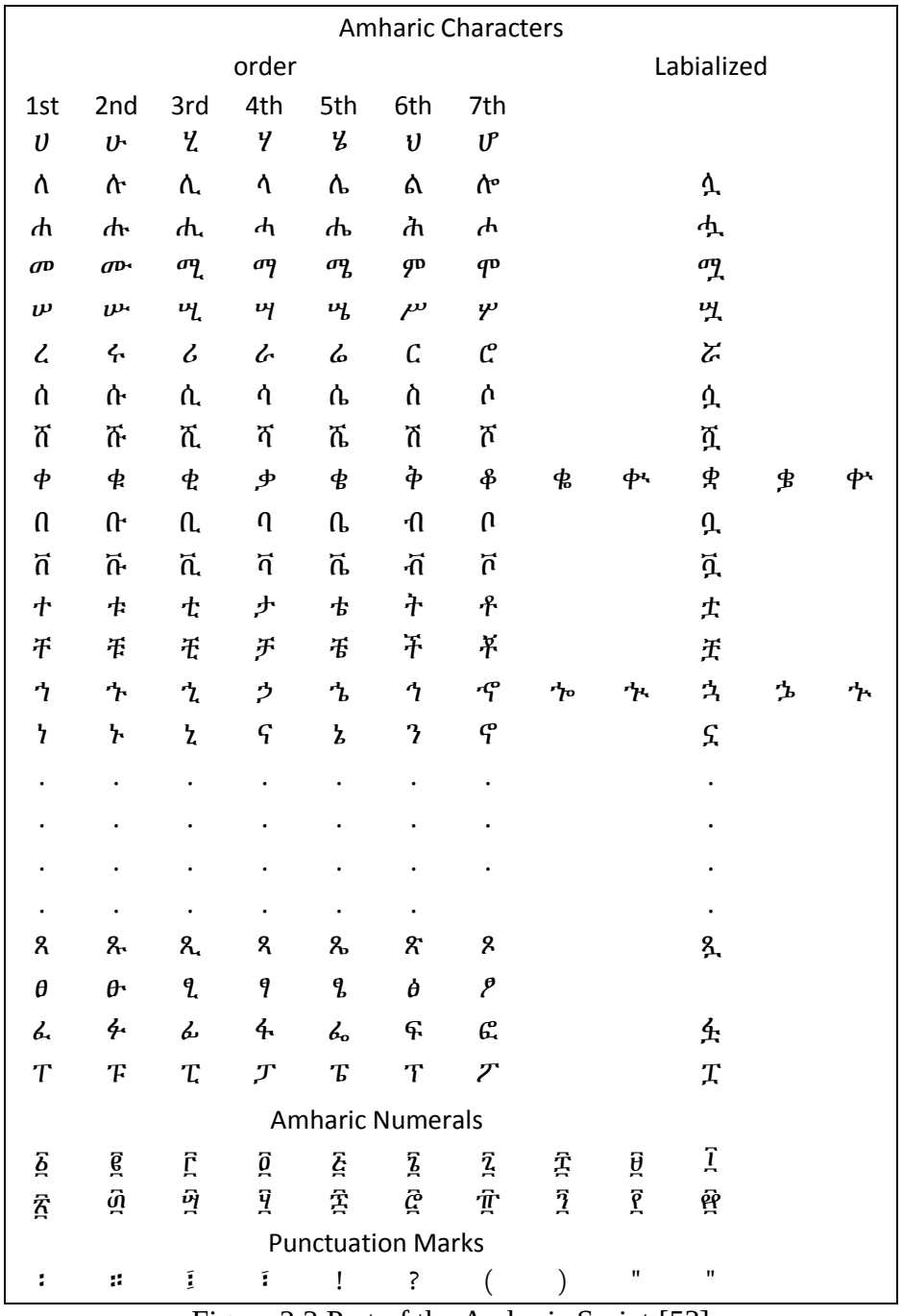


Figure 2.2 Part of the Amharic Script [53]

Amharic Numerals and punctuation marks

Amharic writing system has 20 signs for numerals. But, this numeral system cannot be used for scientific calculations or complex mathematical operations. The problem is because the Amharic numeral system doesn't have symbol for zero, decimal points, negative sign and mathematical operators. Therefore, the Hindu-Arabic numerals and the Latin mathematical operators are used for computational purposes. As can be seen in Figure 2.2 the unique

characteristics in the Amharic numerals are that they are enclosed in a top and bottom frame in order to uniquely identify them from the alphabetic character sets [18].

According to omniglot.com [53], Amharic writing system has six punctuation marks; comma (፣), full stop (።), colon (፤), semicolon (፥), preface (:-) and question mark (፧) which is no longer in use. But there are other punctuation marks which are adapted from Latin based symbols like question mark (?), exclamation mark (!), quotes (“”) and parenthesis. The old south Arabia monumental scripts regularly employ a vertical stroke as a word separator. This style is adapted in Ethiopic texts as (፡) to separate words in a sentence.

2.4.2 Amharic Braille

Many countries have adapted and/or defined Braille code for their local languages. In Africa Braille was introduced with the help of Missionaries and voluntary organizations during the colonial periods. The oldest records of Braille appear for dozen African tribal languages such as Swahili, Kikulu, kikmba, malgachi (Madagascar) Bemba, Chinyanja, Nyanja, Xosa, Shona, Hausa, Mundang, Ibo, Twi and Kabili [12].

Amharic, along with Arabic and Afrikaans is one of the first three languages, which are neither tribal nor linguistically African languages, used in Africa and have their braille forms. Amharic and Arabic were the only languages having their braille notations built with their own symbols. The rest of African braille was formed from European symbols such as English, French, Norwegian and Dutch [12]. Amharic with a braille of comparatively recent date is found in 1924 by American mission in western Ethiopia who together with other voluntary and non government organizations established the first school for blind in “Dembidolo” [12] [52].

The First version Amharic Braille was comprehensive and complete. It contained all Amharic characters including characters with similar sounds, punctuations, numerals, mathematical signs and musical symbols [52]. But, it was subject to many revisions. It was first modified in 1951 in consultation with Dr. A.N. Tucker, D.litt., Ph.D., M.A university of London, Miss Hazel McGeery, American Presbyterian Mission, Sayo, and other workers for the blind in Ethiopia and Sir Clutha Mackenzie, UNESCO Consultant on Braille [12].

According to the 1995 ENAB report [51], a revision was made once to eliminate redundant Amharic Characters which have similar sounds while the other revision was made by other missionaries to replace Ethiopic numerals with Arabic numerals and to further reduce the

variants of Amharic characters. Later in 2001 improvement was made considering issues set by the World Blind Association (WBA) in adopting Braille to local languages. The current version is fourth version which was published in 2001 by ENAB [51].

The Amharic Braille characters are composed of three parts, core and special letters, numerals and punctuation marks [52]. As shown in figure 2.3 below in the fourth version, each Amharic character has variants except the 6th order character. The 6th order character does not require a vowel code to get the required sound of the character. Table 2.1 shows all dots combinations for the vowel variants required in the 4th version Amharic Braille.

Character Variant	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**	**7th**
Vowel	2:6	1:3:6	2:4	1	1:5	N/A	1:3:5

Table 2.1 Fourth version Amharic Braille Variants

ሀ	(125, 26)		መ	(134, 26)		ሰ	(1456, 26)	
ሁ	(125, 136)		ሙ	(134, 136)		ሱ	(1456, 136)	
ሂ	(125, 24)		ሚ	(134, 24)		ሲ	(1456, 24)	
ሃ	(125, 1)		ማ	(134, 1)		ሳ	(1456, 1)	
ሄ	(125, 15)		ሜ	(134, 15)		ሴ	(1456, 15)	
ህ	(125)		ም	(134)		ስ	(1456)	
ሆ	(125, 135)		ሞ	(134, 135)		ሶ	(1456, 135)	
ለ	(123, 26)		ሠ	(234, 26)		ሸ	(146, 26)	
ሉ	(123, 136)		ሡ	(234, 136)		ሹ	(146, 136)	
ሊ	(123, 24)		ሢ	(234, 24)		ሺ	(146, 24)	
ላ	(123, 1)		ሣ	(234, 1)		ሻ	(146, 1)	
ሌ	(123, 15)		ሤ	(234, 15)		ሼ	(146, 15)	
ል	(123)		ሥ	(234)		ሽ	(146)	
ሎ	(123, 135)		ሦ	(234, 135)		ሾ	(146, 135)	
ሐ	(126, 26)		ረ	(1235, 26)		ቀ	(12345, 26)	
ሑ	(126, 136)		ሩ	(1235, 136)		ቁ	(12345, 136)	
ሒ	(126, 24)		ሪ	(1235, 24)		ቂ	(12345, 24)	
ሓ	(126, 1)		ራ	(1235, 1)		ቃ	(12345, 1)	
ሔ	(126, 15)		ሬ	(1235, 15)		ቄ	(12345, 15)	
ሕ	(126)		ር	(1235)		ቅ	(12345)	
ሖ	(126, 135)		ሮ	(1235, 135)		ቆ	(12345, 135)	

Figure 2.3 Part of Amharic Alphabet Braille representation [46]

Amharic uses both Arabic and Ethiopic numerals. Combination of all dots (1:2:3:4:5:6) are used to indicate Ethiopic numerals, while the combination of dots (3:4:5:6) are used to indicate Arabic numerals. The fourth version Amharic braille numeral representation is shown in figure 2.4 below.

Numeral	Corresponding Amharic Braille Representation									
Ethiopic	፩	፪	፫	፬	፭	፮	፯	፰	፱	፲
Arabic	1	2	3	4	5	6	7	8	9	0

Figure 2.4: Fourth Version Amharic Braille Numerals representation

Amharic Braille also consists of punctuation marks most of which are directly adopted from English [32]. The only punctuation marks required are full stop and interrogation mark (2-3-6). It is also to be noted that Braille dots should not be placed between the words as is the custom visual Amharic text. Capital letters are not used; except for "ዕ" which will be proceeded by Dot6. The 4th version Amharic braille representation for punctuation marks is shown in figure 2.5 below.

Punctuation	→	←	↑	↓	*	.	( )	[	]	…
Braille code										
Punctuation	፤	፡	!	÷	፦	:	"	-	—	/
Amharic Braille code										

Figure 2.5 Fourth version Amharic Braille Punctuation signs

This study will focus on the fourth version of Amharic Braille and sample Braille code for the characters is shown in figure 2.3. The complete list of Amharic braille versions can be found in Appendix III, IV, V and VI.

2.5. Automatic Braille Recognition

Braille recognition system is believed to simplify the communication between visually impaired and vision society. This system can allow visually impaired people to access literatures of vision society while at the same time enable them to share their works and to preserve for future access. In recent years, a lot of researches are being carried out on the area of Braille recognition locally and internationally.

Before converting a hard coded braille into a computer editable and printable text, an ABR system goes through many processes. The most common operations are image acquisition, preprocessing, braille dot detection, braille cell formation and interpretation. These main processes are discussed here below in light of what researchers have done and recommended on the area in recent years.

2.5.1 Image Acquisition

The first operation in any pattern recognition system is the acquisition of pattern data where in the case of OBR would be the acquisition of braille image. For any optical recognition process the quality of image captured has a huge impact on the recognition result. Therefore, care should be taken while selecting the type of digitizing equipment to be used and on parameter selection. The acquisition of images can be performed with the help of Scanner or digital camera [13] [30]. Flat bed type of scanner is preferred by most researchers [3] [6] [22] [30] [44] with an image resolution between 80 and 300 dpi [7] [11] [30] [37] [44]. The resolution should not be lower so that important information about the dots will not be missed and it should not be high so that unimportant details will not manifest.

2.5.2 Preprocessing

Preprocessing operations enable removing of noise and representing the scanned image with only pixels which are relevant for the operation. It is mostly affects the classification approach to be followed. Since different researchers follow different classification approach, the type of preprocessing operation performed and the algorithms used by one researcher vary from another. But, at this stage, such operations like conversion of full color image into gray, detection and removal of noise, edge enhancement, morphological operation, and skewed angle correction are commonly performed [31].

Computers store scanned full color image as 3D arrays. But, performing operations on such images is very difficult and resource consuming. Therefore, the scanned braille image is converted to gray scale image that represents the image in a 2D array [39]. On the gray scale image [13] [44] applied a low pass Gaussian filter to reduce impulse noise and enhance the edge, while [6] used an average filter followed by contrast enhancement to reduce the effect of random noise.

Morphological operation enables enhancing of dots edges while skew angle correction detects the angle in a wrongly scanned Braille image and corrects its degree. Skew correction was not given attention by many researchers but there were some efforts by researchers in [3] [1] [32] [35] recently. In [3] tried to perform a horizontal projection followed by number of row count after rotating the image one time to left and one time to right. If the row count for the left and right rotated image is the same the image is not skewed, else image is skewed and iterative processing is done until the deviation between the left and right is more than 1/16.

2.5.3 Braille Dot Detection

Braille dot detection is the extraction of dot regions from the braille image. This is usually undertaken by extracting the shadows created from the dots while scanning. In doing this, the image is reduced to binary values. There are different methods to binarize a digital image. One of these techniques, which is the simplest and most commonly used, is thresholding. Thresholding applies a value that can classify the pixels into different regions of intensity. Thresholding can be global or local. Further, the threshold value can be single or multiple. In using thresholding for image binarizaton, the main problem is finding an optimal threshold value [19] [26] [31].

Two types of thresholding, global and local, can be applied in braille image segmentation [8] [31]. Global thresholding calculates a threshold value from the total intensity level of the pixels that make up the image. Then every pixel in the image will be compared against this threshold value and its class value will be set. Local thresholding calculates threshold value after classifying the image into several areas. This will result in different threshold value for different regions of the image. Then every pixel will be compared against the locally computed threshold and its value will be decided [19] [26]. Local thresholding performs better in cleaning out noise and separating the area of dot from background [31] [34]. In [37] an innovative thresholding technique that uses an iterative algorithm that looks for pixel

value from an optimum area of Braille spots is applied. In [7] used a statistical method based on beta distribution of a gray scale image to find an optimum threshold values.

2.5.4 Braille Cell Formation

On the threshold image, algorithms are applied to extract areas of dots and group the dots into cell. The most commonly used algorithms used mesh grid formation and local measure. In [7] used grid formation by selecting a starting point and then generating a horizontal and vertical line on a regular interval. Separate grids are formed for the recto and verso dots by checking the color distribution. This technique can help in removing of wrong dots and recovery of possible dots. But, it has problem likes difficulty of identifying dots which can be connected. In addition the whole performance could fell if the scanned Braille document is not correctly angled [3] [7] [32]. [3] Performed a horizontal pixel search for part of dot using 8 pixels height for dot. The 8 pixels are composed of a dark and a bright region with a space between them. Pixel values in the image with such property will be identified as dot parts and based on the color distribution a global measure will identify the image into columns of recto, verso or undefined column. But, [16] used a little different approach, that use a predefined cell frame size of 95X80 that can move over the image to check for Braille cell and when found the dot size of 20X15 pixels will be used to check existence of dot on the possible regions inside the frame. However, this technique may not be good for double sided image. Therefore, [25] applied a grid inside the frame that classify the frame into three and check existence of recto or verso dots using the Braille distance rule.

2.5.5 Interpretation

After all dots are detected and grouped into cell, a binary code is created for every cell based on the location of dots in the cell [25]. In the binary code a value of 1 is attached where dot exists and 0 otherwise. These binary codes can be converted into decimal values and interpretation to ordinary text can be performed by comparing the values with a stored dictionary or lookup table [3].

2.6. Review of Braille Recognition Works

2.6.1 Review of Amharic Braille Recognition works

The attempt for applying OCR technology for Amharic Braille documents was started recently. The recent research works on the area that attempt to find good techniques for

recognition of Amharic Braille Characters includes the works of Teshome [43], Ebrahim [18], Shumet [41] Miftah[32] and Berihu[9].

To the best of the researchers knowledge Teshomes' [43] work was the first attempt in developing Amharic Braille Recognizer. To acquire the image of the Braille document he has used a regular flatbed scanner with 200 dpi. Braille images were binarized using global tresholding technique which has shown better computational performance than local tresholding. To segment the dots from the image, Teshome [43] used mesh-grid construction and adopted a modified region based approach to extract features. The features extracted are used to build a model, using the MATLAB feed forward Artificial Neural Network (ANN), which is used for Braille character recognition. The experiment has shown that the designed system can perform with 92.5% accuracy.

Teshome [43] performed his experiment on clean Braille document. However, in real life application of Amharic Braille recognition, it is essential that the system recognize a noisy Braille. This is because Braille document by their very nature are damaged due to repeated use, bend and scratch. This will create noise on the image of the Braille. This noises and impurities highly affect the recognition result. Ibrahim [18] aimed to explore different noise detection and removal algorithms in order to enhance the performance of the Amharic Braille recognizer in real life Braille document images. With this regard, Ibrahim [18] examined and identified attributes and features of Amharic Braille characters and selected the best noise detection and removal algorithms. After this, a program is developed for the noise removal algorithm and is integrated into the Amharic Braille recognizer.

The system was implemented using ANN for 238 Amharic characters, 10 numerals and 19 punctuation marks. Ibrahim [18] used Gaussian filtering with adaptive histogram equalization and morphological operation to detect and remove noise in real life Amharic Braille documents. The noise detection and removal and thresholding algorithms are integrated with the previous algorithms developed, by Teshome [43], for clean Amharic document and the system is able to achieve an accuracy of 95.5%, 95.5%, 90.5% and 65% for clean, small level noisy, medium level noisy and high level noisy documents respectively.

The effort made by Shumete [41] was to enhance the Amharic Braille Recognition system by focusing on feature extraction and classification. With this regard, three feature extraction algorithms: fixed cell measure, horizontal and vertical projection and grid construction were experimented. The test result shows that the fixed cell measure performs very well. To build

a classification model for prediction of Amharic characters from Braille cell, J48 decision tree and the support vector machine (SVM) classifiers were tested. Based on the experimental results, SVM outperforms decision tree classifier in predicting unseen extracted Braille features. After application of the explored feature extraction and classification techniques to real life Braille documents, an overall accuracy of 90.67% was achieved.

Miftah [32] presented an approach for recognition of Amharic Braille document which focused on three major areas; noise detection and removal, Braille dot recognition and skew correction. The main focus of his work was the application of direction field tensor, that uses Gaussian filters and derivative of Gaussians for noise removal and isolation of Braille dots from their background. The other contribution of this work was skew correction which exploits the horizontal direction nature of Braille dots. Other contribution of this work includes the construction of Braille character lines to allow subsequent operations to be performed only on the character line. A half character detection method which helps differentiate dots from noises is applied. After detection of half characters, a horizontal distance analysis between the half characters is performed to find complete Braille cells. After formulating the Braille cells, a lookup table is used to translate them into corresponding Amharic characters. This system has achieved 99.9% and 96.5% accuracy for good quality and poor quality Braille documents respectively

As a continuation to previous works in the area of Amharic OBR the main focus of Berihu [9] research was to add Tigrigna Braille characters to the already designed Amharic OBR and to design a generic Amharic - Tigrigna Braille recognizer that can handle the various artifacts like dot size, merged dots and alignment of dots in Braille documents that includes designing of a noise tolerant segmentation technique and translation algorithm that helped to improve performance of the system in recognizing Amharic-Tigrigna real-life braille documents. The system was tested with ANN and SVM classifiers but it has achieved better result in using lookup table. The average accuracy level registered is 96.5%.

As we observe from the above research works, all are focused on single sided Amharic Braille recognition. But there are research works that attempted to recognize double sided Braille for English, Chinese, Arabic and Spanish.

2.6.2 Review of double-sided Braille recognition works

Regular feature extraction for recognition of Braille

In [13] presented an automatic system for recognition of both single-sided and double-sided Braille pages and converting to an editable English or Chinese text document. Before the system gets the Braille document from the scanner, consideration was made to the environmental factor in order to minimize the problem of image analysis. With this regard, the embossed Braille page is illuminated with a yellow polarized light source placed at an angle of 45 degree away from the page top and the dots of both sides are able to be captured clearly in the scanned image. The images are captured with a digital camera at a resolution of 512*512 pixels. On the captured image, two pre-processing operations are performed; noise filtering and edge enhancement. To filter the noise, a low-pass spatial Gaussian filter is applied that clean out the high spatial frequency introduced in capturing the image. But, in addition to filtering noise, the low pass spatial Gaussian filter has helped preservation of the detailed edge information of the Braille dots. The edge enhancement operation is performed to sharpen the fine details of the image that has been blurred. With this regard, two independent filtering operations using convolution Sobel kernels are applied.

For segmentation, the researchers considered a histogram made up of only those pixels that lie at or near the edges of the Braille dots. The pixels that lie on or near an edge are determined by using the Laplacian of Gaussian operator. The average gray-level value of these edge pixels are then used as the global threshold value. This has improved the segmentation result. Before feature extraction the boundary points of Braille dots are obtained using boundary based chain code algorithm which detect the boundary coordinates of the Braille dots. In case of double sided writing, the dots on both sides should be separated before any further processing. Using the boundary coordinates and illumination characteristics, two standard templates that represent front-faced and back-faced dots are constructed. Then the templates are applied to every pixel position of the image and the correlation at each pixel position is evaluated. Based on the correlation value obtained, the separation of the Braille dots is performed. In order to group the dots into cells, the distance between the centroid and the four possible neighbors are determined. Then within the cells, the dots positions is determined and represented as a bit string. Finally, the bit strings of the cells are searched against a Braille dictionary and retrieved characters are grouped into words.

This work had shown that recognition process starts before image acquisition and application of a yellow polarized light source from an angle of 45 degree has trivialized the image processing work. The edge enhancement technique applied has improved the image very well and the segmentation technique applied was highly effective. In addition, the application of a post recognition activity has helped to improve the recognition output. As a result, the system has achieved a recognition accuracy of 100% for single-sided Braille and 97% for the double sided Braille.

An Application of Eight Connectivity based Two-pass Connected-Component Labeling Algorithm for Double Sided Braille Dot Recognition

The objective of the system proposed in [48] is to develop an optical Braille character recognition system that takes into consideration differences in scanned image quality and resolution. First 24bit images of Braille are captured with Hp Scan jet 3400C A4 size scanner with an image resolution of 200dpi and spatial resolution of 1501 x 2121. The algorithm begins by converting the captured colour image to gray scale image. In the scanned image, it is observed that the shadow produced by the depressions relatively account for a lesser number of pixel count when compared to those produced by the protrusions. This is due to the reflection property of the light. Therefore, it is believed that retaining only the brighter areas can allow recognition of both sides of dots at once.

In order to remove any inherent noise present in the image, thresholding is performed on the gray scale image. A single thresholding is applied where pixel values less than the set threshold are replaced by zero and values greater than the threshold are retained as they are. But, in the threshold image an impulse noise is observed. To remove the impulse noise a median filtering is applied. Further morphological dilation is performed in order to increase the area of the dot. The morphological dilation is more helpful in increasing the area of the verso dot rather than recto dot.

The main purpose of this project is separation of verso and recto dot. With this regard, the eight connectivity property of the pixel relationship is employed. Eight-connectivity for any pixel means that it is connected horizontally, vertically and diagonally. When applying the eight connectivity pixel operation, the count of the recto is found to be less compared to the verso. Therefore, differentiation between the recto and verso dot is done by applying thresholding operation on the basis of eight connectivity based pixel count value. Those dots

with eight connectivity count greater than the threshold are considered as the verso dots and those dots for which the eight connectivity count value is less than the threshold are considered as recto dots. The separated dots are then placed into separate document in a location similar with the original document. After separating each side, the Braille dots are transcribed into their corresponding natural text. In order to do the transcription it is required to group the dots into cells first. Usually grouping dots is made by constructing grid over the dots with support of the standard Braille dimension. This technique is ineffective and time consuming. Therefore, this paper proposed the use of adaptive grid construction technique which is found to reduce computation time and is also effective in avoiding any misclassification of dots. This is possible because the adaptive grid construction technique applies an algorithm that considers three factors of Braille image; distance calculation, horizontal projection profile and vertical projection profile.

During the dot recognition process all the valid Braille dots have been detected on either sides of the document and the two images are formed for each side of the document. To convert the recognized dots into their corresponding natural characters, the scanning pattern is used for dividing each cell into grids consisting of six parts and corresponding code for each cell is generated according to the presence or absence of a dot in each grid. The presence or absence of all the valid Braille dots is found by multiplying the grid constructed image with the dot extracted image. If the product is true then it indicates the presence of dot and its value is indicated with 1 and if the product is false then it indicates the absence of dot and its value will be 0. The dot positions are determined with number 1 to 6 as per the universal numbering of Braille dots. Within each cell, the dot pattern is determined and is also represented by a bit string which is later converted into equivalent decimal codes. To retrieve the natural characters corresponding to the Braille characters, a matching algorithm is applied in which, decimal code generated from the processed image are searched against corresponding Unicode characters stored in the lookup table. Then the MATLAB function, ***unicode2native***, is used to convert the Unicode character into corresponding natural text.

This paper presented a novel technique that make use of an eight Connectivity based two-pass Connected-Component Labeling algorithm and thresholding technique with an adaptive grid construction to separate and recognize double sided Braille. The proposed system was able to separate and as well recognize double sided Braille document with a minor error. The errors happened due to failure to handle all noises in Braille image. The proposed system also failed to handle deformation, skew and merging of verso and recto dots.

Braille character Recognition using Generalized Feature Vector Approach

The main objective of [25] is the application of a multi-parameter based feature detection algorithm along with a sliding window and a grid to efficiently detect the embossed dots from a Braille print. Based on two steps model selection process, model is selected from the segmented image and parameters are calculated. Second, on the image segmented with colour parameter further calculation is made for Generalized Feature Vector.

On the acquired image a portion is select and on the segmented image a model is selected and parameters are calculated. To detect the dots, a sliding window with a fixed interval is used. The size of the sliding window is decided considering dot dimension measure of print Braille and the standard inter-dot and inter-cell spacing. The dot position for each cell is then detected using GFV. GFV is a multimodal probability distribution in a multidimensional feature space which essentially encapsulates multiple features for consistent detection. It enables the use of multiple features to reduce error and false alarms or wrong identification.

Each cell is divided into grids and the presence and absence of a dot is sensed and corresponding code for every cell will be generated. In the case of double sided Braille, the algorithm is further modified to enable it distinguishes the front side dots alone from the Braille image. The process is similar, using a sliding window, but uses a grid method to distinguish between the front and back side. The sliding window will be sub-divided into three vertical regions. The front sided dots will be extracted based on their location in the grids. For example, a dot fully appearing in region 1 and 2 is a front side dot. A dot appearing partially in any of the regions is backside dot or a false dot created when dots from the two sides are merged. In addition to the sliding window and the grid, here the GFV parameter which consist size ratio analysis along with color and energy matching will be applied to differentiate between a single dot and a merged dot. Finally, after dots are extracted, a code based on the position of dots in a cell will be generated. The generated binary code will be matching up with a lookup table.

The unique operations of the proposed system are the selection of sliding window dimension and interval which will enable extraction of cell position before dots that make up the cell. The other novel technique is the use of generalized feature vector for detection of dots. The researchers claim that the GFV approach can efficiently detect Braille dots even from low quality image. Though, the accuracy of the proposed system is not quantified, it is shown that

the system can successfully distinguish between front and back side dots. One of the weaknesses of this system is that, simultaneous recognition is not possible.

The present work made an effort to recognize both single-sided and double-sided Amharic braille documents. In the current work, an effort is made to improve the image acquisition process by considering the environmental factors that affect the scan quality in order to minimize the problem of image analysis. This work has introduced a combined technique that uses both projection profile and grid construction to segment and extract features of braille dots using only those dot regions which are perfectly constructed. It also uses a grid improving technique that is found to reduce errors to minimum. As a result, it can be said that the current work highly reduces the process of image analysis but with improved result.

CHAPTER THREE

AMHARIC BRAILLE RECOGNITION

Vision is the one sense that we come to depend upon above all others and indeed the one that provides most of the data that we receive from our environment. The eye which is the input pathway of visual information from the outside world into the brain provides two times greater information to the brain than all other senses collectively. Though the process between the eye and the brain is very complex, the visual perception and interpretation activity is normally carried out in tenth of a second. In the past few decades people have been trying to get machines do much of their works. Machines have been doing simple mechanistic tasks with no difficulty. But, for more complex tasks such as those that require sense of vision, the thing that is trivial for humans has been very challenging for machine to perform and computationally demanding. This fact is evident in character recognition systems [5] [19].

Character recognition systems are used to convert characters in hard coded documents into computer readable and editable text. One of such system is Braille Character Recognition which is a system that reads braille documents and converts the braille characters into their equivalent textual representations. This recognition process is dependent on image processing, artificial intelligence and computer vision capacity [29].

Braille recognition can be either single-sided or double-sided based on the way the braille document is prepared. Unlike human recognition that recognizes objects as they are, optical recognition systems work by deducting every object separately from the background [19]. This task makes recognition in general and that of braille very difficult. Specially, with double-sided documents where the dots of both sides are visible from one side, the dots will be highly connected to each other and separation of a single dot is a challenging task [26].

The whole braille recognition task is divided into five main operations; image acquisition, preprocessing, segmentation, feature extraction and translation. The recognition process starts by acquiring image of the braille document through a scanner, digital camera or any other digitization means. The digitized images are preprocessed to filter out noise and unnecessary information which are not relevant for the recognition process. Various techniques are applied to segment braille dots from the preprocessed image and form a braille cell. Finally,

features of the segmented objects are used to translate them into the desired language textual representation.

This research is concerned with the application of different braille recognition techniques for conversion of double-sided Amharic braille documents into Amharic print characters. For this purpose, various approaches used in braille recognition such as image digitization, thresholding, segmentation, feature extraction and recognition are reviewed.

3.1. General Design of Amharic Braille Recognition

The overall design of the Amharic Braille Recognition system is shown in figure 3.1. The main focuses of this work are highlighted with double rectangles.

The system is able to recognize both single and double sided Amharic braille documents. To translate Braille code characters into their equivalent textual representation a series of steps are required. In every character recognition system including braille recognition systems, data is feed to the system in a form of an image. Therefore, sample Amharic braille documents are collected and an image of the braille document is acquired using a flatbed scanner with 200 dpi. The acquired braille color images are then converted to gray scale. On the gray scale image preprocessing operations such as noise filtering, intensity adjustment, size reduction, thresholding and morphological adjustments are performed. Before binarizing the image for segmentation, separation of the recto and verso dots is performed using a moving window filter. The front and back side dots are separated and put into two binary images. The binarized braille images are feed to an improved segmentation and feature extraction techniques. Finally, a lookup table is developed based on which the extracted features are parsed to their equivalent Amharic textual character representation.

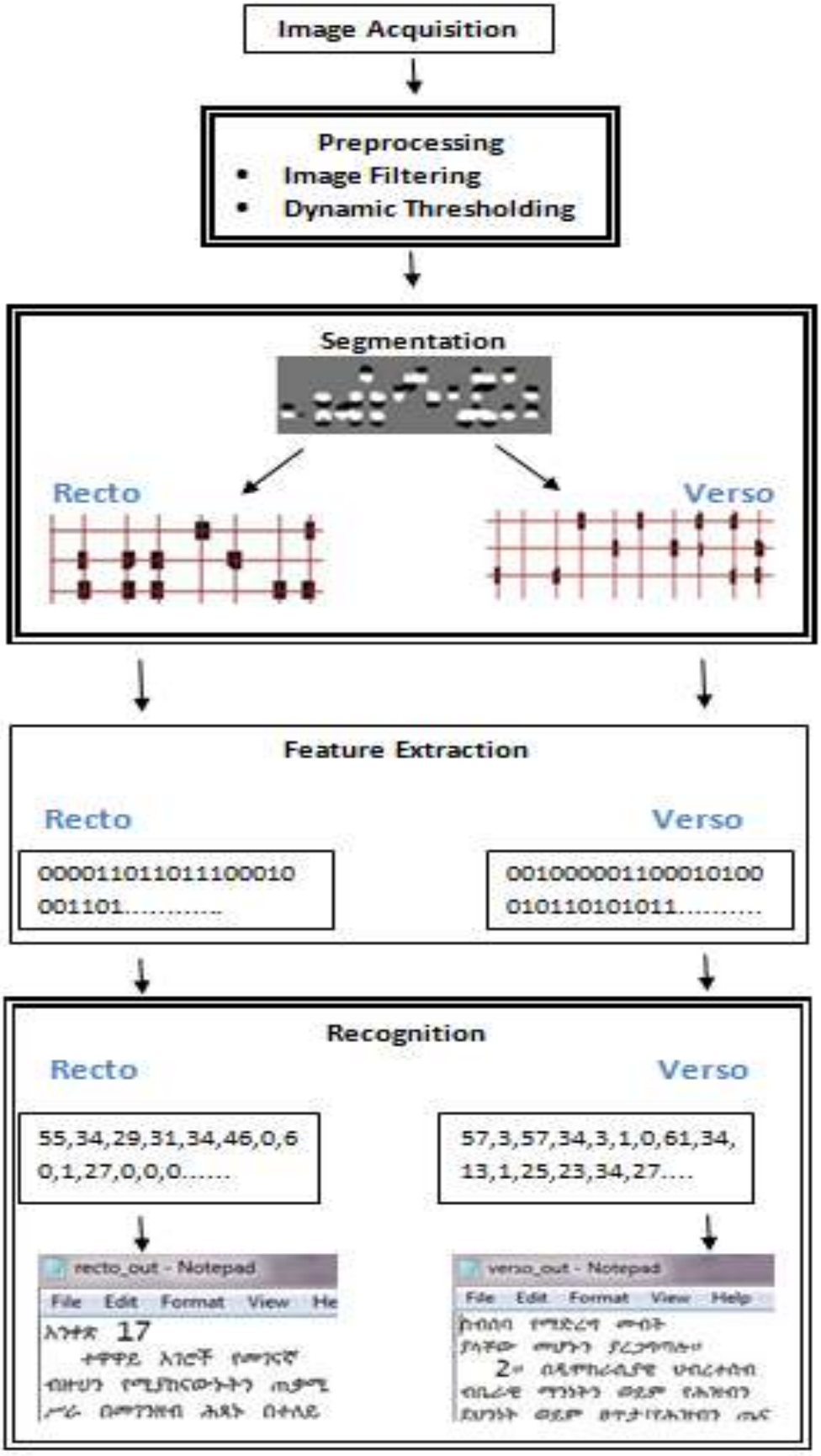


Figure 3.1 Design of Amharic Braille Recognition System

3.2. Image Acquisition/Digitization

Any character recognition system starts by acquiring the document in a form of an image. The acquisition process can be done with the means of digital camera or scanner. A flat bed type of scanner is the most preferred technology by many researches. This is because flat bed scanners are cheap alternatives that can be operated easily and produce the required image with minimal time. In OBR also, scanners have been in use dominantly.

To be able to distinguish the dots from the background, the recommended spatial resolution setting is between 80 and 200 dots per inch (dpi) [7],[16], [26],[30]. Other things to consider while scanning a braille document is that the color of the document to be scanned can affect the output. This happens due to the effect of the reflected light from the scanner during the process. We have observed that such effects can be handled by setting scanner to apply adaptive light mode, rather than the default settings that apply constant lighting for any colored document type. But, this option may not be available on every scanner. Another option can be scanning in gray scale or reducing contrast and brightness during scanning. In this work an adaptive lighting is used to get a quality color image of the braille documents.

3.3. Preprocessing of double-sided Braille Images

In OCR, image preprocessing is an important step which includes the activities of converting RGB image into gray, filtering image to reduce noise, skew angle correction and many others. After pre-processing the image, only important regions of the image will be extracted. This process is called Thresholding [19] [38]. Thresholding, in braille recognition, enables the extraction of Braille dot regions from the background from a filtered gray scale braille image. In double-sided braille recognition, only three regions can be useful; light area, shadow (dark) area and background area. Therefore, it is enough to represent the images into three color levels, White, black and average gray, rather than full 255 gray levels. The dot areas appear as regions with combination of black and white, while the background will hold an average gray value of the image.

There are different threshold values setting techniques applied by different scholars for braille recognition. In this work, the thresholding technique used by [3] is adopted with some modification. The thresholding technique is detailed in algorithm 3.1 below.

1. *Calculate the average gray level for the whole image*
   Let image be an array representing the whole image
   Let x and y be the horizontal and vertical coordinates
   set x=y=0; // set coordinates to first pixels
   for x=0 to n rows in image do
   for y=0 to m columns in image
   sum+=image(x,y)
   while end of image
   *avg=sum/(n*m)*
2. *calculate values for a and b*
   Let max(image) be highest value in each column
   Let min(image) be lowest value in each column
   a=mean (max(image)+ avg)/2;
   b=mean (min(image)+ avg)/2
3. *add values higher than a to array y and values less than b to array x.*
4. *Calculate the max_avg and the min_avg*
   max_avg=mean(y);
   min_avg=mean(x)
5. *Compute the classification parameters, h and l*
   h = avg + (max_avg – avg)/3;
   l = avg - (avg – min_avg)/3
6. *Set Pixel values higher than h to 255 (white)*
   pixel>h=255
7. *Set pixel values less than l to 0 (black)*
   pixel<l=0 and
8. *Set pixel values between h and l to mean(average)*
   h>avg>l

Algorithm 3.1 Thresholding algorithm [3]

This thresholding technique calculates two values, higher and lower bound, from the image that classifies the image into three regions. Any pixel that lie below the lower threshold value is set to 0 (black), any pixel that lay above the higher threshold value is 255 (white) and any pixel in between the two threshold values is considered as background and is set to average pixel value of the image.

3.4. Segmentation

The thresholding operation leaves us with an image having three intensity levels only; black, white and an average gray color. The black and white regions represent the foreground (dot regions) while the average gray region is the background. This image can hold both recto and

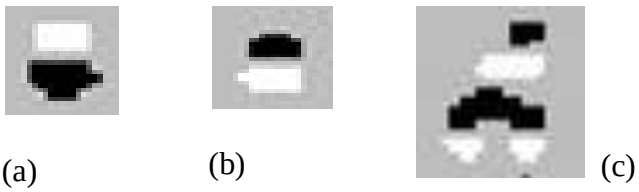
verso dots. Therefore, before any other operation is performed, the regions of recto and verso dots should be identified. After the regions of the dots are extracted, we separate the recto and verso regions into different array. For each image, image holding recto and image holding verso, we can extract the dots from the region identified and later group them to form a cell.

3.4.1. Braille Dot Detection

In the preprocessed image, a single dot appear as two separate regions and in the case of double-sided writing, the dots from both sides are highly connected. This makes detection of a complete dot at once difficult. As discussed in [3], detection of dot parts rather than detection of whole part produces better result. As per observation from the scanned image, the size of the dots from the back side is smaller as compared to those from the front. However, from experiment, the average dot height is found to be 8 pixels for recto and 7 pixels for verso. This information is used in the algorithm that separates the dot region from the background.

3.4.2. Separation of recto and verso

The recto and verso dot separation is performed with the help of the black and white region orientation. The black and white regions are formed as a result of the shadow created from the scanner in the process of acquiring the braille image. As discussed above, in the threshold image, the recto dots appear as white region from above and black region from below, while the verso dots appear as black from above and white from below. Sample, recto and verso dots can be seen in figure 3.1 below. In a double sided braille image, the dots are highly connected. The connection, most of the time, happens when similar region from recto and verso are connected. This way two or more dots can be connected together which makes the separation of a single dot from the image very cumbersome.



(a) (b) (c)

Figure 3.1 Images of recto and verso dots a) recto dot, b) Verso dot and c) Recto Verso dots connected

Initially, an effort is made to separate connected dots using morphological operations. But, in the resulting image, the shape of the dot was highly affected and it was not successful in separating all connected dots. As a result recognition performance was very low. In this

work, we have used a moving window filtering technique which is adopted from [3] and modified to enable recognition of frontal side and backside dots without the need to separate connected dots. The technique is explained in algorithm 3.2 below. The algorithm is designed in such a way that it does not scan the same area twice and it skips 8 pixels when a recto or verso region is found. This has reduced computational time and number of false recognitions.

At the end, we have two arrays holding the dot regions for the recto and verso dots separately.

1. *Create a matrix w of size 8x1 with the first four of its value 255 and the last four 0*
2. *Create two empty arrays that will hold the recto and verso dots separately*
3. *Start from the first pixel and perform a vertical scan with a window size as w*
4. *Compare the selected window against criteria*

   Criteria:-

   a. *first two or three pixels are bright(255) while last two are dark(0)*

   b. *first two pixels are dark(0) while last three are bright(255)*
5. *While criteria 'a' is true*

   - multiply the selected window with w to get the bright region only

   - register the result to recto array

   - skip 8 pixels vertically

   -go back to step 4 until end of image
6. *While criteria 'b' is true*

   - multiply the selected window with w but this time w fliped vertically to get the bright region only

   - register the result to verso array

   - skip 8 pixels vertically

   -go back to step 4 until end of image

   Note:- the value 8 which is used as a window size to filter the image is selected with experiment to be the average dot height.

Algorithm 3.2 Front and back side dots separation algorithm [3]

A single dot, on average, appears with 3 pixels height and 3 pixels width. But sometimes, due to the quality of the scanned braille image, a dot region may not be detected at all or may be detected with more than one separate region. In order to recover such improperly detected regions, a neighborhood analysis is performed that will look if a pixel's horizontal neighbors

are a dot region. If the case is true, that pixel will also be considered as a dot region. This technique can also help to remove most of the wrongly detected regions; since those regions mostly appear as an isolated one column region, the neighborhood pixels constitute a none dot region. This and other morphological operations had improved the shape and size of the dots detected.

3.4.3. Braille Cell Grouping

After the dots are detected, they are grouped into cell that makes a single character or part of a character. Having identified all possible dot regions and separated the image into recto and verso, we have two separate binary images to be processed; image holding the recto dots only and image holding the verso dots only. However, the verso image is flipped horizontally to bring it to its normal reading position. As discussed in [41], there are different techniques for grouping braille dots into cell such as fixed cell measure, horizontal and vertical projection and grid construction. All this techniques try to make use of the fixed matrix arrangement nature of braille dots.

In this work, an improved flexible grid construction technique that uses horizontal and vertical projection results to find the coordinate value for the grid is applied. First, for each separate image, we find the horizontal and vertical projection profile. But, building a horizontal and vertical projection on all dots in the image will produce incorrect result. This mostly happens due to two reasons:

1. Over projection:-
   When dots which are smaller in size are separated in distance from other dots in the line, they create their own projection. This will create horizontal or vertical projection beyond the number expected.
2. Under projection:-
   When dots are bigger in size they will fill up the vertical and/or the horizontal space that should exist between dots. In this case, the horizontal or vertical projection will be less than the number expected.

Therefore, to alleviate these problems, a dynamic algorithm is applied that extracts properly sized dots only and build the horizontal and vertical projection for the whole image from those selected dots only. The grid is constructed from the horizontal and vertical projection profile. The coordinate values to construct the grids are calculated from the centeroid values

of each connected component in the horizontal and vertical projection array. As discussed above, the size of braille dots is not uniform. Horizontal and vertical projection works on an area that encloses every dots region such that no dot exists outside this region. This is computationally ineffective and resource consuming. But, using the centeroid values from the projection profile enables us to get a line that crosses most of the dots in that region. The technique is explained in detail in the algorithm 3.3 below.

1. *Calculate the area of every object in the image using*
   Let n be total number of objects in image
   For i=0 to n do
   Calculate area
2. *Calculate the average area of the dots*
   -Let m be matrix holding area of every object
   Average_area=average(m)
3. *filter object with area between average_area - 2 and average_area +2*
   let s be objects filtered
   (average_area -2)>s> (average_area +2)
4. *Create a vertical projection array which is the size of the new image height and the sum of connected pixels in every row*
5. *Create a horizontal projection array which is the size of the new image width and the sum of pixels in every column*
6. *Compute centroids of every connected component in the projection*
7. *Connect centroid point from vertical projection to connect pixel in y axis*
8. *Connect centroid point of horizontal projection to connect pixel in x axis*

Algorithm 3.3 Grid construction algorithm

When building grid on a perfectly extracted braille dot region with the centroid values of the horizontal and vertical projection results, the line will perfectly fall over the center point of the dots. The vertical lines in the grid appear as two parallel lines while the horizontal line appears as three parallel lines.

On a well constructed grid image, any possible cell region can be reached. This is possible because, when three horizontal parallel lines are crossed by two vertical parallel lines they create six intersection points. The braille cells are extracted by simply looking for any six intersections with smallest distance between them. The six intersection points will be the

position of the six dots in a braille cell. But the above algorithm finds grid only where there is at least one perfectly sized dot in the region of the image. In order to have a complete picture of the cell, it is needed to have grids on possible dot areas as well. This problem is solved using an algorithm that refines the grid by checking if a gridline exists in all potential dot areas and draw a grid where not exists. From the constructed grid, the algorithm finds two and three consecutive and perfectly constructed vertical and horizontal line that represents the two columns and three rows in a cell respectively. The space between these lines is used to estimate the other lines that should exist in the image. This technique is detailed in algorithm 3.4.

1. *Refine horizontal grid line*
    a. *Loop through all line and find three consecutive lines having similar space between them.*
    b. *Calculate GH using estimated vertical space between dots (vsb) and estimated vertical space between cells(vsd)*
    GH=vsd + vsc
    c. *Take the coordinate value of those selected lines*
    d. *Starting from the selected lines move iteratively through the image, both towards top and down, in step of GH to check if a grid exists on or near the point. The checkpoints are the potential dot areas.*
    e. *Add new horizontal gridline where none is identified on the check point.*
2. *Refine vertical grid*
    a. *Loop through all line and find two consecutive lines having average horizontal dot space between them.*
    b. *Calculate GV using estimated horizontal space between dots (hsb) and estimated horizontal space between cells (hsc)*
    GV=hsd + hsc-
    c. *Take the coordinate value of those selected lines*
    d. *Starting from the selected lines move iteratively through the image, both towards left and right, in step of GV to check if a grid exists on or near the check point. The checkpoints are the potential dot areas.*
    e. *Add new vertical gridline where none is identified on the check point.*

Algorithm 3.4 Grid refining algorithm

3.5. Feature extraction

In this work, to extract features of a given cell, the existence of dots on or near the intersection points of the grid is checked. First, an array is built, to hold the feature of the image. By starting from the left top corner, the array is populated with either zero or one based on the existence of dot on the intersection points. One implies there is a dot on the intersection point, while zero implies there isn't. The scan is made in such a way that it will scan one cell first in the order from the first dot to the sixth and move on to the next until the end of the grid. To do this, the algorithm start from the first intersection point in the grid, move on horizontally to the next intersection point in the same vertical line and next to the third intersection which is again on the next horizontal line on the same vertical point. This makes half a cell. To complete the other half, we move to the next vertical point on the same horizontal line as the first intersection and do the scan the same way we did for the first half. The method for extracting the features is detailed in algorithm 3.5 below.

1. *Create an empty array with the size of the intersection points in the grid*
2. *Take the first three lines in the horizontal grid which represent the three lines in a single cell,*
    a. *Start on the first intersection point and check if there exists dot*
    b. *Move to second line of the first three and check the same*
    c. *Move to the third line of the first three and check the same*
    d. *Move back to the first line and perform a check on the second intersection*
    e. *Jump to the second line and perform a check on the second intersection*
    f. *jump to the third line and perform a check on the second intersection*

   This make up six checkpoints which represent the six dot positions in a cell.
3. *For every checkpoint, if a check results true put 1 on the array else put o.*
4. *Scan the remaining intersection points on the selected horizontal lines of the grid the same way*
5. *Move to the next three lines in the grid perform step 2-4 until end of grid*

Algorithm 3.5 feature extraction algorithm

At the end of this process, we have the feature of the cell in an array of zeros and ones. The size of the array is six times the possible cell positions identified by the grid. Braille cells are represented with six binary values that represent the existence of dot on six possible dot positions. To make up a binary code value for each cell, we look for any six consecutive binary values starting from the first point and moving in a step of six in the array holding the features. After we get the binary value of each cell, it is converted in to decimal code representation for simplicity of subsequent operation which is translation [3]. As shown in formula 3.1, the decimal code is the sum of the multiple of the six binary values with six decimal points; 1, 2, 4, 8, 16 and 32 in the order from the first dot to the sixth. The six decimal points imply the coefficients for converting a binary (base 2) number in to decimal (base 10) number.

$$\boldsymbol{number\text{-}2} = [\boldsymbol{d_N} \ \ldots \boldsymbol{d_2}\ \boldsymbol{d_1}\ \boldsymbol{d_0}\ ]_{\boldsymbol{n}} = \sum_{\boldsymbol{n=0}}^{\boldsymbol{N}} \boldsymbol{d_n}\ \boldsymbol{b^n} = \boldsymbol{d_0}\ \boldsymbol{b^0} + \boldsymbol{d_1 b^1} + \boldsymbol{d_2}\ \boldsymbol{b^2} + \cdots + \boldsymbol{d_N}\ \boldsymbol{b^N}$$

Where b - Numeral System base

d_n - the nth digit

n - can start from negative number is number contains fraction

N+1- Number if digits

Equation 3.1 binary to decimal conversion rule

For example a braille cell that have dot on the 1st, 2nd and 5th position will have a binary code of 1,1,0,0,1,0 and its decimal value can be calculated as:

$$\text{Decimal code (base 10)} = \mathbf{1}*2^0 + \mathbf{1}*2^1 + \mathbf{0}*2^2 + \mathbf{0}*2^3 + \mathbf{1}*2^4 + \mathbf{0}*2^5$$

$$\text{Decimal code} = \mathbf{1}*1 + \mathbf{1}*2 + \mathbf{0}*4 + \mathbf{0}*8 + \mathbf{1}*16 + \mathbf{0}*32 = 19$$

3.6. Translation

The translation of braille features into their equivalent textual representation is achieved by building a simple lookup table that holds a dictionary of features and their equivalent textual representation. The translation of English braille, that holds few numbers of characters, is a simple one-to-one-matching. But, Amharic has a large number of characters. Therefore, a single textual character can be represented with either one or more braille cell and hence, one or more features. This fact produces a huge challenge in building a lookup table for Amharic.

For this particular work, we have built a lookup table for 281 characters, 22 punctuation marks and 20 numerals.

First, the extracted features are read and saved to a text file and an algorithm is designed that matches every value in the text file against its corresponding Amharic textual representation in the lookup table.

CHAPTER FOUR

EXPERIMENTATION

Braille is the preferred means of writing for visually impaired people that allows them the ability to share their ideas to the world and also provide them access to what the world has in braille written form. Digitizing braille can facilitate two way communications between vision and visually impaired society. It can also enable to preserve literatures found in braille form through electronic means which is a space saving and better secured option. However, digitization of braille is not a simple process. Several countries have attempted to develop a braille recognition system for their own languages. In Ethiopia, there were efforts made by Teshome[43], Ibrahime[18], Miftah[32], Shumet[41] and Berihu[9] to develop braille recognizer for local languages such as Amharic and Tigrigna. As a continuation to this efforts which focused on single sided braille only, this study explored techniques that can allow recognition of both single-sided and double-sided braille documents at once.

First, both single and double sided braille documents are collected and scanned using flatbed scanner at a resolution of 200 dpi by applying the appropriate setting for light and contrast. The digital image is preprocessed to reduce the noise and scale the image into relevant regions only. Based on the orientation of colors (Black and White) in the dots the separation of dots into recto and verso is performed. The important regions of the dots are segmented and based on the segmentation result the features are extracted and passed to the translator that matches features to their equivalent visual Amharic characters in the lookup table. This chapter provides a detailed performance explanation of the main activities of the Amharic Braille Recognition developed in this work.

4.1. Data collection

The data set used in this work is prepared from braille documents collected from Meserach Center. Braille documents can be prepared with any color. The color of document used in this work is mainly white and light yellow. This is because they are the most widely used and available ones. The braille documents are standard 11 inches by 11.5 inches size. Braille can be written single side or double-side. The focus of this work is the recognition of double-sided Amharic braille document. But, the system is designed to work for single-sided braille recognition as well. Therefore, the documents collected include both double-sided and single-sided braille which are made of paper sheet embossed with the help of an embosser.

The detail of the sample braille documents collected for the purpose of experimentation is given in table 4.1.

Dataset property	Description
Number of Braille sheets single-sided, double-sided	 6 30
Braille Type	Grade I
Total number of characters	23,142
Average number of characters per sheet single-sided, double-sided	 387 694
Color type	White, Yellow
Resolution (horizontal/vertical)	200 dpi
Image size	12.5mb
Digital Format	BMP
Document size	11" x 11.5"

Table 4.1 Database created

4.2. Digitization / Image Acquisition

The input to any braille recognition system is a digital image. Digital braille image can be achieved by scanning braille documents with a scanner or by simply taking a picture with a digital camera. Both techniques can provide the expected result. But, using digital camera requires adjusting light source and angle of camera which is most of the time imperfect and creates a skewed kind of image with non-uniform light intensity and shadow size. This requires additional preprocessing to improve the image. Therefore, in this work, we have

preferred to use scanner which is a cheap alternative that can produce an excellent digital image with minimal effort.

The type of scanner used in this project is an HP flatbed scanner that produces digital image of the braille document into windows bitmap (BMP) format. The scanning is performed with 200 dots per inch (DPI) resolution which is the recommended level to get a good quality braille image for subsequent processes. Braille can be written with any colored paper sheet. The most dominantly used ones are gray, yellow, green and white. When scanning braille sheets with lighter colors, such as white and light yellow, the light released from the machine on the document during the scanning will affect the bright region of the dots making only the shadow region to appear in scanned image. These images could not enable identification of recto and verso dots. To solve this problem, we experiment other options such as scanning in gray scale rather than full color, reducing contrast and applying adaptive lighting system.

From experiment, we have identified that the use of adaptive lighting system enable producing a better quality digital image for any colored braille document. The kinds of braille sheet used in this experiment are white, yellow and gray. Sample digitized braille is presented in figure 4.1.

(a) (b) (c)

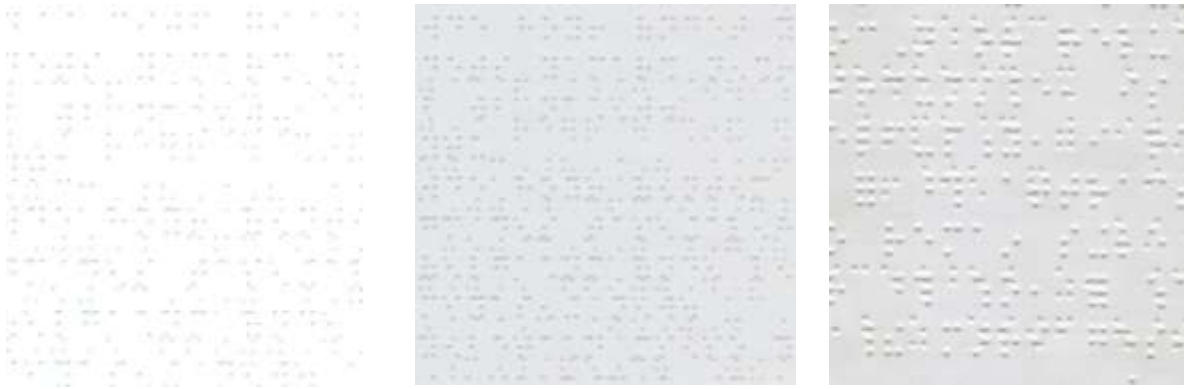


Figure 4.1 Scanned braille image; (a) ordinary scanning (b) gray scale scanning (c) scanning with adaptive lighting

Since, adaptive lighting system is used in this work, the scanned braille images are full color (RGB) images. Therefore, they must be converted to gray scale image before any further processing. To do this, the color space value in the size of the scanned image is check. If the color space value is three, it means the image is full color and should be converted to gray

scale. The MATLAB code for converting an RGB image to gray scale is shown in figure 4.2 below.

```
image=imread('D:\dataset\dset1\clean_01.bmp'); % get image
[m,n,z]=size(image);%Compute Size of Image, z is the color Space Value
if z==3
    img1=rgb2gray(img);
else
    img1=img;
end
```

Figure 4.2 MATLAB code for converting full color image into gray scale

4.3. Thresholding

In OCR systems, binarization is the technique that enables to separate the foreground (characters) from the background. In this research, binarization is performed with a thresholding approach. There are different thresholding techniques such as, global and local. Global thresholding is used in this work with the reason being the appropriate approach for binarizing an image having uniform color intensity such as braille. In addition, global thresholding is computationally less expensive.

The main problem in using thresholding is the selection of threshold value to binarize the image. Threshold value can be set manually or can be calculated from the pixel value of the image using different technique. However, using a manually set threshold value will not separate the foreground (dot) and the background properly. Moreover, it is not possible to use a constant threshold value for all kinds of braille images since the scanned braille images differ in their intensity level.

In this work, an initial attempt was made to define a threshold value for the image using the matlab image processing toolbox function *multithresh*. The *multithresh(A, 2)* returns two threshold values computed from the input image 'A' by using Otsu's method. The two threshold values are used as an input argument for the function *imquantize*, that convert image 'A' into 3 (2+1) discrete levels. This approach identifies the dots from the background with some noise and it is observed that the dots are seen bigger in size and highly connected to each other. In addition, it is seen that for some images with unusual intensity level, this kind of thresholding doesn't extract the objects from the background clearly. Therefore, a

better thresholding technique that can handle the above problem is adopted from [3] and modified to improve result.

In Thresholding the image, the algorithm begins by calculating the average gray level of the whole image. After the average value is calculated, it defines the maximum and minimum values for each column of the image. Then from the maximum and minimum it computes the values for 'a' and 'b'; if the value of a given pixel of the image is greater than 'a', then the value is added to the variable 'y' and if the value of a given pixel is less than 'b', then the value is added to the variable 'x'. Finally, 'H' and 'L' are calculated based on max_avg and min_avg values. 'H' and 'L' are the thresholding values such that, if the value of a pixel is greater than 'H', then the pixel will be converted to white; or if the value of the pixel is less than 'L', it will be converted to black; otherwise it is converted to average value of the image.

The MATLAB implementation of the algorithm for thresholding is given in Figure 4.3.

```
%% Thresholding %img3 is filtered braille imahe
immean=mean(mean(img3)); % avg pixel value for the image
a=mean(max(img3)+immean)/2; b=mean(min(img3)+immean)/2;
[M N]=size(img3); %get size of the image
y=0; x=0;
for i=1:M
    for j=1:N
        if  img3(i,j)> a ;
            q=img3(i,j);
            y=[y,q];
        elseif img3(i,j)< b ;
            w=img3(i,j);
            x=[x,w];
        end
    end
end
max_avg=mean(y);
H=immean+(max_avg-immean)/3;
min_avg=mean(x);
L=immean - (immean - min_avg)/3; for i=1:M
    for j=1:N
        if img3(i,j)> (H)
            img3(i,j)=255;
        elseif img3(i,j) < (L)
            img3(i,j)=0;
        else
            img3(i,j)=immean;
        end
    end
end
```

Figure 4.3 MATLAB code for thresholding

The algorithm is implemented on matlab and has produced good result with any kind of braille image as compared to the built-in matlab algorithm.

The image in figure 4.4 presents a sample RGB image and its binarized representation.

(a) (b)

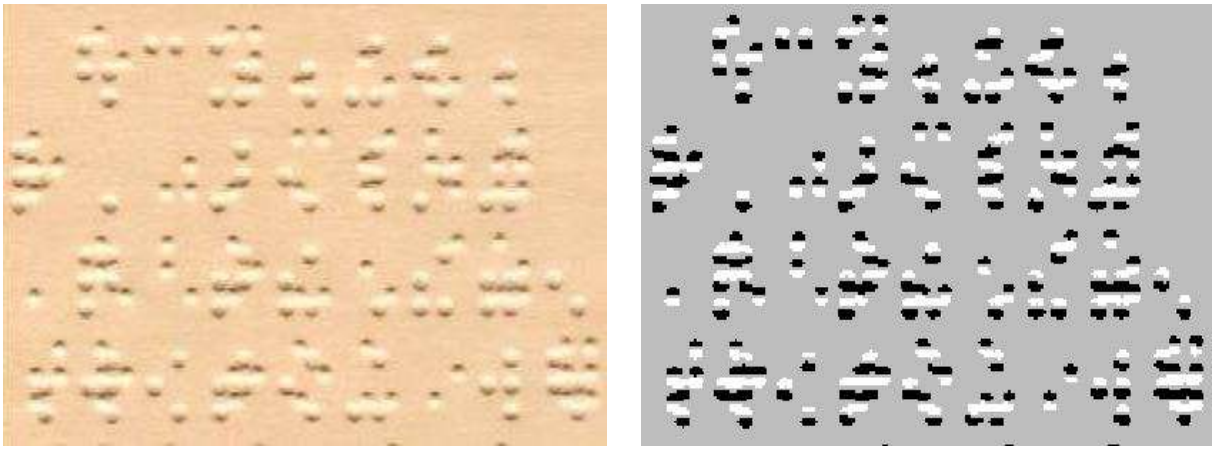


Figure 4.4 Sample thresholded image: (a) Scanned RGB Image (b) image a after thresholding

As mentioned above, braille document by their very nature can have their quality affected due to repeated use, scratch and bend. And in double sided writing the process of embossing itself will degrade the dots and the paper itself. These problems can affect the thresholding performance.

4.4. Separation of dots for Double-Sided Braille

The major challenge in recognition of double-sided braille is the separation of dots of recto and verso. The writing system is designed in such a way that the verso dots appear perfectly between the recto dots. In a scanned braille image, the recto and verso dots are found connected to each other. But, the relief is in that those dots have different color orientation. Therefore, the technique used in this work exploits this behavior to separate the dots. The algorithm is adopted from [3]. It works by filtering the image column-wise with a window of 8 by 1 pixels and loop through all columns in the image. A window size of eight is selected because in the experiment it is observed that the average window height for recto dots and verso dots is 8 and 7 pixels respectively. Therefore, a general window size of 8 pixels is used to work on the dots of both sides. Part of the MATLAB implementation of the code can be seen in figure 4.5.

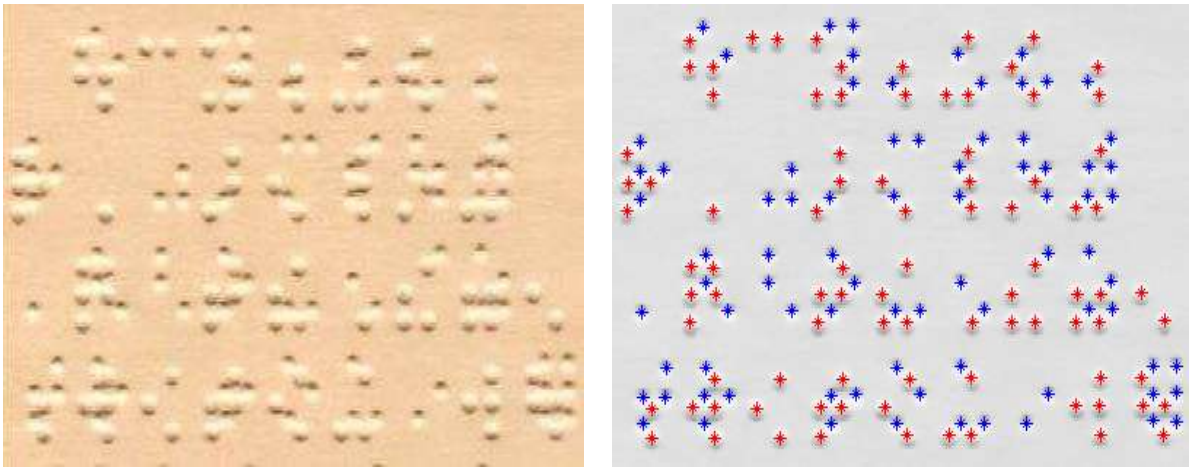
```
%separate recto and verso...
Newval=uint8([255;255;255;255;0;0;0;0]);
windowHeight = 8; windowWidth =1;
imagej=ones(size(img3));
RectoImg=zeros(imageHeight,imageWidth,dim);
VersoImg=zeros(imageHeight,imageWidth,dim);
for i = 1:imageWidth - windowWidth + 1
    j=1;
    while j < imageHeight - windowHeight + 1
       if imagej(j,i)==255
            j=j+8;
        else
            m=j;
            j=j+1;
            window = img3(m:m + windowHeight - 1, i:i + windowWidth - 1);
            if (window(1, 1) ~= (255||immean) && window(2, 1) == 255 ...
                && window(3, 1) == 255 && window(7, 1) == 0 && ...
                 window(8, 1) == 0);
               RectoImg(m:m + windowHeight - 1,i:i + windowWidth - 1) = ...
                    immultiply(window , newval);
                imagej(m:m + windowHeight - 1,i:i + windowWidth - 1) = 255;
            elseif (window(1, 1) ~= immean && window(2, 1) == 0 && ...
                    window(3, 1) == 0 && window(6, 1) == 255 && ...
                    window(7, 1) == 255 &&  window(8, 1) ~= immean);
               VersoImg(m:m + windowHeight - 1,i:i + windowWidth - 1) = ...
                    immultiply(window, flipdim(newval, 1));
                imagej(m:m + windowHeight - 1,i:i + windowWidth - 1) = 255;
            end
        end
    end
end
```

Figure 4.5 Part of MATLAB code for front and back side dot separation

The algorithm starts by creating two separate arrays to hold the recto and verso dots. For every window, the filter checks if the upper pixels are white and the lower are black. If this check results true, it registers it to the recto array. But if the upper pixels are black and the lower pixels are white, the result is written on the verso array.

Sample double-sided braille image and the result of separating the recto and verso dots can be seen in figure 4.6. The filtering is made in such a way that it will not check same region again. This technique had reduced computational time and number of false recognitions.

(a) (b)



(c) (d)

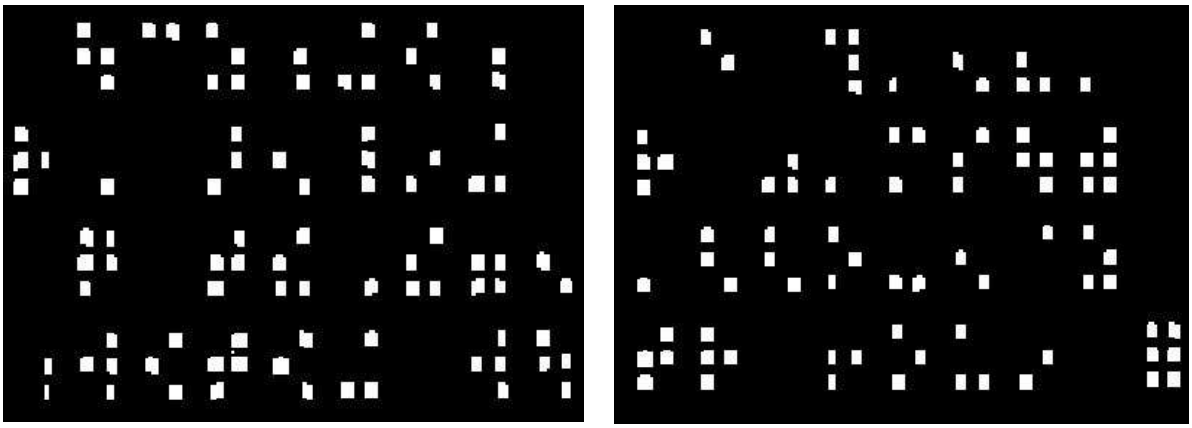


Figure 4.6 Recto and Verso dots separation; (a) scanned RGB image, (b) Gray Scale image showing the recto and verso dot position (Red- Recto, Blue - Verso), (c) recto dot regions after separation and (d) verso dot regions after separation.

This algorithm is implemented on MATLAB and had produced the desired result on any kind of braille documents. Though it is designed to separate the recto and verso dots, it can also help to detect regions of dots in single-sided writing as well. At the end of this process, we have two binary images holding the recto and verso dot regions separately. But, it is to be recalled that as the level of noise increase, defects start to be noticed. Some of the defects identified are:-

- A dot region in the threshold image starts to lose form. As a result, it is difficult to find the dot regions in this process.

- Dots will be highly connected and due to imperfection of the thresholing operation, a dot region may be detected with fewer pixels, none dot region may be detected as a dot region and a dot region may be detected with many pixels.

These and other problems highly affect subsequent operation of braille cell formulation. Therefore, in order to reduce the effect of this problem, morphological operations are performed that can improve the shape and size of detected regions. On average, after detection, a dot area has 3 pixels height and 4 pixels width. But, some dots may be identified as two or more separate regions having 1 pixel width. Therefore, to keep these regions as valid ones, we have implemented an algorithm that performs a neighborhood search and connect such separate regions. In addition, it is observed that some of the detected dot regions do not form perfect rectangles. This will diminish the vertical space between lines of dot in a cell. Therefore, another morphological operation is performed that improves the shape of dot regions detected. Sample binary images of braille before and after morphological operation are performed can be seen in figure 4.7 below.

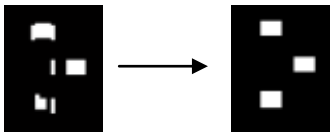


Figure 4.7 Binary braille dote regions before and after morphological improvement

4.5. Braille Cell formulation

In order to group the braille dots into cell; the technique used in this research is the construction of grid. Grids are constructed by using the horizontal and vertical projection profile of perfectly positioned braille dot. Therefore, in this work, only perfectly positioned braille dots are used in building the grid. To do this, an algorithm that performs automatic area analysis and define the average area 'K' for the dots and filters out dots that are less than or greater than 2±k pixel counts is implemented. The lower and higher bounds are defined with experiment to work for any noise level braille documents.

After excluding improperly sized dots, which bias the result of horizontal and vertical projection, a few numbers of dots in the image remain. With the properly sized and placed dots, the horizontal and vertical projection is developed and produces the expected result.

Part of the MATLAB implementation of this algorithm is shown in figure 4.8 below.

```
% Perform area analysis with 4 connectivity
imgbw = bwconncomp(img4bw, 4);
label = labelmatrix(imgbw);
bwarea = regionprops(imgbw, 'Area');
K=round(mean(bwarea.Area);

% Remove big pixels
rimage2 = ismember(L, find([bwarea.Area] <= K + 2 ));

% Remove small pixels
rimage2 = ismember(L, find([bwarea.Area] >= K - 2 ));

%% Horizontal Projection
[imageheight, imagewidth] = size(rimage2);
hp = zeros(,1);
for i = 1: imageheight
    for j = 2: imagewidth
        if rimage2(i,j)~=rimage2(i,j-1)
            hp(i) = hp(i)+1;
        end
    end
end

%% Vertical projection front
vp = zeros(1, imagewidth);
for j = 1: imagewidth
    for i = 2: imageheight
        if rimage2(i,j)~=rimage2(i-1,j)
            vp(j) = vp(j)+1;
        end
    end
end
```

Figure 4.8 Part of the MATLAB code for building grid

The coordinate values to build the grid line are derived from the position of the center points of the connected components in the horizontal and vertical projection results. From the center points in the vertical projection, we build vertical lines connecting all the points in ‘y’ axis. From center points in the horizontal projection, we build horizontal lines connecting all the points in ‘x’ axis.

Sample braille grid image constructed with all the dots in the image and after removing improperly sized dots can be seen in figure 4.9 below.

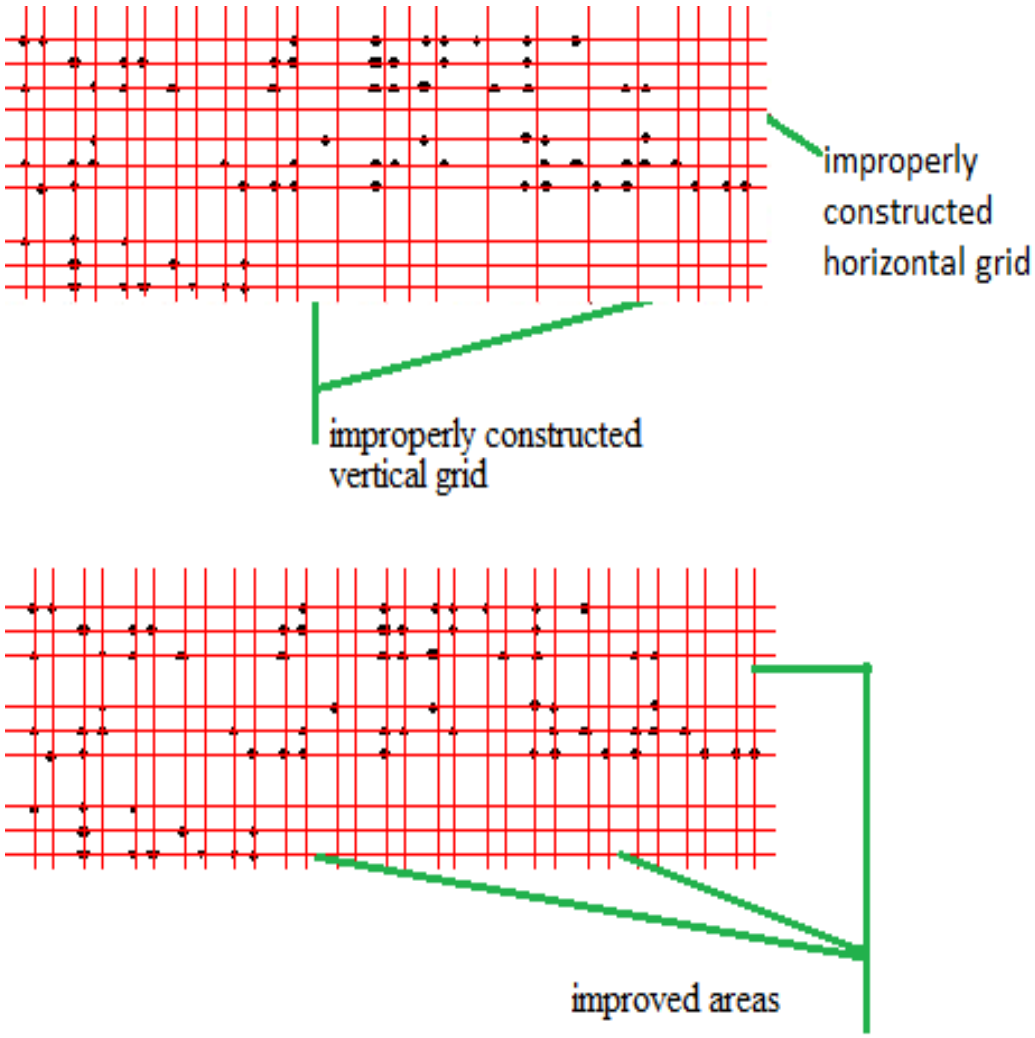


Figure 4.9 Results of grid construction; (Top) with all dots, (Bottom) with selected dots

This way we can build grids line over the points only where there is a dot. But, it is a must to find the possible dot regions as well although they do not have a dot at the time. Therefore, in order to find those missed points, we have designed a grid refining algorithm that analyzes the horizontal and vertical spacing between dots in a cell and between cells themselves. This algorithm builds grid line where they should exist. The output of this operation is shown in figure 4.10 below.

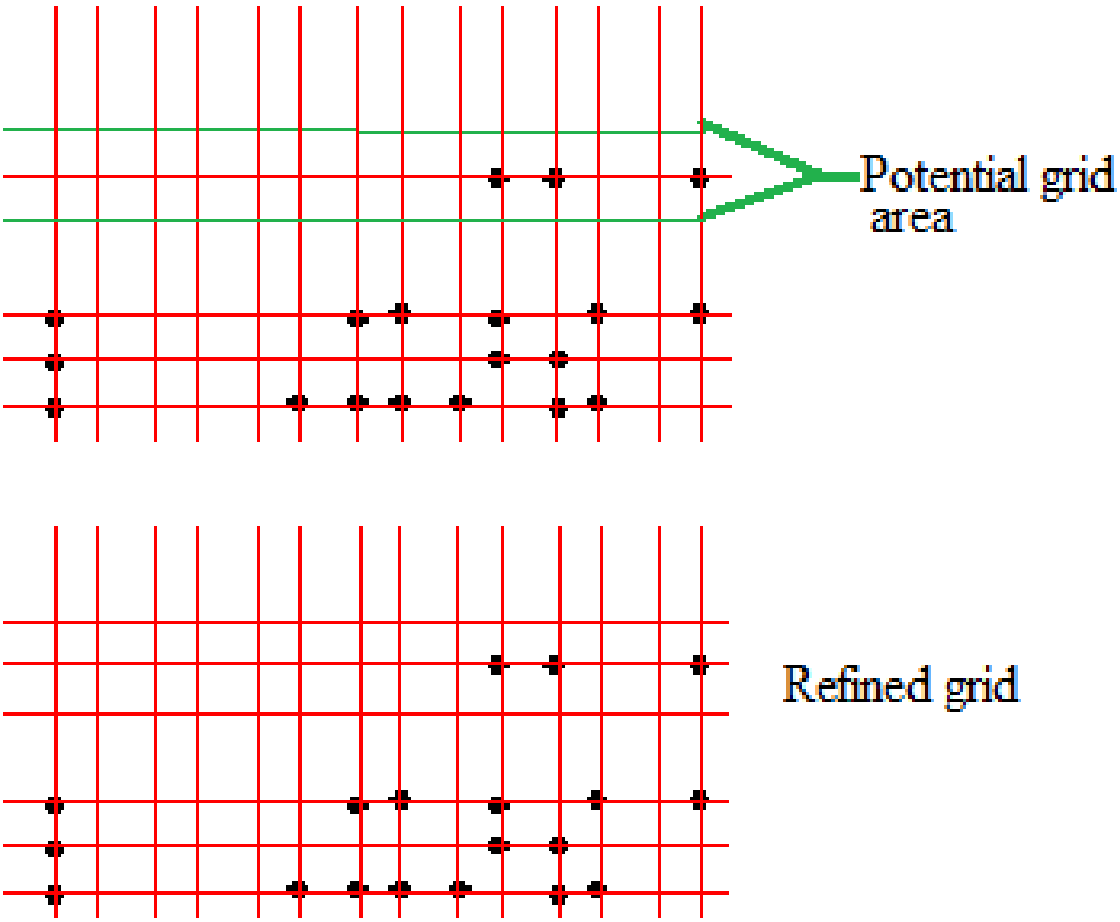


Figure 4.10 Results of grid refinement

4.6. Amharic Braille feature extraction

The extraction of braille feature is performed by checking the existence of dot on or around the intersection points of the grid line. Six intersection points which are closer to each other make up feature for single cell. This way the feature of every cell will be extracted and saved in an array. The array holding the features is an array of zeros and ones. Zeros indicate that there is no dot in the specified position while ones indicate the existence of dot. The feature for a single cell is composed of six binaries. However, for simplicity the six binary values will be converted to decimal code representation.

The MATLAB implementation of the algorithm is given in figure 4.11 below.

```
%% save coordinates to array for recto...

pp1=[];
cc1 = 1;

for nt = [1:num12]
    X = centroids12(nt:nt,2);
    for cn = [1:num22]

        Y=centroids22(cn:cn,1);
        pp1(cc1,1) = X;
        pp1(cc1,2) = Y;
        cc1 = cc1 + 1;

    end
end
%% extract features

A=[];
si=size(pp1);
n=1;

segi=zeros(size(rimage));
sizei=size(centroids12,1);
sizej=size(centroids22,1);

for o =(1:3:sizei) % loop through the first line of every cell
    c=o;
    for w =(1:sizej)
        j=centroids22(w:w,1);
        x=round(j);
        for y =(c:c+2)
            i = centroids12(y:y,2);
            y=round(i);
            if rimage(y,x)==1 || rimage(y,x-1)==1 || rimage(y,x+1)==1
                B=1;
                A=[A;B];
            else
                B=0;
                A=[A;B];
            end
        end
    end
end

%% convert extracted binary features to decimal code representation

d=[]; % Array to hold the decimal codes
for i=1:6:(size(A,1)-5)
    w=A(i:i+5);
    decval=w(1)*1 + w(2)*2 + w(3)*4 + w(4)*8+ w(5)*16 + w(6)*32;
    d=[d; decval]; % append to d
    txtout = fopen('result_01.txt', 'w'); % Save to text file
    fprintf(txtout, '%d\n', d);
end
```

Figure 4.11 MATLAB code for feature extraction

4.7. Translation

The final step in the designed system is translation. Here we convert the feature, which is a decimal code representation for every cell, into its corresponding Amharic letters to get its translation. Previous works have used classification techniques such as SVM, neural network and j48. However, braille translation is a one-to-one matching. Therefore, building a lookup table and matching features against it will be sufficient.

To build the translator, it is first required to understand the context of the language. In Amharic braille writing a single cell can represent a single character, part of a character, a numeral or punctuation mark. All the 34 core characters have a single cell representation while the six variants of the 34 core characters, which make a total of 204 (34*6), are represented by two cells. The numerals are represented with two cells while punctuations are represented with either two or three cells. Sample characters and their normalized representation can be seen from table 4.2.

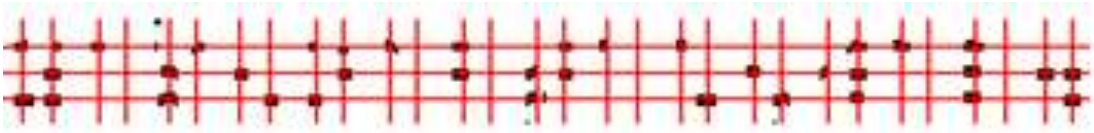
characters	First cell							Second cell							Third cell							Lookup table representation
	1	2	3	4	5	6	Decimal Code	1	2	3	4	5	6	Decimal Code	1	2	3	4	5	6	Decimal Code	
ሀ	1	1	0	0	1	0	19	0	1	0	0	0	1	34	0	0	0	0	0	0	0	1
ሁ	1	1	0	0	1	0	19	1	0	1	0	0	1	37	0	0	0	0	0	0	0	2
ሂ	1	1	0	0	1	0	19	0	1	0	1	0	0	10	0	0	0	0	0	0	0	3
ሃ	1	1	0	0	1	0	19	1	0	0	0	0	0	1	0	0	0	0	0	0	0	4
ሄ	1	1	0	0	1	0	19	1	0	0	0	1	0	17	0	0	0	0	0	0	0	5
ህ	1	1	0	0	1	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
ሆ	1	1	0	0	1	0	19	1	0	1	0	1	0	21	0	0	0	0	0	0	0	7
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
፩	0	0	1	1	1	1	60	1	0	0	0	0	0	1	0	0	0	0	0	0	0	283
፭	1	1	1	1	1	1	63	1	0	0	0	0	0	1	0	0	0	0	0	0	0	294
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
፡	0	0	0	0	0	1	32	0	0	1	0	0	0	4	0	0	0	0	0	0	0	313
…	0	0	1	0	0	0	4	0	0	1	0	0	0	4	0	0	1	0	0	0	4	305

Table 4.2 Sample Amharic features and their normalized representation

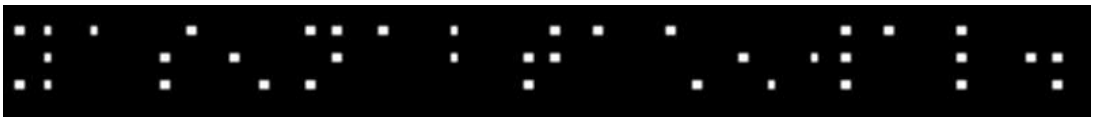
As can be seen in lookup table Amharic characters have vowel and consonant combinations. The decimal codes for the vowels are 34, 37, 10, 1, 17 and 21 which represent the 1st, 2nd,

3rd,4th, 5th and 7th sounds of a 7 order character. Therefore, any number followed by those decimal values is a consonant for a character represented with two cells. Any decimal code followed by preceded by 60 is a Latin numeral while any decimal value preceded by 63 is a geez numeral. Every punctuation mark has its own decimal representation which could be in two or three cell. Every vowel and consonant combination, every numeral and every punctuation mark is assigned a single unique numeral for uniformity. Finally a lookup table is built with python that contains the character and their respective numeral representation. The translation code will group numbers and match the cell representation of the features extracted and saved in the text file against their corresponding Amharic characters in the lookup table. For example if converting a binary future value to decimal results in number 19 followed by 37 it is assigned the unique number 2 in lookup table and the translation is Amharic Character "ሁ". The complete lookup table can be found in appendix I. A single word along with its features and translation steps can be seen in figure 4.12 below.

-Grid construction for feature extruction



- Extracted features (dots represented by 4 pixels)



- Features of dot in binary representation

101111 100000 011100 010001 101110 100000 110000 011110 100000 100001 010001 010111 100000 111000 010011

- Fatures of dot in decimal representation with consonant voul grouping

61,1 14,34 29,1 3 30,1 33,34 58,1 7 50

- Lookup table representation for consonant voul groupings

158 29 95 69 74 78 130 13 319

- Translation agrainst lookup table

ያ ሰ ና ብ ታ ች ዋ ል ::

Figure 4.12 Steps of translation

4.8. Performance Evaluation

The performance of the double-sided Amharic braille recognizer was tested using three separate datasets classified based on noise level; clean, medium noisy and high noise images. Each dataset contains images of 10 double-sided and 2 single-sided Amharic braille documents. In order to report the performance of the system on single-sided and double sided separately, each dataset is gain classified into two; the first group contains 2 single-sided braille images in each noise level and the second dataset contains 10 double-sided braille images in each noise level.

The results achieved are detailed in table 4.4 and 4.5 below. First, the performance of the system is measured after segmentation on dots level. The total number of potential dots is defined by manually counting the total dots region in every of the dataset which includes both the front side and back side dot. There are two kinds of errors that affect the performance. These are dots added (false positive) and dots missed (false negative). Dots added is the total number of none dot areas recognized as dot while the dots missed is a true dot region recognized as a background. Segmentation accuracy is measured as a percentage of correct recognition against total recognition.

Test Set	Total number of Potential dot areas in the braille sheet (A)	Dots added (B)	Dots missed (C)	Segmentation Accuracy $\frac{(A-B-C)}{A}$ (%)
Test set 1 (clean)	14,472	0	0	100.00
Test set 2 (medium level noise)	13,932	28	11	99.72
Test set 3 (high noise)	14,112	438	107	96.14
Average	42,516	466	118	98.63

Table 4.4 Performance results for single-sided segmentation

Test Set	Total number of Potential dot areas in the braille sheet (A)	Dots added (B)	Dots missed (C)	Segmentation Accuracy $\frac{(A-B-C)}{A}$ (%)
Test set 1 (clean)	68,040	0	0	100.00
Test set 2 (medium level noise)	69,600	174	210	99.45
Test set 3 (high noise)	68,718	12054	1971	79.59
Average	206,358	12228	2181	93.02

Table 4.5 Performance results for double-sided segmentation

As can be seen in table 4.4 and 4.5 the performance of the system on both single-sided and double-sided clean documents is 100%. However, the performance reduces as the level of noise in braille documents increases. It is also observed that the average performance of the system on single-sided is slightly better than that of the average performance registered on double-sided braille documents. This is because single-sided documents are less noisy than that of double-sided documents. But, the main reason lays in the fact that the number of dots in double-sided documents is almost twice that of dots in single-sided. This makes the number of connected dots in double-sided braille to be very high as compared to single-sided braille dots which are rarely connected.

As can be seen from the table an average accuracy of 98.63% and 93.02% are registered on segmentations of single-sided and double sided braille documents respectively. Sample braille image and segmentation performance of the system on the three noise levels can be seen in figure 4.13 below.

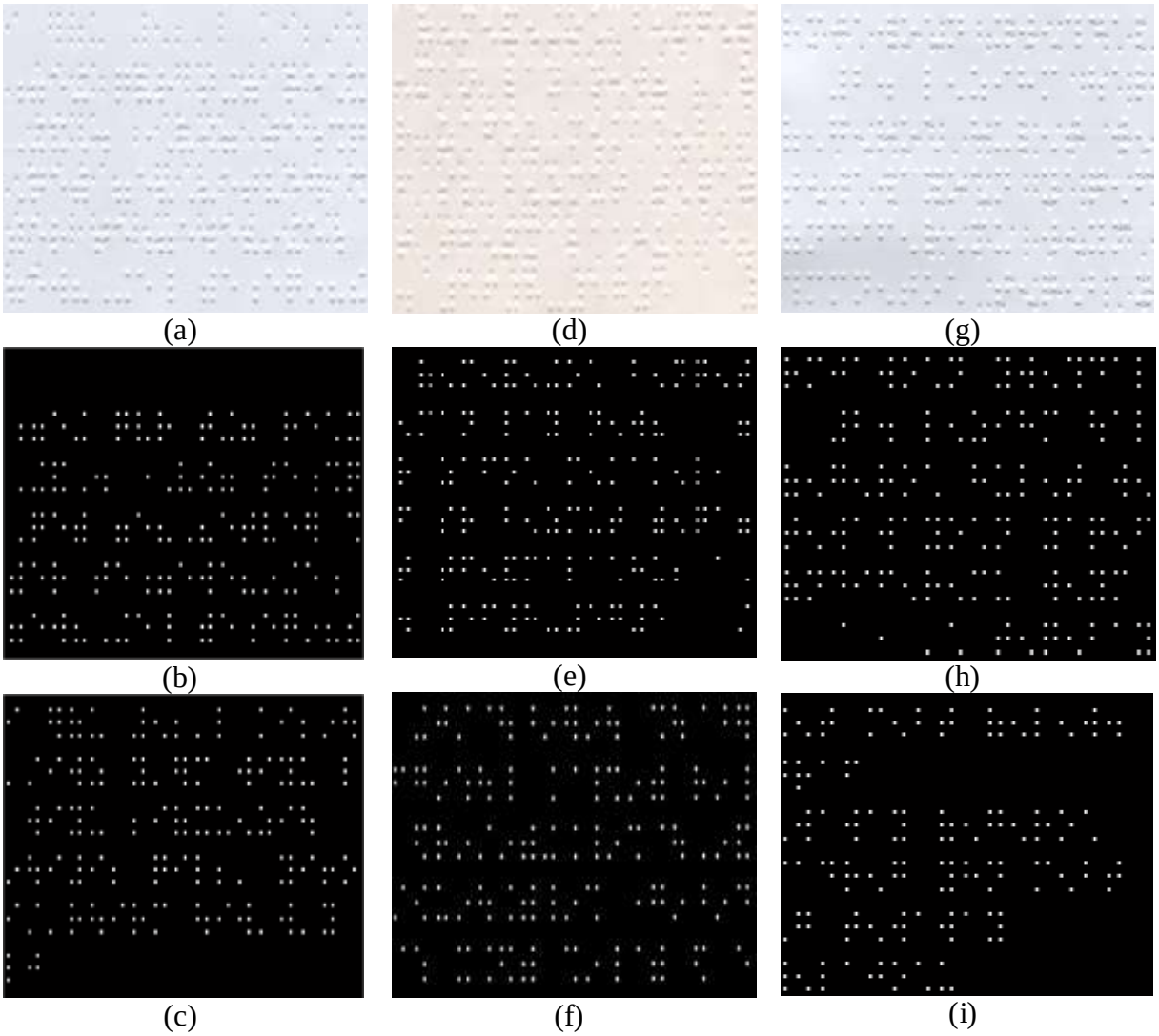


Figure 4.13 Segmentation results (a) Clean braille image (b) recto dots extracted from a (c) verso dots extracted from a (d) Average noisy braille image (e) recto dots extracted from d (f) verso dots extracted from d (g) High noisy braille image (h) recto dots extracted from g (i) verso dots extracted from g

It is also measured the accuracy of the system at the level of correctly translated characters. At this level the overall performance of the system is tested. The results on single-sided braille and double-sided braille images are detailed in table 4.6. and 4.7 respectively.

Test Set	Total number of characters (A)	Incorrectly recognized Characters (B)	Segmentation Accuracy (%)
Test set 1 (clean)	2,412	0	100
Test set 2 (medium level noise)	2,322	6	99.74
Test set 3 (High level noise)	2,352	258	89.03
Average	7,086	264	96.27

Table 4.6 Overall performance achieved on single sided recognition

Test set	Total number of characters (A)	Incorrectly recognized Characters (B)	Segmentation Accuracy (%)
Test set 1 (clean)	11,340	0	100.00
Test set 2 (medium level noise)	11,600	57	99.51
Test set 3 (High level noise)	11,453	4015	64.94
Average	34,393	4072	88.16

Table 4.7 Overall performance achieved on double-sided recognition

As can be seen from table 4.6 and 4.7 above, the system had achieved an overall accuracy of 100% on clean braille images of both single-sided and double-sided braille documents. But, the overall performance of the system fails as noise increase, the same way as the test in segmentation stage. This is because the segmentation output highly affects the grid construction process. Since feature extraction depends on correctly positioned grids, miss placing a single grid can highly affect the accuracy in extracting features. But, it is also to be noted that correctly placed grids have the effect of increasing accuracy by cutting out incorrectly segmented dots. The overall accuracy achieved is 96.27% for single-sided and 88.16% for double-sided braille documents.

4.9. Discussion and Challenges

The system designed has registered a remarkable result on both clean and average single-sided and double-sided braille documents. The reason for achieving such result is due to effectiveness in the thresholding techniques and dot extraction mechanism used in this work, which also helps to exclude noise from the image. It is also worth mention that the grid construction and refining technique introduced in this work has also contributed a significant role in extracting features correctly. It is difficult to compare performance of the current system against previous works for single-sided braille recognition. But, since double-sided braille recognition is more difficult, due to the addition in noise and number of feature to be extracted, it is sound to say that the results registered are commendable as compared to previous works for single-sided. Sample braille image and translation result can be found on Appendix II.

It is also to be noted that, the performance of the current system is highly affected on high level noise documents. The errors encountered on high noisy documents are due to the quality of the braille document and the performance of the thresholding operation which in turn affects subsequent operations. In this work we have tried to adopt a thresholding algorithm that dynamically calculates the threshold values for any kind of image. But, the performance of the thresholding operation is slightly affected with noisy and degradation. When dots are degraded the thresholding algorithms fails to build them with the proper orientation of black and white. Some of the problems identified during thresholding can be seen in figure 4.14 below. Red boxes represent segmentation errors observed.

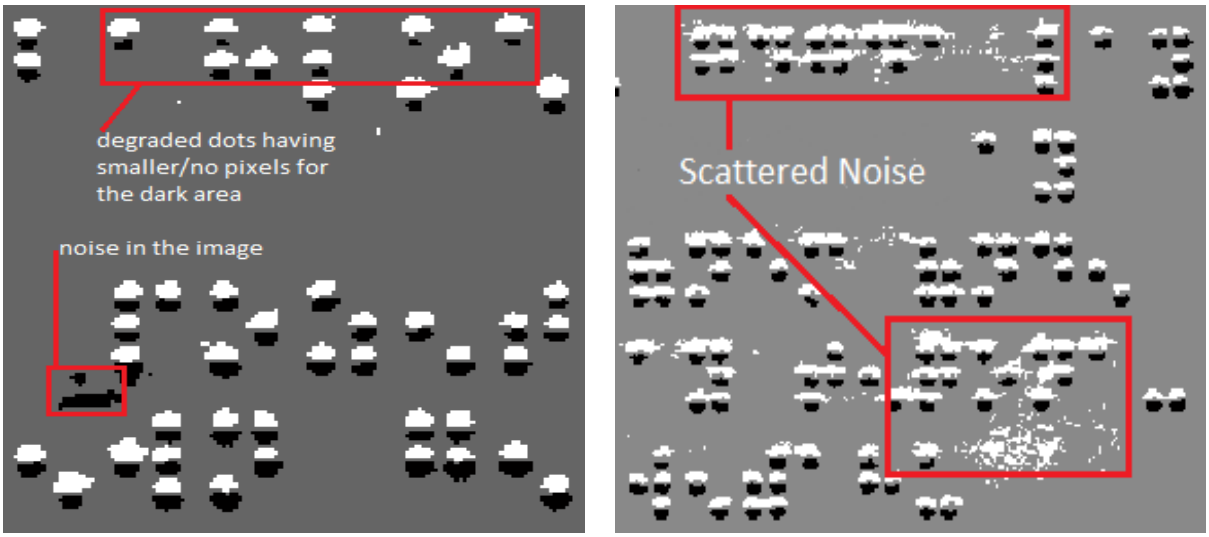


Figure 4.14 Problems identified in thresholding

These slight defects are seen to have huge impact on the performance of the segmentation and feature extraction processes.

Another thing worth mentioning is, skew detection and correction. Due to time limitation, this system is not designed to detect and correct skew. As a result it will not work properly for skewed images with a skew angle less that -0.5 and greater than +0.5. However, we have tried to manually correct skew images and check if the recognizer performs properly. The test had shown that the recognizer performed very well.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1. Conclusion

For visually impaired people, lack of vision has been an obstacle to access printed contents as well as write their ideas on paper. In the long effort of solving this important problem Luis Braille invented a writing system called brialle, which is named after him. Braille is an efficient means of writing and communication for the blind. It has been widely adopted by different counties for use with their own languages. Amharic is one of the languages that Adopted Braille.

The invention of Braille has solved the communication through written form problem between visually impaired and vision people. However, the use of Braille for communication with vision society requires vision people to understand braille. OBR is a technology with the answer for this problem. It is a system that converts hardcoded braille documents into machine readable and editable text that vision people can understand. In addition, OBR has allowed preservation of Braille works and it is also an easy way to copy Braille.

In this work an Amharic Braille Recognition System is developed. The developed system is able to recognize both single-sided and double-sided Amharic braille documents. Amharic, unlike English has very large number of characters and hence very complex braille representation. In order to accommodate the large number of characters, a single character can be represented with up to 3 cells. Understanding this property is important in building braille recognition system for the language. In this work, different techniques for the recognition process adopted from previous works with some modification and also designed new algorithms for improving the performance of the system. Image noise filtering techniques that were recommended by previous researchers on Braille images have been tested and used. A dynamic thresholding technique is adopted that computes different threshold values for different images automatically. The main problem, separation of front and back side dots, have been solved by using a sliding window that filters the image to detect dot areas and separate the dots into front and back side dots.

An improved grid construction technique, that automatically define and sort dot size and uses only perfectly sized dots, have been designed to build a grid over the image. We have also

introduced a technique that refines the grid based on the horizontal and vertical spacing property between dots and between cells. This technique works by taking a sample from perfectly constructed grid lines. It has the advantage of flexibility in automatically defining the inter cell and intra cell distance rather than calculating the dot size and distance which previous researches have been doing.

The segmentation technique and most importantly the grid construction process have enabled the detection of dots with high performance. A prototype has been designed for the system and its performance have been tested against three sets of braile images; clean, average noisy and high noisy. The designed system has registered higher results for clean and average noisy braille documents while a small error is registered in recognizing high noisy Braille documents. As a result, an average accuracy of 96.07% and 86.16% has been achieved for single-sided and double sided braille documents respectively.

The commendable results achieved are due to the effectiveness in the thresholding, segmentation and feature extraction techniques. The small error observed on the noisy documents is due to the defects on the Braille document that alter the shape of the dots. Such problems have an impact on the thresholding performance and this in turn affects accuracy in segmentation of dots.

The performance of any character recognition system is highly dependent on the quality of image it process. As a result, a lot of consideration should be done in selecting the preprocessing techniques to use. The preprocessing technique should be able to provide the image as per the requirement for the segmentation. The major challenge unique double-sided recognition is the separation of verso and recto dots. In order to achieve good result in separation, the algorithm should focus on the unique property of the dots of images of recto against verso. Such uniqueness can be found in the orientation of bright and shadow regions. In this work, exploiting this nature of braille image has enabled separation of recto and verso dots.

This work has added to the search for comprehensive Amharic Braille recognition system by working on double-sided braille recognition that has not been locally attempted so far. The performance of the system on single-sided documents is also better compared to previous works for Amharic braille.

5.2. Recommendation

This study has attempted to adopt and design an Amharic OBR that recognizes both single-sided and double-sided Braille documents. Based on the findings from the study, the following recommendations are forwarded for future researches in the area,

- The proposed system detects dots based on the bright and dark region orientation. A dot area to be detected should have both properties. But, due to quality of the Braille document, in the threshold image either of the two regions might not appear. Therefore, this dot will be considered as a background. Future works can focus on recovering such dots to improve the performance of the system.
- In this work a vertical scan is used to checks the existence of bright and shadow areas. In case of skewed images, correcting the skew angle will make the bright and shadow area to take a horizontal position rather than vertical orientation. As a result this technique will fail to detect most of the dots. Therefore, future works should search for recognition techniques of skewed braille images.
- In this work separation of recto and verso dots is dependent on the black and white regions of dots formed during scanning. However, braille documents are mostly degraded and it is difficult to get a clean image with clear separation for the black and white regions. Therefore, future works can explore techniques that can crop out dots by matching features with model designed using a learning system which is preferable.
- Braille documents by their very nature are prone to physical damaged. As a result almost all braille images are noisy and noise decreases the performance of a system. But, the overall error registered in this system is very small. Therefore, integrating post recognition process that corrects wrongly recognized words by checking them against their closest match in a dictionary can improve performance of the system.
- In this work, no effort was made in keeping the format of the Braille after translation. However, keeping the format might be important in such cases where the content of the Braille is a poem. Therefore, future works can integrate techniques for keeping the format of Braille after translation.

Reference

[1] A. Papandreou and B. Gatos, 'A novel skew detection technique based on vertical projections', Document Analysis and Recognition (ICDAR), International Conference on (pp. 384-388), IEEE, 2015.

[2] A. Abdelmonem, B. El-Hoseiny, C. Ali, D. Emara, E. Hafez and F. Gamal, 'Dynamic Optical Braille Recognition (OBR) System', IPCV, pp. 779-786, 2009.

[3] A. Al-Salman, Y. AlOhli, M. AlKanhal and A. AlRajih, 'An Arabic Optical Braille Recognition System', Proceedings of the International Conference in Information and Communication Technology & Accessibility (ICTA'07), pp. 81-86, 2007.

[4] A. Antonacopoulos and D. Bridson, 'A robust Braille recognition system', In Document Analysis Systems VI, pp. 533-545, 2004.

[5] A. Cheung, M. Bennamoun and N. Bergmann, 'An Arabic optical character recognition system using recognition-based segmentation', Pattern Recognition, vol. 34, no. 2, pp. 215-233, 2001.

[6] A. Mousa, H. Hiary, R. Alomari and L. Alnemer, 'Smart Braille System Recognizer', IJCSI International Journal of Computer Science Issues, vol. 10, no. 6, 2013.

[7] Al-Saleh, 'Dot Detection of Braille Images Using A Mixture of Beta Distributions', Journal of Computer Science, vol. 7, no. 11, pp. 1749-1759, 2011.

[8] B. Sankur, M. Sezgin 'Survey over image thresholding techniques and quantitative performance evaluation', Journal of Electronic Imaging , vol. 13, no. 1, p. 146-165, 2004.

[9] B. Yohannes, 'Recognition of Noisy Ethiopic Braille Documents', MSc. Thesis, Addis Ababa University, School of Information Science, Addis Ababa, Ethiopia 2013.

[10] C. Kothari, Research methodology. New Delhi: New Age International (P) Ltd., 2004.

[11] C. M. Ng, V. Ng and Y. Lau, 'Statistical Template Matching for Translation of Braille', In Proceedings of the Spring Conference on Computer Graphics (SCCG), pp. 197-200, 1999.

[12] C. Mackenzie, World braille usage. [Paris]: UNESCO, 1954.

[13] C. Ng, V. Ng and Y. Lau, 'Regular Feature Extraction for Recognition of Braille', Computational Intelligence and Multimedia Applications, ICCIMA '99. pp. 302-306, 1999.

[14] Csa.gov.et, 'Census 2007', 2015. [Online]. Available: http://www.csa.gov.et/index.php/2013-02-20-14-51-51/2013-04-01-11-53-00/census-2007. [Accessed: 03- Jun- 2015].

[15] D. Bradley and G. Roth, 'Adaptive Thresholding using the Integral Image', Journal of Graphics, GPU, and Game Tools, vol. 12, no. 2, pp. 13-21, 2007.

[16] D. Saad Al-Shamma and S. Fathi, ”Arabic Braille Recognition and Transcription into Text and Voice”, 2010 5th Cairo International Biomedical Engineering Conference Cairo, Egypt, December 16-18, 2010.Arabic to text and voice

[17] Data.worldbank.org, 'Ethiopia | Data', 2015. [Online]. Available: http://data.worldbank.org/country/ethiopia. [Accessed: 03- Jun- 2015].

[18] E. Chekol, 'Recognition of Amharic Braille Documents', MSc. Thesis, Addis Ababa University, Department of Information Science, 2010.

[19] E. Davies, Computer and machine vision. Waltham, Mass.: Elsevier, 2012.

[20] Encyclopedia Britannica, 'Amharic language', 2013. [Online]. Available: http://www.britannica.com/EBchecked/topic/20500/Amharic-language. [Accessed: 03- Jun- 2015].

[21] Ethiopians.com, 'Ethiopian Writing System - Baye Yimam', 2015. [Online]. Available: http://www.ethiopians.com/bayeyima.html. [Accessed: 03- Jun- 2015].

[22] G. Morgavi and M. Mauro, 'A neural network hybrid model for an optical braille recognitor', International Conference on Signal, Speech and Image Processing (ICOSSIP), 2002.

[23] H. M. Taha,. Robust braille recognition system using image preprocessing and feature extraction algorithms (Doctoral dissertation, Universiti Tun Hussein Onn Malaysia) 2014.

[24] J. Abualkishik, M. Abdallah and O. Khairuddin, 'Quranic Braille System', International Journal of Humanities and Social Sciences, pp. 313-319, 2009.

[25] J. Bhattacharya and S. Majumder, 'braille character recognition using a generalized feature vector', Computer Networks and Intelligent Computing Springer Berlin Heidelberg, pp. 171-180, 2011.

[26] J. Mennens, L. Van Tichelen, G. Francois and J. Engelen, 'Optical recognition of Braille writing using standard equipment', IEEE Trans. Rehab. Eng., vol. 2, no. 4, pp. 207-212, 1994.

[27] J. Mennens, L. Van Tichelen, G. Francois and J. Engelen, 'Optical recognition of Braille writing', IEEE Trans. Rehab. Eng., vol. 2, no. 4, pp. 207-212, 1993.

[28] J. Sadri, C.Y.Suen and T.B.Bui, 'Application of support Vector Machine for Recognition of Hand Written Arabic/Persian Digits', Center for Pattern Recognition and Machine Intelligence, 2004. MVIP vol. , 2004

[29] L. Shapiro and C. Stockman G., Computer Vision. Prentice Hall, 2011.

[30] L.Wong, W., Abdulla, & S. Hussmann, A software algorithm prototype for optical recognition of embossed Braille. In Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on (Vol. 2, pp. 586-589). IEEE, 2004.

[31] M. Cheriet, N. Kharma, L. Liu C. and C. Suen, Character recognition systems: a guide for students and practitioners. Hoboken, N.J.: Wiley-Interscience, 2007.

[32] M. Hassen, 'Recognition of Amharic Braille Using Direction Field', MSc. Thesis, Addis Ababa University, Department of Computer Science, 2011.

[33] M. Meshesha, 'A Generalized Approach to Character Recognition of Amharic Texts', MSc. Thesis, Addis Ababa University, School of Information Studies for Africa, 2000.

[34] M. Wanas N, A. Said D, H. Hegazy N and M. Darwish N, 'A study of local and global thresholding techniques in text categorization', In Proceedings of the fifth Australasian conference on Data mining and analystics, vol. 61, pp. 91-101 Australian Computer Society, Inc., 2006.

[35] M. Yousefi, M. Famouri, B. Nasihatkon, Z. Azimifar and P. Fieguth, 'A robust probabilistic Braille recognition system', International Journal on Document Analysis and Recognition (IJDAR), vol. 15, no. 3, pp. 253-266, 2011.

[36] Matlab, Mathworks, Matlab Image processing toolbox, 2013

[37] N. Falcon, C. Travieso, J. Alonso and M. Ferrer, 'Image processing techniques for braille writing recognition', Computer Aided Systems Theory–EUROCAST 2005 Springer Berlin Heidelberg, pp. 379-385, 2015.

[38] R. Gonzalez, R. Woods and S. Eddins, Digital Image processing using MATLAB. Upper Saddle River, N.J.: Pearson Prentice Hall, 2004.

[39] R. Jain, R. Kasturi and B. Schunck, Machine vision. New York: McGraw-Hill, 1995.

[40] S. T and V. Udayashankara, 'A Review on Software Algorithms for Optical Recognition of Embossed Braille Characters', International Journal of Computer Applications, vol. 81, no. 3, pp. 25-35, 2013.

[41] S. Tadesse, 'Feature Extraction And Classification Schemes For Enhancing Amharic Braille Recognition System', Msc. Thesis, Addis Ababa University, School of Information Science, 2011.

[42] Simpson, The rules of unified English Braille, 2nd ed. International Council on English Braille(IECB), 2013.

[43] T. Alemu, 'Recognition of Amharic Braille', MSc. Thesis, Addis Ababa University, Department of Information Science, 2009.

[44] T. Shreekanth, V. Udayashankara,. An Algorithmic Approach for Double Sided Braille Dot Recognition Using Image Processing Techniques. In International Journal of Image Processing and Visual Communication ISSN (Vol. 2, issue 4). 2014

[45] T. Pang-Ning, S. Michael and K. Vipin, Introduction to data mining. Addison Wesley, 2006.

[46] UNESCO, 'World Braille Usage', National Library Service for the Blind and Physically Handicapped Library of Congress, Perkins International Council on English Braille, Washington, D.C, 2013.

[47] UNESCO, 'World Braille Usage', National Library Service for the Blind and Physically Handicapped Library of Congress, Perkins International Council on English Braille, Washington, D.C., 1990.

[48] V. Udayashankara, 'An Application of Eight Connectivity based Two-pass Connected-Component Labelling Algorithm For Double Sided Braille Dot Recognition', International Journal of Image Processing (IJIP), 8(5), 294., vol. 8, no. 5, pp. 220 - 396, 2014.

[49] Who.int, 'WHO | Blindness', 2015. [Online]. Available At: http://www.who.int/topics/blindness/en/ [Accessed: 05- Sep- 2014].

[50] X. Hermida, A. Rodríguez and F. Rodríguez, 'A Braille OCR for Blind People', Proceedings of ICSPAT-96. Boston (USA). 1996.

[51] **ኢትዮጵያ አይነስውራን ብሔራዊ ማህበር** 12**ኛ ዙር ስራ አስፈፃሚ ኮሚቴ የተቋቋመው የአማርኛ ብሬል አሻሻይ ኮሚቴ። የአማርኛ ብሬል አሻሻይ ኮሚቴ ዘገባ። ታህሳስ** 19 **እና** 20 **ቀን** 1995 **ዓ.ም. ለተጠራው አገር አቀፍ ጉባኤ የቀረበ።**

[52] **ነብየልዑል ዮሃንስ ዘመነ ብርሃን አዲስ አበባ ብርሃንና ሰላም ማተሚያ ቤት** 1954 **።**

[53] Omniglot.com,. 'Amharic Alphabet, Pronunciation And Language'. N.p., 2015. Web. 20 Oct. 2015.

Appendix

I. List of the seven braille orders

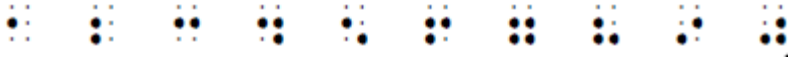
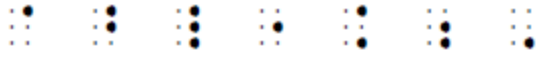
Order1	
Order 2	
Order 3	
Order 4	
Order 5	
Order 6	
Order 7	

Table I List of the seven braille orders

II. Complete list of Amharic Alphabet

Amharic Characters											
order							Labialized				
1st	2nd	3rd	4th	5th	6th	7th					
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ					
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ			ሏ		
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ			ሗ		
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ			ሟ		
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ			ሧ		
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ			ሯ		
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ			ሷ		
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ			ሿ		
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ	ቈ	ቍ	ቋ	ቌ	ቊ
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ			ቧ		
ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ			ቯ		
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ			ቷ		
ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ			ቿ		
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ	ኈ	ኍ	ኋ	ኌ	ኊ
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ			ኗ		
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ			ኟ		
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ			ኧ		
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ	ኰ	ኵ	ኳ	ኴ	ኸ
ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ					
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ					
ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ					
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ			ዟ		
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ			ዧ		
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ					
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ			ዷ		
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ			ጇ		
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ	ጐ	ጕ	ጓ	ጔ	ጘ
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ			ጧ		
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ			ጯ		
ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ			ጷ		
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ			ጿ		
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ					
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ			ፏ		
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ			ፗ		
Amharic Numerals											
፩	፪	፫	፬	፭	፮	፯	፸	፹	፻		
፼	፲	፳	፴	፵	፶	፷	፸	፹	፺		
Punctuation Marks											
፡	።	፣	፤	!	?	(	)	"	"		

Table II.1 Complete list of Amharic Alphabet

III. First Version Amharic Braille

ሀ	01:02:04	ኧ	02:03:04
ለ	01:04:05	ወ	2:3:4:5
ሐ	1:2:4:5	ዐ	01:06
መ	01:02:05	ዘ	01:04:06
ሠ	02:04:05	ዠ	1:3:4:6
ረ	01:03	የ	1:4:5:6
ሰ	01:02:03	ደ	01:05:06
ሸ	1:2:3:6	ጀ	1:2:4;6
ቀ	01:03:04	ገ	1:2:4:5:6
በ	1:3:4:5	ጠ	1:2:5:6
ተ	01:03:05	ጨ	1:2:3:5:6
ቸ	1:3:5:6	ጰ	2:4;5;6
ኀ	1:2:3:4	ጸ	2:4:5:6
ነ	1:2:3:4:5	ፀ	01:03:06
ኘ	1:2:3:4:5:6	ፈ	1:3:4:5:6
አ	1:2:3:5	ፐ	2:3:4:5:6
ከ	01:02:06		

TableIII.1 List of basic Amharic Braille characters in the first version

Vowels	01:04	02:05	03:06	01:05	02:04	02:06	02:06
Variant	1st	2nd	3rd	4th	5th	6th	7th

TableIII.2 List of vowel variant for Amharic Braille characters in the first version

IV. Second Version Amharic Braille

1st		6th		1st		6th	
ሀ	01:02:05	ሀ	1:3:4:6	ኸ	02:03:06	ኸ	5 2:3:6
ለ	04:05:06	ል	01:02:03	ወ	2:4:5:6	ው	02:04:06
ሐ	Not apply			ዐ	Not apply		
መ	01:03:04	ም	02:03	ዘ	1:3:5:6	ዝ	2:3:4:6
ሠ	02:03:04	ሥ	5	ዠ	03:05:06	ዟ	05:06
ረ	1:2:3:5	ር	01:02:05	የ	1:3:4:5:6	ይ	1:4:5:6
ሰ	Not apply			ደ	01:04:05	ድ	1:4:5:6
ሸ	01:04:06	ሽ	01:05:06	ጀ	02:04:05	ጅ	01:02:06
ቀ	1:2:3:4:5	ቅ	04:06	ገ	1:2:4:5	ግ	2:3:5:6
በ	01:02	ብ	04:05	ጠ	2:3:4:5:6	ጥ	1:2:3:5:6
ተ	2:3:4:5	ት	123456	ጨ	01:04	ጭ	03:06
ቸ	01:06	ች	02:05	ጰ	02:03:05	ጵ	3:4:5:6
ኀ	Not apply			ፀ	1:2:3:4:6	ፅ	3:4:5:6
ነ	1:3:4:5	ን	1:2:4:6	ጸ	Not apply		
ኘ	03:04:06	ኝ	02:06	ፈ	01:02:04	ፍ	5 1:2:4
አ	3	እ	03:04	ፐ			
ከ	01:03	ክ	03:05				

Table IV.1 List of basic Amharic Braille characters in the 2nd version

Vowel	None	01:03:06	02:04	1	01:05	none	01:03:05
Variant	1st	2nd	3rd	4th	5th	6th	7th

Table IV.2 List of vowel variants for Amharic Braille characters in the 2nd version

V. Third Version Amharic Braille

6th		6th	
ሀ	01:02:05	ኽ	02:03:06
ል	01:02:03	ዉ	2:4:5:6
Q	Not apply	ዕ	Not apply
ም	01:03:04	ዝ	1:3:5:6
ሥ	02:03:04	ዥ	03:05:06
ር	1:2:3:5	ይ	1:3:4:5:6
ስ	Not apply	ድ	01:04:05
ሽ	01:04:06	ጅ	02:04:05
ቅ	1:2:3:4:5	ግ	1:2:4:5
ብ	01:02	ጥ	2:3:4:5:6
ት	2:3:4:5	ጭ	01:04
ች	01:06	ጵ	02:03:05
ኀ	Not apply	ፅ	1:2:3:4:6
ን	1:3:4:5	ጽ	Not apply
ኝ	03:04:06	ፍ	01:02:04
እ	*1:2:3:5:6	ý	1:2:3:4
ክ	01:03		

** Symbol indicates Braille code change made on the new version*

Table V.1 List of basic Amharic Braille characters in the 3nd version

Vowel	02:06	01:03:06	02:04	1	01:05	Independent	01:03:05
Character Variant	1st	2nd	3rd	4th	5th	6th	7th

Table V.2 List of vowel variants for Amharic Braille characters in the 3nd version

VI. Fourth Version Amharic Braille

Punctuation mark	Braille representation
.	3
:	6 and 3
፡	2:5
/	5 and 2
‐	4 and 1
:-	3:6
'	2
&	2:3
#	2:3:6
$	3:5:6
’	3:5:6 and 3
?	2:3:6
::	2:5:6
!	2:3:5
…	3 , 3and 3
( )	2:3:5:6
[	6 and 2:3:5:6
]	2:3:5:6 and 3
*	3:5 and 3:5
_	4:6 and 4:6
→	2:4:6, 2:5 and 2:5
←	2:5,2:5 and 1:3:5
↑	4:5:6 and 1:5
↓	4:5:6 and 3:5
እና/ወይም	3:4
=//=	6 and 3
X	1:3:4:6
$	4:5

Table VI.1 List of fourth Version Amharic Braille punctuation marks

I. Lookup Table

lookup={

'0':" ",'1':"ሀ",'2':"ሁ",'3':"ሂ",'4':"ሃ",'5':"ሄ",'6':"ህ",'7':"ሆ",

'8':"ለ",'9':"ሉ",'10':"ሊ",'11':"ላ",'12':"ሌ",'13':"ል",'14':"ሎ",

'15':"ሐ",'16':"ሑ",'17':"ሒ",'18':"ሓ",'19':"ሔ",'20':"ሕ",'21':"ሖ",

'22':"መ",'23':"ሙ",'24':"ሚ",'25':"ማ",'26':"ሜ",'27':"ም",'28':"ሞ",

'29':"ሠ",'30':"ሡ",'31':"ሢ",'32':"ሣ",'33':"ሤ",'34':"ሥ",'35':"ሦ",

'36':"ረ",'37':"ሩ",'38':"ሪ",'39':"ራ",'40':"ሬ",'41':"ር",'42':"ሮ",

'43':"ሰ",'44':"ሱ",'45':"ሲ",'46':"ሳ",'47':"ሴ",'48':"ስ",'49':"ሶ",

'50':"ሸ",'51':"ሹ",'52':"ሺ",'53':"ሻ",'54':"ሼ",'55':"ሽ",'56':"ሾ",

'57':"ቀ",'58':"ቁ",'59':"ቂ",'60':"ቃ",'61':"ቄ",'62':"ቅ",'63':"ቆ",

'64':"በ",'65':"ቡ",'66':"ቢ",'67':"ባ",'68':"ቤ",'69':"ብ",'70':"ቦ",

'71':"ተ",'72':"ቱ",'73':"ቲ",'74':"ታ",'75':"ቴ",'76':"ት",'77':"ቶ",

'78':"ቸ",'79':"ቹ",'80':"ቺ",'81':"ቻ",'82':"ቼ",'83':"ች",'84':"ቾ",

'85':"ኀ",'86':"ኁ",'87':"ኂ",'88':"ኃ",'89':"ኄ",'90':"ኅ",'91':"ኆ",

'92':"ነ",'93':"ኑ",'94':"ኒ",'95':"ና",'96':"ኔ",'97':"ን",'98':"ኖ",

'99':"ኘ",'100':"ኙ",'101':"ኚ",'102':"ኛ",'103':"ኜ",'104':"ኝ",'105':"ኞ",

'106':"አ",'107':"ኡ",'108':"ኢ",'109':"ኣ",'110':"ኤ",'111':"እ",'112':"ኦ",

'113':"ከ",'114':"ኩ",'115':"ኪ",'116':"ካ",'117':"ኬ",'118':"ክ",'119':"ኮ",

'120':"ኸ",'121':"ኹ",'122':"ኺ",'123':"ኻ",'124':"ኼ",'125':"ኽ",'126':"ኾ",

'127':"ወ",'128':"ዉ",'129':"ዊ",'130':"ዋ",'131':"ዌ",'132':"ው",'133':"ዎ",

'134':"ዐ",'135':"ዑ",'136':"ዒ",'137':"ዓ",'138':"ዔ",'139':"ዕ",'140':"ዖ",

'141':"ዘ",'142':"ዙ",'143':"ዚ",'144':"ዛ",'145':"ዜ",'146':"ዝ",'147':"ዞ",

'148':"ዠ",'149':"ዡ",'150':"ዢ",'151':"ዣ",'152':"ዤ",'153':"ዥ",'154':"ዦ",

'155':"የ",'156':"ዩ",'157':"ዪ",'158':"ያ",'159':"ዬ",'160':"ይ",'161':"ዮ",

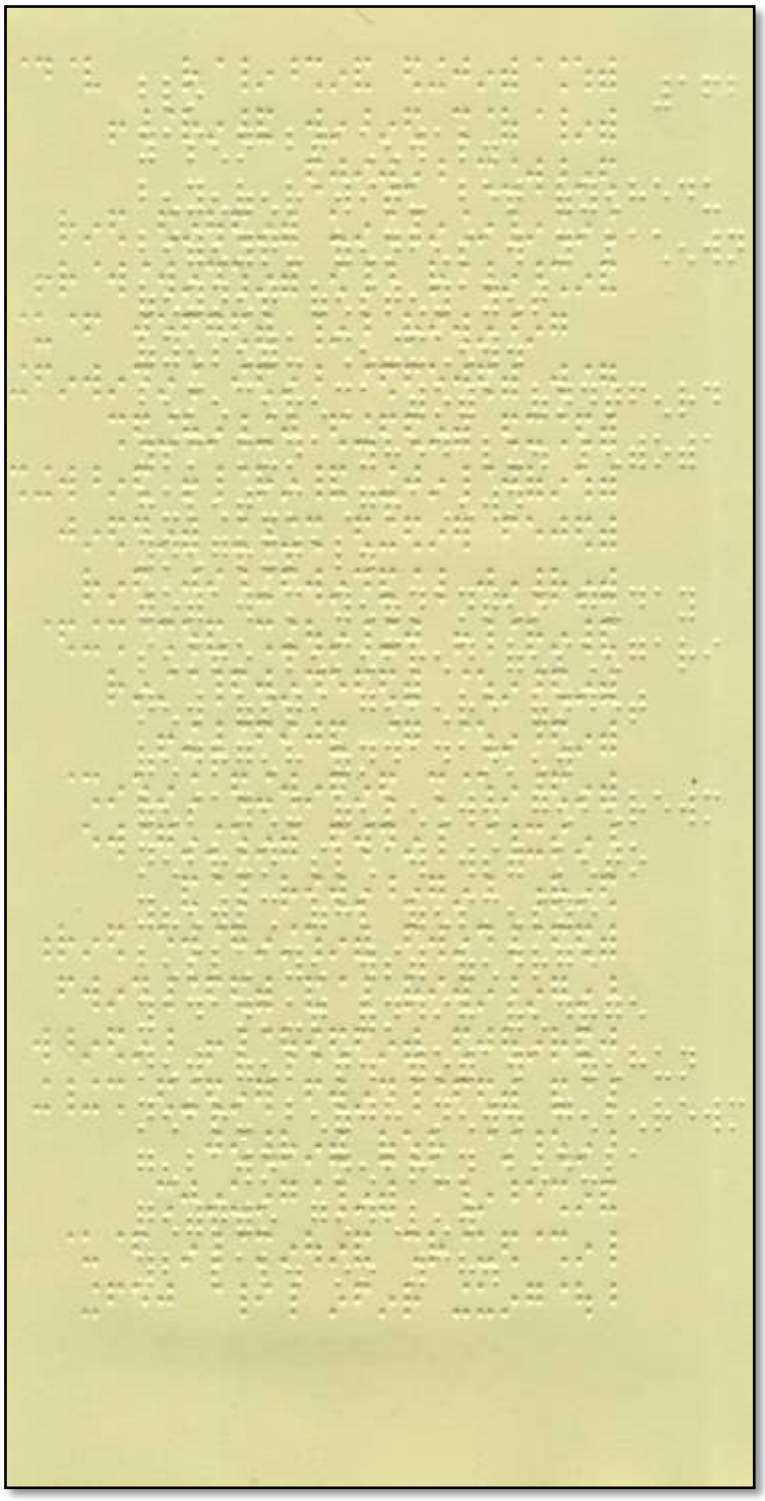
'162':"ደ",'163':"ዱ",'164':"ዲ",'165':"ዳ",'166':"ዴ",'167':"ድ",'168':"ዶ",

'169':"ጀ",'170':"ጁ",'171':"ጂ",'172':"ጃ",'173':"ጄ",'174':"ጅ",'175':"ጆ",

…. Continued on next page

'176':"ገ",'177':"ጉ",'178':"ጊ",'179':"ጋ",'180':"ጌ",'181':"ግ",'182':"ጎ",
'183':"ጠ",'184':"ጡ",'185':"ጢ",'186':"ጣ",'187':"ጤ",'188':"ጥ",'189':"ጦ",
'190':"ጨ",'191':"ጩ",'192':"ጪ",'193':"ጫ",'194':"ጬ",'195':"ጭ",'196':"ጮ",
'197':"ጸ",'198':"ጹ",'199':"ጺ",'200':"ጻ",'201':"ጼ",'202':"ጽ",'2403':"ጾ",
'204':"ጸ",'205':"ጹ",'206':"ጺ",'207':"ጻ",'208':"ጼ",'209':"ጽ",'210':"ጾ",
'211':"ፀ",'212':"ፁ",'213':"ፂ",'214':"ፃ",'215':"ፄ",'216':"ፅ",'217':"ፆ",
'218':"ፈ",'219':"ፉ",'220':"ፊ",'221':"ፋ",'222':"ፌ",'223':"ፍ",'224':"ፎ",
'225':"ፐ",'226':"ፑ",'227':"ፒ",'228':"ፓ",'229':"ፔ",'230':"ፕ",'231':"ፖ",
'232':"ፘ",'233':"ፙ",'234':"ፚ",'235':"፛",'236':"፜",'237':"፝",'238':"፞",
'239':"ሏ",'240':"ሗ",'241':"ሟ",'242':" ሿ",'243':"ሯ",'244':"ሷ",'245':"ቿ",
'246':"ቋ",'247':"ቧ",'248':"ቷ",'249':"ቿ",'250':"ኋ",'251':"ኗ",'252':"ኟ",
'253':"ኸ",'254':"ኻ",'255':"ዟ",'256':"ዧ",'257':"ዷ",'258':"ጇ",'259':"ጓ",
'260':"ጧ",'261':"ቈ",'262':"ቌ",'263':"ቔ",'264':"ቜ",'265':"ፗ",'266':"ፘ",
'282':"@",'283':"ፏ",'284':"ፗ",'285':"ፚ",'286':"ፘ",'287':"ፙ",'288':"ፚ",
'289':"፛",'290':"፜",'291':"፱",'292':"፲",'293':"0",'294':"1",'295':"2",
'296':"3",'297':"4",'298':"5",'299':"6",'300':"7",'301':"8",'302':"9",
'304':"፧",'305':"…",'306':"፤",'307':"",'308':"/",'309':"፦-",'310':"–",
'311':"*",'312':"!",'313':"፡",'314':"‹…",'315':"[…]",'316':"—",'317':"?",
'318':"«",'319':"፨",'320':"»",'321':"…›",'322':"(",'323':")",'324':"**",
'325':"+",'350':"\n"
}

II. Sample Amharic Braille And Translation Result



እሥካአሁን የምበላው 161
አላጣም ነበር ይለዋል።
ነብሩም በሬህን
ከከለከልከኝ አንተን ነው
የምበላው ብሎ አሥፈራራው።
ገበሬው በል እንግዲህ ዳኛ
ጋር እንሂድና በኔ ከፈረዱ
እንኳን አንዱን ሁለቱንም
እሠጥሃለሁ ይለዋል። ነብሩ
እንሂድ ቶኑን ሙሉ አፍህ
አውለህ ላይፈርዱቤህ ነው
ይለዋል። ሁለቱም ይሥማሙና
አንበሣ እንዲፈርድላቸው
አንበሣ ጋ ይሄዳሉ።
አንበሣም ነገሩን ከሠማ
በኋላ ለፍርድ አሥቸጋሪ
ነው፤ በገበሬው ላይ
እንዳልፈርድ አሥቸጋሪ ነው፤
በገበሬው ላይ እንዳልፈርድ
ህይወትህን አድኖታል። በነብር
ላይ እንዳልፈርድ ቶኑን ሙሉ
በሥልቻ ውሥጥ አውሎህ
የሚበላ አላገኘህም
ለማንኛውም የተሻለ ዳኛ
ብታገኙ ወደ ሌላ ሂዱ

ብሎ 162
ያሠናብታቸዋል።
በየአራዊቱ ቤት እየሄዱ
ቢጠይቁ አንበሣው ያላቸውን
አይነት አሥተያየት
እየሠጡ ሸኟቸው።
በመጨረሻ ጦጣ እንድትፈርድ
ይሥማሙና ይሄዳሉ። ለጦጣ
ሁለቱም ችግራቸውን
ይነግሯታል።
ጦጣዋም የእናንተን ጉዳይ
ሠምቶ ለመፍረድ አሥቸጋሪ
ነው እሥቲ ገበሬ እንዴት
አድርገህ እንዳዳንከው
አሣየኝ አለችው። ገበሬውም
ነብሩን ሥልቻው ውሥጥ ከቶ
ይሸውልሽ እህል አሥመሥዬ
በሥልቻየ ውሥጥ አድርጌ
ያዳንኩት እንዲህ አድርጌ
ነው አላት። ጦጣዋ አሁን
ነብሩ በእጅህ ነው። ገበሬውም
ገባውና ነብሩን በዱላ
ቀጥቅጦ ገደለ
ስባላል። 38።

Left - Sample double sided Amharic braille

Top right - front page translation of left image

Bottom right - back page translation of left image

Declaration

I declare that the thesis is my original work and has not been presented for a degree in any other university.

_____

June 5, 2015

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

_____

Dr. Dereje Teferi